Sam Williams

prefazione di Angelo Raffaele Meo

codice libero

FREE AS IN FREEDOM

Richard Stallman e la crociata per il software libero



APOGEO

Sam Williams

prefazione di Angelo Raffaele Meo

codice libero

FREE AS IN FREEDOM

Richard Stallman e la crociata per il software libero



APOGEO

Codice Libero (Free as in Freedom)

Richard Stallman e la crociata per il software libero

Sam Williams

APOGEO

Copyright © Free Software Foundation (http://www.fsf.org/). È garantito il permesso di copiare, distribuire e/o modificare questo documento seguendo i termini della Licenza per Documentazione Libera GNU, Versione 1.2 o ogni versione successiva pubblicata dalla Free Software Foundation; senza alcuna sezione non modificabile, senza testo di copertina e senza testo di quarta di copertina. Una copia della licenza è acclusa nella sezione intitolata “Licenza per Documentazione Libera GNU”.

Edizione italiana: Apogeo srl - Socio Unico Giangiacomo Feltrinelli Editore srl

ISBN: 978-88-503-1122-4

L'edizione cartacea è in vendita nelle migliori librerie.

~

Seguici su Twitter @apogeonline

Sito web: Apogeonline

A mia moglie Tracy

È successo raramente nella storia, ma è successo, che in un momento in cui sembrava che le regole del gioco fossero immutabili, che i vincitori fossero imbattibili e i perdenti senza alcuna possibilità di riscatto, un uomo solo, assolutamente privo di potere, ricchezze, fama, bellezza, amicizie, un uomo qualunque della specie innumerevole dei perdenti, riuscisse a sovvertire le regole del gioco e a far saltare il banco. La storia raccontata in questo libro è la storia di uno di quegli eventi rari ed è la storia di un uomo eccezionale che nasce perdente e diventa vincente, che non è bello ma affascinante, che non è simpatico ma è adorato come un dio; non ha amicizie vere ma conta migliaia di ammiratori; mette i lunghi capelli in bocca o nel piatto ove sta mangiando ma è conteso ospite alla tavola dei ricchi e dei potenti; non sorride mai ma si traveste da santo mettendosi in testa, a mo' di aureola, la superficie attiva di un hard disk della prima generazione; non ha i soldi per pagarsi una cena al ristorante ma ha sconvolto un mercato da migliaia di miliardi di dollari.

Il gioco è un comparto industriale delle dimensioni di quello dell'automobile, dominato da pochissimi attori, meno delle dita di due mani, in virtù della regola che chi vince prende tutto il piatto. È il settore industriale del software, ad altissima intensità di lavoro e poverissima intensità di capitali, che produce oggetti immateriali senza impiego di energia, puro spirito, frutto di sola intelligenza, fantasia, tenacia, miliardi di ore di lavoro.

Il protagonista della storia è Richard Stallman, figlio di un veterano della seconda guerra mondiale e di un insegnante. Bimbo povero ma felice, è improvvisamente sottoposto al dolore per il divorzio dei genitori e, in breve tempo, per la scomparsa dei nonni adorati. Quegli eventi lo trasformano in un ragazzo infelice, predisposto all'isolamento sociale ed emotivo, "al confine

dell’autismo”, come lo stesso Richard confessa. Lo sostengono un’eccezionale intelligenza, l’amore per lo studio e in particolare per le discipline scientifiche, l’amicizia dei compagni del college, l’unica vera casa della sua vita.

All’università diventa un hacker, che significa molto di più della bravura nel programmare un virus o violare un codice crittografico. La cultura hacker ha divertenti componenti goliardiche, un linguaggio rigorosamente scientifico che incorpora le primitive del linguaggio di programmazione LISP, un lessico variopinto che riflette un feroce atteggiamento critico verso i meccanismi del potere.

Ad esempio, “suit” è uno “scomodo abito da lavoro caratterizzato da uno *strangling device* che un hacker non indosserebbe mai”, ed è anche un individuo della specie umana vestito con uno “suit” a cui lo strangling device riduce molto l’irrorazione cerebrale.

Soprattutto, la cultura hacker è caratterizzata da un rigoroso senso morale, per cui un vero hacker non danneggerebbe mai un sistema informativo o un programma applicativo; e da una istintiva vocazione per la fraternità, la solidarietà e l’uguaglianza. Così Richard, che nella prima giovinezza aveva assunto atteggiamenti conservatori, si converte al credo progressista della madre.

Comunque, nella sua ideologia politica, la libertà è più importante dell’uguaglianza, per cui si terrà alla larga dal marxismo. Un giorno, in occasione di un convegno, si irritò molto con me, perché, come schematizzazione didattica scherzosa, avevo assimilato la rivoluzione del software libero a quella bolscevica e avevo affermato: “Stallman sta a Torvalds come Lenin sta a Stalin”. “Io non sono affatto comunista”, proclamò con forza quel giorno, forse anche per difendere il suo movimento dall’accusa ricorrente di “infocomunismo”, che suona come gravemente infamante nella società americana.

La madre di Stallman è ebrea, ma Richard si proclama ateo. Per un certo periodo di tempo, negli anni ’70, va in giro esibendo una spilla con la scritta “Processiamo Dio”. Spiega che se Dio fosse così potente da aver creato il

mondo senza far nulla per correggere i problemi, perché mai dovremmo adorarlo? Non sarebbe più ragionevole processarlo? E qualche volta dà una risposta contemporaneamente scherzosa e provocatoria: “Il mio nome è Jehovah. Ho un progetto speciale per la salvezza dell’universo, ma non posso rivelarlo. Devi avere fiducia in me, perché soltanto io sono in grado di vedere come stanno le cose. Se non avrai fede in me, ti metterò nella lista dei nemici per gettarti nell’abisso ove l’Ufficio Infernale delle Imposte passerà al vaglio le tue dichiarazioni dei redditi da qui all’eternità.”

Comunque, in contrasto con la professione di ateismo, i suoi atteggiamenti e le sue idee appaiono caratterizzate da una profonda, rigida religiosità. Per altro, soltanto gli uomini dotati di profonda, convinta religiosità, sono in grado di determinare le eccezionali trasformazioni delle regole del gioco. Come ogni autentico credente, Richard considera sacri i principi della sua fede e non accetta mai alcun tipo di compromesso. Anzi, nella sua intransigenza, diventa quasi violento quando lo si contraddice sui dogmi.

La storia della sua azione scientifica e politica inizia nei primi anni ’80, quando, come diffusamente raccontato nel libro, percepisce che i principi della libera diffusione e condivisione del software e delle idee che hanno caratterizzato i primi tre decenni dello sviluppo dell’informatica, sono violati da un nuovo atteggiamento delle aziende informatiche più importanti. Queste, non distribuiscono più il codice sorgente dei loro programmi per evitare la loro copiatura e non consentono, quindi, ai loro utilizzatori, l’adeguamento alle proprie esigenze e il miglioramento di funzionalità e prestazioni.

Il nuovo atteggiamento determinerà la nascita di un nuovo comparto industriale, caratterizzato oggi da fatturati annui dell’ordine del miliardo di dollari, ma avrà pesanti implicazioni negative come chiaramente percepito da Stallman: il condizionamento del progresso scientifico e tecnologico, l’inutile duplicazione di sforzi da parte di aziende concorrenti per realizzare gli stessi prodotti, danni gravi alla formazione per l’impossibilità di studiare le funzionalità e la struttura del software in assenza del codice sorgente.

Stallman comprende questi pericoli incombenti e per fronteggiarli fa una scelta radicale, in linea con i suoi principi e il suo stile di vita. Abbandona,

infatti, il suo mestiere sicuro di ricercatore e professionista dell'informatica e si butta, anima e corpo, nel progetto di realizzare un universo di programmi disponibili in linguaggio sorgente e aperti a un continuo progresso. Fonda la "Free Software Foundation" e dà a questa un preciso obiettivo importante: la realizzazione di un nuovo sistema operativo compatibile con UNIX, ma libero. A questo attribuisce il nome e il simbolo GNU, suggerito dalla definizione ricorsiva come nella tradizione hacker "Gnu is Not Unix". In altri termini, GNU non è lo UNIX coperto da copyright di A.T.T., ma ha le stesse funzionalità.

Nell'arco di 7-8 anni la Free Software Foundation costruisce un enorme patrimonio di programmi: compilatori, manipolatori di testi, strumenti di sviluppo, ma non completa lo sviluppo del nucleo del sistema operativo. Fortunatamente, nel 1991, uno studente finlandese, allora ventunenne, Linus Torvalds, decide di sviluppare il nucleo, "just for fun", come spiegherà in un libro dedicato al racconto affascinante del suo progetto. Parte da un prodotto didattico molto diffuso nelle università e chiama a raccolta un certo numero di progettisti e programmatori volontari. Così, in tempi incredibilmente brevi, l'opera di Stallman viene completata e nasce il nuovo sistema operativo GNU/Linux.

Il successo di questo nuovo prodotto è travolgente. Nell'arco di pochi anni, il nuovo sistema operativo si manifesta come un serio pericolo anche per Windows di Microsoft. Contemporaneamente, esplode il mondo dei prodotti e delle applicazioni di Internet, i cui componenti fondamentali – protocolli e strumenti di base – sono stati sviluppati applicando le logiche del software libero.

I successi inducono molte aziende importanti, come IBM e SUN, ad accettare almeno in parte il credo di Stallman. Ma i loro atteggiamenti tradiscono l'utilitarismo e non sono graditi a Richard, che denuncia con intransigenza la commistione peccaminosa di software libero e software proprietario. È l'innesco della feroce guerra civile – brillantemente descritta nel libro – fra gli intransigenti puristi di Stallman e altri protagonisti, più elastici, della storia del software libero.

Ero seduto un giorno a un convegno a fianco di Stallman, quando fu chiamato alla tribuna il rappresentante di una delle aziende peccaminose. Stallman aprì allora il suo computer portatile e si mise a lavorare ostentando indifferenza. Ma quell'atteggiamento apparve subito molto difficile da mantenere. Nei primi cinque minuti dell'intervento Richard iniziò a emettere alcuni grugniti di disapprovazione, per passare a interruzioni sempre più vivaci nei successivi cinque. Ad un certo punto perse definitivamente la pazienza, chiuse di scatto il portatile e si allontanò dalla sala con incedere provocatorio per rientrarvi solo alla fine dell'intervento dell'eresiarca.

Confesso di aver sempre ammirato Stallman e di aver condiviso le sue idee, ma di aver sempre ritenuto utopistico il suo progetto politico e scientifico. Quando alcuni anni fa proposi all'allora ministro della Ricerca Scientifica, Luigi Berlinguer, un programma nazionale di ricerca sul software libero, rimasto ignorato nei suoi cassetti, mi aspettavo soltanto risultati parziali e benefici, relativamente limitati per il settore dell'informatica italiana. Oggi sono invece convinto che Stallman sia molto vicino all'arco di trionfo.

A indurmi all'ottimismo sono molte novità clamorose: l'esplosione di Internet, strumento fondamentale della comunità dei programmatori liberi; la cresciuta importanza delle tecnologie "soffici" rispetto a quelle "dure"; le iniziative dei governi dei paesi più avanzati, compreso il nostro; l'interesse di quasi tutte le multinazionali dell'informatica e delle telecomunicazioni.

Soprattutto, la crescita esponenziale delle conoscenze ha portato all'esplosione della complessità e ha reso sempre più difficile per le multinazionali del software la competizione con quello che il filosofo Polanyi chiama "La Repubblica della Scienza", ossia la comunità dei ricercatori pubblici di tutto il mondo. Oggi il patrimonio collettivo del software libero è molto più ricco di quello del software proprietario.

Il gioco continua a essere del tipo: "Il più forte prende tutto", ma oggi il più forte è il mondo della collaborazione. Oggi collaborare è più conveniente che competere. Un nuovo modello di sviluppo basato sulla solidarietà è oggi possibile, non soltanto per una scelta razionale dell'umanità, ma anche in virtù

delle novità che ho elencato e della conseguente trasformazione radicale delle regole dell'economia mondiale.

Angelo Raffaele Meo
Torino, 19 dicembre 2002
Politecnico di Torino

L’opera di Richard M. Stallman parla letteralmente da sola. Dalla documentazione dei codici sorgenti ai contributi scritti agli interventi pubblici, poche persone hanno manifestato così tanta determinazione nel voler rendere perfettamente chiaro il proprio pensiero e la propria attività su tematiche particolari.

Un approccio così aperto – mi si vorrà perdonare il temporaneo ricorso a un aggettivo così atipico per Stallman – è sicuramente rinfrescante. In fondo viviamo in una società che tratta l’informazione, in particolar modo i dati personali, al pari di una preziosa merce di scambio. Nasce spontanea una domanda: perché mai qualcuno dovrebbe condividere una tale quantità di informazioni senza chiedere apparentemente nulla in cambio?

Come vedremo nei capitoli successivi, Stallman non condivide le sue parole e il suo lavoro per puro spirito d’altruismo. Ogni programma, discorso o intervento pubblico porta con sé un certo costo, pur se non del tipo che molti sono soliti pagare.

Non vado affermando ciò per mettere le mani avanti, quanto piuttosto come un’ammissione d’intenti. Dopo aver trascorso un anno a scavare negli eventi della storia personale di Stallman, fa decisamente paura voler procedere contro il suo modus operandi. “Non metterti mai a litigare con qualcuno che compra l’inchiostro a barili”, recita un vecchio adagio di Mark Twain. Nel caso di Stallman, non cercare mai di scrivere la biografia definitiva di qualcuno che rende di dominio pubblico ogni propria idea.

Per quei lettori che hanno deciso di dedicare qualche ora del proprio tempo all’esplorazione di questo volume, posso sostenere con fiducia che qui troverete fatti e citazioni impossibili da scovare in un articolo su Slashdot o

tramite una ricerca con Google. Tuttavia, anche l'accesso a questi eventi richiede il pagamento di una tariffa. Nel caso del libro in versione cartacea, potete pagare in maniera tradizionale, acquistandolo cioè nei normali canali di vendita. Invece per l'edizione elettronica si può pagare seguendo il metodo del software libero. Grazie alla decisione dell'editore O'Reilly & Associates, questo volume viene distribuito sotto la GNU Free Documentation License (Licenza per Documentazione Libera GNU), rendendo così possibile a chiunque migliorarne i contenuti o crearne versioni personalizzate per poi ridistribuirle sotto la medesima licenza.

Nel caso stiate leggendo l'edizione elettronica e preferiate la seconda opzione di pagamento, ovvero, se pensate di migliorare o espandere il libro a vantaggio di futuri lettori, ogni vostro commento è benvenuto. A partire dal giugno 2002, mi occuperò di pubblicare una versione HTML del libro sul sito *http://www.faifzilla.org*. Conto di aggiornarlo regolarmente e ampliarne i contenuti quando si renderà necessario. Se decidete di optare per questa seconda opzione, vi invito a leggere attentamente l'Appendice C del libro. Il testo specifica i vostri diritti sulla base dalla *GNU Free Documentation License* (Licenza per Documentazione Libera GNU).

Per quanti invece prevedono soltanto di sedersi tranquillamente a leggere, online o altrove, considero la vostra attenzione un'analoga e valida forma di pagamento. Nessuna sorpresa, però, se anche voi finirete per trovarvi alla ricerca di altre modalità con cui ricompensare la buona volontà che ha reso possibile la diffusione di questo testo.

Un'annotazione conclusiva: qui non si tratta soltanto di un lavoro giornalistico, ma anche di documentazione tecnica. Nel corso della stesura e della correzione del libro, il sottoscritto e i redattori della casa editrice hanno soppesato i commenti e i suggerimenti di varie persone coinvolte in quest'opera, incluso lo stesso Richard Stallman. Siamo consapevoli della presenza di numerosi dettagli tecnici che potrebbero giovarsi di informazioni complementari o più dettagliate. Poiché il volume viene distribuito sotto la GFDL, siamo pronti ad accogliere ogni possibile "patch", proprio come nel caso di un programma di software libero. I cambiamenti accettati verranno

prima diffusi elettronicamente per trovare infine spazio nelle future edizioni a stampa del volume. Potete inviare contributi e suggerimenti relativi all'edizione originale inglese, all'indirizzo e-mail: *sam@inow.com*

Commenti e domande

Per commenti e domande riguardanti il libro (sempre riguardo l'edizione inglese), si può contattare direttamente l'editore originale:

O'Reilly & Associates, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
USA
(800) 998-9938 (solo in USA o Canada)
(707) 829-0515 (internazionale/locale)
(707) 829-0104 (fax)

È stata predisposta una pagina web (in inglese) relativa al libro, dove verranno elencati errori, esempi e informazioni aggiuntive. Il sito include anche un forum aperto per discutere con l'autore e gli altri lettori. L'indirizzo del sito è:

http://www.oreilly.com/catalog/freedom/

Per commenti o domande di carattere tecnico sul libro, contattare via e-mail:

libri@apogeonline.com
bookquestions@oreilly.com (in inglese)

Per ulteriori informazioni su libri, conferenze, risorse e altre attività di Apogeo, si veda:

http://www.apogeonline.com

Ringraziamenti

Un ringraziamento speciale va a Henning Gutmann per aver sempre creduto in questo progetto. Lo stesso ad Aaron Oas per aver subito suggerito l'idea a Tracy. Grazie a Laurie Petrycki, Jeffrey Holcomb e tutti gli altri di O'Reilly &

Associates. Ringrazio Tim O’Reilly per aver sostenuto il mio lavoro. Grazie a quanti hanno rivisto le prime bozze: Bruce Perens, Eric Raymond, Eric Allman, Jon Orwant, Julie e Gerald Jay Sussman, Hal Abelson, Guy Steele. Spero che questa versione priva di refusi sia di vostro gradimento. Ringrazio Alice Lippman per le interviste, i biscotti e le fotografie. Grazie alla mia famiglia, Steve, Jane, Tish, Dave. E infine, *last but not least*: grazie a Richard Stallman per avere il fegato e la perseveranza di “farci vedere il codice”.

Sam Williams

Per volontà di una stampante

Temo i Greci, anche se portano doni.
– Virgilio, L’Eneide

La nuova stampante si era bloccata un’altra volta.

Richard M. Stallman, programmatore dello staff del laboratorio di intelligenza artificiale (AI Lab) presso il Massachusetts Institute of Technology, aveva fatto le spese di quel malfunzionamento. Un’ora dopo aver inviato un file di 50 pagine alla stampante laser, Stallman, 27 anni, decise d’interrompere una produttiva sessione di lavoro per andare a recuperare il documento. Arrivato in loco, trovò appena quattro pagine stampate. Al colmo della frustrazione, si accorse che quelle pagine erano di un altro utente, confermando così che il suo documento per intero e parte delle stampe di qualcun altro erano ancora intrappolati nel circuito elettrico della rete interna.

Per un programmatore il fatto di dover aspettare i comodi di una macchina è un rischio del mestiere, perciò Stallman decise di affrontare la cosa con un po’ di buonsenso.

Tuttavia, la differenza tra l’attesa del funzionamento *di* una macchina e quella *su* una macchina è alquanto considerevole. Non era la prima volta che era costretto a chinarsi sopra la stampante per seguire l’uscita delle pagine una alla volta. Abituato a trascorrere gran parte del giorno e della notte a migliorare l’efficienza degli apparecchi e dei relativi programmi di controllo, Stallman sentì il bisogno naturale di aprire quella macchina, dare un’attenta occhiata all’interno e isolare le radici del problema.

Purtroppo le sue capacità di programmatore informatico non si estendevano all’ambito dell’ingegneria meccanica. Mentre la stampante continuava a

produrre documenti freschi d’inchiostro, Stallman ebbe l’opportunità di riflettere su altre modalità per eludere il problema dell’intasamento dei fogli. Quanto tempo era passato da quando i membri del laboratorio avevano accolto a braccia aperte la nuova stampante? Stallman se lo andava chiedendo. La macchina era stata donata dalla Xerox Corporation. Si trattava di un prototipo d’avanguardia, versione modificata della diffusa fotocopiatrice Xerox. Soltanto che, anziché effettuare copie, seguiva le istruzioni di un apposito software diffuse tramite la rete locale interna per trasformare le informazioni in documenti di qualità professionale. Creata dagli ingegneri dello Xerox Palo Alto Research Center, un centro di fama mondiale, rappresentava, per dirla semplicemente, un assaggio della rivoluzione della stampa desktop che avrebbe poi raggiunto il resto dell’industria informatica entro la fine del decennio.

Guidati da una passione innata a giocare con le migliori apparecchiature esistenti, i programmatori del laboratorio di intelligenza artificiale si erano dati immediatamente da fare per integrare la nuova macchina all’interno della sofisticata infrastruttura informatica. I risultati apparvero subito positivi. Al contrario della vecchia stampante laser, la nuova Xerox era assai veloce. Le pagine volavano fuori al ritmo di una al secondo, trasformando un lavoro di 20 minuti in uno da due minuti. Il lavoro risultava anche più preciso. I cerchi venivano stampati correttamente, non come ovali. Le linee rette rimanevano tali e non diventavano sinusoidali. Sotto tutti gli aspetti e le intenzioni, si trattava di un regalo troppo bello da rifiutare.

Soltanto qualche settimana dopo il suo arrivo iniziarono a evidenziarsi i primi difetti. Tra i problemi maggiori vi era l’insita predisposizione all’inceppamento della carta. Le menti ingegneristiche di quei programmatori compresero al volo la questione: in quanto fotocopiatrice, generalmente la macchina funzionava con la supervisione diretta di un operatore. Ci sarebbe sempre stato qualcuno, insomma, pronto a risolvere problemi di carta inceppata – così dovevano aver ragionato gli ingegneri della Xerox, dedicandosi perciò alla risoluzione di altre noie. In termini tecnici, il sistema di funzionamento faceva affidamento sulla diligenza dell’utente.

Nell’apportare le modifiche necessarie per l’utilizzo come stampante, gli ingegneri della Xerox avevano modificato anche il rapporto utente-macchina in maniera sottile ma profonda. Invece di prevedere l’intervento di un singolo operatore, la macchina sottostava a un intera popolazione di operatori collegati in rete. Anziché starvi letteralmente sopra, un utente di tale rete inviava il comando di stampa attraverso una lunga catena di macchine, contando sul fatto che il contenuto desiderato giungesse a destinazione in maniera corretta e appropriata. Solamente alla fine, quando si recava a ritirare l’output finale, poteva rendersi conto di quanto poco del contenuto fosse stato effettivamente stampato.

Stallman era stato tra i primi a identificare il problema e il primo a suggerirne la soluzione. Qualche anno addietro, quando il laboratorio utilizzava ancora la vecchia stampante, Stallman aveva risolto un problema analogo “aprendo” il software di controllo della stampante collegata alla macchina PDP-11 del laboratorio. Pur non riuscendo a impedire l’inceppamento della carta, egli inserì nel programma un comando che ordinava al PDP-11 di verificare periodicamente i vari meccanismi e inviarne relazione al PDP-10, il computer centrale del laboratorio. Per esser certo che la negligenza di qualcuno non danneggiasse tutta una serie di stampe in corso, Stallman aggiunse inoltre un comando che istruiva il PDP-10 a notificare l’eventuale blocco della stampante a ogni utente con una stampa in attesa. Si trattava di una semplice nota, qualcosa del tipo “La stampante è bloccata, si prega di sistemarla”; e poiché tale nota raggiungeva proprio coloro che avevano maggior necessità di risolvere il problema, esistevano buone probabilità che ciò avvenisse in tempi brevi.

Come per ogni soluzione, quella di Stallman era indiretta ma elegante. Non risolveva la parte meccanica del problema, però ci andava vicino chiudendo il cerchio informativo tra utente e macchina. Grazie all’aggiunta di poche righe di codice, lo staff del laboratorio di intelligenza artificiale poteva così eliminare una perdita di tempo pari a 10-15 minuti ogni settimana per correre avanti e indietro a controllare la stampante. In termini di programmazione, la

soluzione di Stallman traeva vantaggio dall'intelligenza amplificata che sosteneva la rete locale nel suo complesso.

"Se ricevevi quel messaggio, non potevi dare per scontato che qualcun altro avrebbe risolto il problema", spiega Stallman, sottolineando la logica applicata in quell'occasione. "Dovevi controllare di persona. Un minuto o due dopo il mancato funzionamento, i due o tre utenti che avevano ricevuto l'avviso erano andati a risolvere il problema. Di questi, generalmente almeno uno sapeva come fare per risistemare i fogli."

Questo tipo di soluzioni indirette rappresentavano una sorta di marchio del laboratorio di intelligenza artificiale e dei programmatori originari che lo popolavano. Anzi, i migliori tra costoro disdegnavano lo stesso termine programmatore, preferendo invece l'appellativo gergale di hacker. Una qualifica professionale che indicava uno spettro di attività alquanto ampio – qualsiasi cosa compresa tra il caos creativo e il miglioramento di software e sistemi esistenti. Nel termine era tuttavia implicita la vecchia idea di una certa ingenuità *yankee*. Per essere un hacker, bisognava accettare la filosofia secondo cui la scrittura di un programma era soltanto l'inizio. Stava poi nella capacità di migliorarlo la vera prova di abilità per un hacker. [1]

Questa filosofia costituiva il motivo principale che spingeva aziende come la Xerox a seguire la politica di donare macchine e programmi a quelle entità che costituivano il terreno abituale degli hacker. Se questi ultimi riuscivano a migliorarne il software, tali migliorie sarebbero state riutilizzate dalle stesse aziende, incorporandole nelle nuove versioni dei programmi destinate al mercato commerciale. Nella terminologia imprenditoriale, gli hacker rappresentavano una sorta di capitale comunitario, un'unità aggiuntiva per la ricerca e lo sviluppo disponibile a costi minimi.

Fu grazie a questa filosofia del dare e del ricevere che quando Stallman isolò il difetto di carta inceppata nella stampante laser Xerox, non si fece certo prendere dal panico. Semplicemente, si mise alla ricerca di un modo per aggiornare le vecchie procedure e adattarle al nuovo sistema. Tuttavia, mentre si apprestava a dare un'occhiata al software della stampante Stallman scoprì qualcosa di preoccupante. La macchina non sembrava essere dotata di alcun

software, almeno nulla che lui stesso o gli altri programmatori fossero in grado di leggere. Fino ad allora, la maggioranza delle aziende riteneva una forma di cortesia la pubblicazione in formato testuale dei codici sorgenti che documentavano i singoli comandi su cui era basato il comportamento della macchina. In questo caso, la Xerox aveva fornito i file in formato precompilato, ovvero binario. I programmatori erano liberi di aprirli, ma a meno che non fossero degli esperti nel decifrare sequenze infinite di zero e di uno, il testo finale sarebbe rimasto una sfilza di caratteri incomprensibili.

Pur sapendone parecchio d'informatica, Stallman non poteva considerarsi certo un esperto nella decifrazione dei file binari. L'esperienza acquisita come hacker gli consentiva, però, di fare affidamento su altre risorse. Il concetto della condivisione dell'informazione occupava un ruolo talmente centrale nella cultura hacker che sarebbe stata solo questione di tempo prima che un collega di qualche laboratorio universitario o di altra struttura aziendale riuscisse a fornirgli la versione testuale dei sorgenti relativi a quel software.

Dopo i primi blocchi della stampante, Stallman si consolò ripensando a una situazione analoga sperimentata qualche anno addietro. Il laboratorio aveva avuto bisogno di un programma sulla rete per aiutare il PDP-11 ad operare in maniera più efficiente con il PDP-10. Gli hacker del laboratorio potevano facilmente cavarsela da soli, ma Stallman, avendo studiato ad Harvard, si ricordò di un programma analogo scritto dai programmatori del dipartimento d'informatica di quell'università. Tale dipartimento usava un computer del medesimo modello, il PDP-10, pur se dotato di un diverso sistema operativo. La procedure in vigore nel laboratorio richiedevano altresì che tutti i programmi installati sul PDP-10 fossero accompagnati dalla pubblicazione dei relativi codici sorgenti.

Potendo contare sul pieno accesso al laboratorio d'informatica di Harvard, Stallman vi si recò, fece una copia di quei codici e li portò con sé al laboratorio di intelligenza artificiale. Qui prese a riscriverne alcune parti onde renderle compatibili con il sistema operativo locale. Senza troppo clamore, il laboratorio di intelligenza artificiale richiuse una falla importante all'interno della propria infrastruttura informatica. Stallman aggiunse persino alcune

funzioni assenti nel programma originale di Harvard, rendendolo così ancora più utile. “Ne facemmo uso per parecchi anni senza problemi”, sostiene Stallman.

Nella concezione di un programmatore degli anni ’70, questa operazione era l’equivalente software di un vicino di casa che veniva a chiederci in prestito un utensile o una tazza di zucchero. Nel caso del prestito di una copia del software per il laboratorio di intelligenza artificiale, l’unica differenza stava nel fatto che Stallman non aveva privato gli hacker di Harvard dell’utilizzo del programma originale. Anzi, erano stati proprio costoro a trarne vantaggio, considerate le nuove funzionalità aggiuntevi dallo stesso Stallman, funzionalità che rimanevano a loro completa disposizione. Anche se nessuno ad Harvard richiese mai quel programma, Stallman ricorda che lo fece invece uno sviluppatore della società privata Bolt, Beranek & Newman, il quale vi apportò ulteriori aggiunte poi reintegrate da Stallman nell’archivio dei codici sorgenti organizzato presso lo stesso laboratorio di intelligenza artificiale.

“Normalmente ogni programma seguiva uno sviluppo analogo a quello della pianificazione cittadina”, dice Stallman in riferimento all’infrastruttura informatica in vigore in quel laboratorio. “Alcune aree venivano interamente sostituite e ricostruite. Vi si aggiungevano nuovi elementi. Ma era sempre possibile isolarne qualche sezione e dire, ‘A giudicare dallo stile, queste stringhe sono state scritte all’inizio degli anni ’60, mentre queste altre risalgono a metà anni ’70’.”

Grazie a questo semplice sistema di crescita intellettuale, gli hacker operanti nel laboratorio di intelligenza artificiale e in altri ambiti riuscirono a mettere in piedi prodotti particolarmente affidabili. Fu ricorrendo a tale strategia che, sulla costa occidentale degli Stati Uniti, i ricercatori della University of California di Berkeley, in collaborazione con alcuni ingegneri di primo livello presso la AT&T, realizzarono un intero sistema operativo. Chiamato Unix, con un gioco di parole riferito ad un sistema operativo più vecchio e più rispettabile in ambito accademico noto come Multics, il software venne messo a disposizione di qualunque programmatore disposto ad

accollarsi le spese per copiarlo su un nuovo nastro magnetico e per la successiva spedizione.

Non tutti gli sviluppatori coinvolti in questa cultura amavano autodefinirsi hacker, ma la vasta maggioranza condivideva i sentimenti di Richard M. Stallman. Se un programma o una correzione software si dimostravano sufficientemente validi da risolvere i problemi dell'autore, potevano risolvere anche quelli di qualcun altro. Perché allora non condividerli anche soltanto per guadagnarsi un buon karma?

Inizialmente il fatto che la Xerox avesse deciso di non condividere i propri sorgenti non provocò alcun fastidio. Mentre andava alla ricerca di quei file, Stallman sostiene di non essersi minimamente preoccupato di contattare la Xerox: "Ci avevano già regalato la stampante, perché mai importunarli ancora?"

Quando però si accorse che i file non comparivano da nessuna parte, Stallman iniziò a farsi sospettoso. L'anno precedente aveva già subito un brutto colpo da parte di un laureando presso la Carnegie Mellon University. Lo studente, Brian Reid, aveva realizzato un utile programma di formattazione testi chiamato Scribe. Si trattava di uno dei primi software grazie al quale l'utente poteva scegliere font e stili di un documento da inviare attraverso una rete, un antesignano del linguaggio HTML, la lingua franca del World Wide Web. Nel 1979, Reid decise di vendere Scribe ad un'azienda produttrice di software nell'area di Pittsburgh, la Unilogic. Ormai vicino alla laurea, Reid stava semplicemente cercando una soluzione per mollare il programma a qualche sviluppatore disposto ad accollarsi l'onere di impedirne la caduta nel pubblico dominio. Per addolcire la trattativa, Reid si dichiarò inoltre d'accordo all'inserimento di una serie di funzioni con limiti temporali – "bombe a tempo", nel gergo dei programmatori – grazie alle quali le versioni gratuite del programma smettevano di funzionare dopo 90 giorni. Onde evitare la disattivazione, gli utenti dovevano così pagare una tariffa al produttore, il quale a sua volta avrebbe fornito loro il codice necessario a disattivare il blocco.

Secondo Reid, era un buon affare per entrambe le parti. Scribe non si sarebbe eclissato tra i programmi di pubblico dominio, mentre la Unilogic avrebbe recuperato il denaro investito. Per Stallman si trattava invece del puro e semplice tradimento dell'etica del programmatore. Anziché onorare il concetto della condivisione, Reid collaborava con l'industria costringendo gli sviluppatori a pagare per avere accesso all'informazione.

Con il trascorrere delle settimane, mentre i tentativi di rintracciare il codice sorgente della stampante apparivano senza via d'uscita, Stallman sentì incombere un analogo scenario di scambio denaro-codice. Tuttavia, prima di poter dire o fare qualunque cosa al riguardo, ecco la rete dei programmatori produrre una buona notizia. Girava voce che un ricercatore presso il dipartimento d'informatica della Carnegie Mellon University si era appena dimesso dallo Xerox Palo Alto Research Center. E non soltanto aveva lavorato sulla stampante in questione, ma, stando alle stesse voci, se ne stava tuttora occupando come parte dei progetti assegnatigli alla Carnegie Mellon.

Mettendo da parte ogni sospetto iniziale, Stallman decise di andarlo a trovare durante la sua visita successiva al campus di quell'università. Non dovette attendere a lungo. Anche la Carnegie Mellon aveva un laboratorio specializzato in ricerche sull'intelligenza artificiale, e nel giro di qualche mese egli dovette recarvisi per questioni di lavoro. Durante la visita, fece in modo di fermarsi al dipartimento d'informatica. Qualcuno dei ricercatori gli indicò l'ufficio del docente incaricato del progetto Xerox. Entrando in quell'ufficio, lo trovò al lavoro.

Nel tipico confronto tra ingegneri, la conversazione che ne seguì fu cordiale ma schietta. Dopo essersi rapidamente presentato come un collega del MIT, Stallman gli chiese una copia dei sorgenti della stampante laser in modo da poterli adattare al PDP-11. Con sua sorpresa, il professore gli oppose un netto rifiuto.

"Mi disse di aver promesso che non me ne avrebbe fornito una copia", sostiene Stallman.

A volte la memoria gioca brutti scherzi. Vent'anni dopo quel fatto, il ricordo mentale di Stallman è notoriamente pieno di lacune. Non soltanto non

rammenta la motivazione alla base del viaggio di lavoro né l'anno in cui questo avvenne, ma neppure riesce a descrivere in qualche modo la controparte con cui tenne quello scambio di battute. A sentire Reid, probabilmente la persona che avrebbe potuto soddisfare la richiesta di Stallman era Robert Sproull, ex ricercatore presso lo Xerox PARC e attuale direttore dei Sun Laboratories, divisione del conglomerato informatico Sun Microsystems. Negli anni '70, mentre lavorava allo Xerox PARC, Sproull aveva diretto il gruppo di sviluppo del software per la stampante laser in questione. Verso il 1980, assunse il ruolo di ricercatore informatico presso la Carnegie Mellon, dove tra i vari progetti continuò ad occuparsi ancora di quella stampante.

"Il codice oggetto della richiesta di Stallman rappresentava la parte migliore del software che Sproull aveva scritto nell'anno precedente al suo arrivo alla Carnegie Mellon", ricorda Reid. "Credo che Sproull si trovasse alla Carnegie Mellon da meno di un mese quando arrivò quella richiesta."

Interrogato direttamente sulla questione, tuttavia, Sproull cade dalle nuvole. "Non posso offrire alcun commento", mi scrive via e-mail. "Non ho assolutamente il minimo ricordo dell'incidente".

Vista l'amnesia di entrambe le persone coinvolte in quella breve conversazione, inclusa perfino la veridicità sull'effettivo svolgimento, è difficile valutare la rudezza del rifiuto opposto da Sproull, almeno nella ricostruzione di Stallman. Nei suoi interventi pubblici, quest'ultimo ha più volte fatto esplicito riferimento all'incidente, sottolineando come il rifiuto di Sproull fosse dovuto al cosiddetto accordo di non divulgazione (non-disclosure agreement), norma contrattuale stipulata tra la Xerox Corporation e il proprio dipendente, sulla base della quale a quest'ultimo veniva garantito accesso ai codici sorgenti del software in cambio della promessa di mantenere la loro segretezza. Oggi clausola standard nei contratti dell'industria informatica, a quel tempo l'accordo di non divulgazione costituiva una novità, un segnale del valore commerciale acquisito per la Xerox sia dalla stampante sia dalle informazioni necessarie al relativo funzionamento. "In quel periodo la Xerox stava cercando di lanciare la stampante laser come prodotto

commerciale", spiega Reid. "Per loro sarebbe stato folle regalarne in giro i codici sorgenti".

Invece Stallman considerava la faccenda in maniera diametralmente opposta. Si trattava del rifiuto da parte della Xerox e di Sproull, o chiunque fosse la persona che quel giorno si trovò di fronte, di far parte di un sistema che fino ad allora aveva spinto gli sviluppatori a considerare i programmi al pari di risorse comuni. Come un contadino i cui canali d'irrigazione vecchi di secoli si fossero improvvisamente prosciugati, Stallman aveva seguito quei canali fino alla sorgente soltanto per scoprire una nuova diga idroelettrica in cui spiccava il logo della Xerox.

Non fu semplice per Stallman mandar giù il fatto che la Xerox avesse coinvolto alcuni programmatori in questo nuovo e stravagante sistema di segretezza forzata. All'inizio riuscì a comprendere soltanto la natura personale di quel rifiuto. Dato che in genere gli incontri faccia a faccia lo mettevano a disagio e in imbarazzo, la visita non annunciata di Stallman ad un collega poteva essere intesa come una dimostrazione di socialità tra buoni vicini. Ma ora il rifiuto bruciava come un grave atto di scortesia. "Ero talmente arrabbiato che non riuscii neppure ad esprimerlo. Così mi sono girato e sono uscito, senz'aggiungere neanche una parola", rammenta Stallman. "Può anche darsi che abbia sbattuto la porta, chissà? Tutto quel che ricordo è che volevo andarmene subito via".

Vent'anni dopo quel fatto, la rabbia brucia ancora tanto da spingere Stallman a ritenere l'evento un punto di svolta sostanziale. Nei pochi mesi successivi, Stallman e gli hacker del laboratorio di intelligenza artificiale sarebbero stati colpiti da un'ulteriore serie di eventi talmente negativi da far impallidire la tensione di quei 30 secondi in un remoto ufficio della Carnegie Mellon. E tuttavia, quando si tratta di individuare quei singoli episodi che avrebbero trasformato Stallman da hacker solitario, istintivamente sospettoso verso ogni autorità centralizzata, ad attivista animatore di una crociata che applica le nozioni tradizionali di libertà, uguaglianza e fraternità al mondo dello sviluppo software, lo stesso Stallman non esita a isolare l'incontro alla Carnegie Mellon, dedicandogli un'attenzione tutta particolare.

“L’evento mi spinse a riflettere più a fondo su quel che andavo già pensando”, spiega Stallman. “Avevo già l’opinione che il software avrebbe dovuto essere condiviso, ma non sapevo bene come procedere. Non avevo le idee chiare e organizzate al punto da poterle esprimere in maniera concisa al resto del mondo”.

Sebbene alcuni episodi precedenti avessero già suscitato le ire di Stallman, egli sostiene come fino all’incontro della Carnegie Mellon non si fosse reso conto del fatto che quella serie di eventi andasse infiltrandosi in una cultura considerata sacrosanta da molto tempo. Facendo parte di un’élite di programmatori in una della maggiori istituzioni al mondo, Stallman aveva coscientemente deciso di ignorare ogni patto o compromesso sottoscritto dai suoi colleghi fin tanto che ciò non interferiva con il proprio lavoro. Fino all’arrivo della stampante laser della Xerox, egli si era accontentato di dedicarsi a macchine e programmi a malapena tollerati da altri utenti. Nelle rare occasioni in cui tali programmi uscirono dalle stanze del laboratorio di intelligenza artificiale – quando, per esempio, il laboratorio sostituì il venerabile sistema operativo Incompatible Time Sharing con una variante commerciale chiamata TOPS 20 – Stallman e gli altri hacker erano stati liberi di riscrivere, ricostruire e rinominare il software sulla base del gusto personale.

Ora che la stampante laser si era insinuata nella rete del laboratorio, tuttavia, qualcosa era cambiato. La macchina funzionava bene, al di là di occasionali problemi di carta inceppata, ma era scomparsa la possibilità di modificarla secondo il gusto personale. Dal punto di vista dell’industria del software nel suo complesso, la stampante rappresentava un campanello d’allarme. Il software era divenuto un bene talmente prezioso che le aziende non sentivano la necessità di diffonderne i sorgenti, soprattutto quando la loro pubblicazione significava offrire ai potenziali concorrenti la possibilità di duplicare qualche programma a costi irrisori.

Dal punto di vista di Stallman, la stampante era una sorta di cavallo di Troia. Dopo un decennio di tentativi falliti, il software sotto proprietà privata – in seguito gli hacker introdurranno il termine “proprietario” – aveva messo

piede nel laboratorio di intelligenza artificiale grazie al più subdolo dei metodi. Vi era entrato sotto le mentite spoglie di un regalo.

Il fatto che Xerox offrisse ad alcuni programmatori l'accesso a doni ulteriori in cambio del loro riserbo era un'amara scoperta, tuttavia Stallman non può fare a meno di notare come egli stesso, nel caso gli fosse stato proposto un simile scambio nei suoi anni giovanili, probabilmente avrebbe accettato l'offerta della Xerox Corporation. Però l'imbarazzante incontro alla Carnegie Mellon provocò un salutare effetto sulla sua apatia morale. Non soltanto l'evento gli fornì la rabbia necessaria per considerare ogni futuro regalo con una buona dose di sospetto, ma lo costrinse a porsi una domanda inquietante: cosa accadrebbe se un giorno un collega hacker entrasse nell'ufficio di Stallman e improvvisamente egli dovesse trovarsi a rifiutare un'analoga richiesta del codice sorgente?

"Era la prima volta che m'imbattevo in una clausola di non divulgazione, e mi resi immediatamente conto come questi accordi producano delle vittime", sostiene con convinzione Stallman. "In questo caso la vittima ero io. Io e il mio laboratorio".

Fu questa una lezione che avrebbe accompagnato Stallman lungo i tumultuosi anni '80, un decennio nel corso del quale parecchi colleghi al MIT avrebbero abbandonato il laboratorio di intelligenza artificiale per firmare analoghi accordi di non-divulgazione a livello personale. Poiché gran parte di questi accordi prevedono un limite di durata, gli hacker che decisero di sottoscriverli non ci pensarono su due volte. Prima o poi, questo il loro ragionamento, il software diverrà di pubblico dominio. D'altra parte, la promessa di mantenere segreto il programma nel primo periodo di sviluppo faceva parte di quel compromesso che permetteva agli hacker di lavorare sui progetti migliori. Anche se per Stallman ciò non rappresentò altro che il primo passo verso un terreno molto sdrucciolevole.

"Quando ricevevo un invito a tradire in tal modo tutti i miei colleghi, sentivo riaffiorare la rabbia provata quando qualcun altro aveva fatto lo stesso con me e l'intero laboratorio", insiste Stallman. "Allora rispondevo, 'Grazie

mille per l’offerta di questo bel pacchetto software, ma non posso accettarlo alle condizioni che mi avete richiesto, per cui ne farò a meno’.”

Come avrebbe imparato presto, il rifiuto implicava molto di più di qualche sacrificio personale. Significava piuttosto l’isolamento dagli altri hacker che, pur condividendo un’analoga avversione per i segreti, tendevano a esprimerla in maniera più elastica a livello etico. Non trascorse molto tempo prima che Stallman, sempre più isolato all’interno del suo stesso laboratorio, prese ad autodefinirsi “l’ultimo vero hacker”, allontanandosi sempre più da un mercato caduto sotto il dominio del software proprietario. Il rifiuto opposto a una qualsiasi richiesta per il codice sorgente, decise Stallman, rappresentava non soltanto il tradimento della missione scientifica che aveva alimentato lo sviluppo del software a partire dalla fine della Seconda Guerra Mondiale, ma anche una violazione della regola aurea, il fondamento morale che imponeva di non fare altri quello che non vorresti fosse fatto a te.

Da qui l’importanza della stampante laser e del successivo incontro alla Carnegie Mellon. In mancanza di questa circostanza, afferma Stallman, la sua vita avrebbe potuto seguire un percorso più ordinario, alla ricerca di un equilibrio tra la ricchezza di un programmatore commerciale e la frustrazione finale di un’esistenza spesa a scrivere codice invisibile. Non si sarebbe evidenziato alcun senso di chiarezza, nessuna urgenza di affrontare un problema di cui nessun altro pareva curarsi. Fatto ancor più importante, non avrebbe preso corpo quella giusta ira, un sentimento che, lo vedremo più avanti, rimarrà il motore propulsore della carriera di Stallman tanto quanto l’ideologia politica o le considerazioni etiche.

“Da quel giorno in poi decisi che non avrei mai potuto partecipare a quel sistema”, dice Stallman riferendosi alla pratica di barattare la libertà personale per pura convenienza – così descrive gli accordi di non divulgazione – ma anche e soprattutto alla cultura globale che incoraggiava simili accordi moralmente dubbi. “Presi la decisione di non causare mai altre vittime, come era invece successo a me.”

[1] Per maggiori dettagli sul termine “hacker”, si veda l’Appendice B.

2001: Odissea di un hacker

Il dipartimento d'informatica della New York University si trova all'interno della Warren Weaver Hall, edificio simile a una fortezza situato due isolati a destra dopo Washington Square Park. Una serie di condotte di dimensioni industriali per l'aria condizionata creano tutt'intorno una cortina di aria calda, scoraggiando la presenza di fannulloni e questuanti. I visitatori che riescono a superare quella cortina si trovano davanti a un'altra formidabile barriera, il bancone con gli addetti alla sicurezza, posto immediatamente nell'atrio dell'unico ingresso all'edificio.

Superati i controlli, l'atmosfera si fa un po' più rilassata. Eppure colpiscono i numerosi cartelli sparsi lungo tutto il primo piano che informano sui pericoli di porte poco sicure e sulle uscite d'emergenza in caso d'incendio. Considerati nel loro insieme, quegli avvisi implicano un ammonimento preciso: perfino nel contesto relativamente tranquillo della New York precedente l'11 settembre 2001, nessuno può dirsi troppo prudente o sospettoso.

Uno scenario che si pone come stimolante contraltare al crescente numero di visitatori che va raccogliendosi nell'atrio interno. Alcuni paiono studenti della stessa università. La maggior parte sembra come raccolta davanti all'ingresso di un locale notturno nell'attesa di un eccitante concerto. Per una breve mattinata, le masse hanno conquistato la Warren Weaver Hall, lasciando gli addetti alla sicurezza con null'altro da fare se non seguire la TV e indicare con una scrollata di spalle l'adiacente auditorium ogni volta che qualcuno chiede qualcosa sul "discorso".

Una volta all'interno dell'auditorium, ci si trova davanti a colui che ha imposto questa temporanea chiusura delle procedure di sicurezza nell'edificio. Si tratta di Richard M. Stallman, fondatore del progetto GNU, presidente

fondatore della Free Software Foundation, vincitore nel 1990 della MacArthur Fellowship, del premio Grace Murray Hopper assegnato (nello stesso anno) dalla Association of Computing Machinery, nonché nel 2001 co-vincitore del premio Takeda della Takeda Foundation, e già hacker presso il laboratorio di intelligenza artificiale. Come annunciato su una schiera di siti web del mondo hacker, incluso quello dello stesso progetto GNU (*http://www.gnu.org*), Stallman si trova a Manhattan, suo luogo di nascita, per tenere un atteso intervento in replica alla recente campagna lanciata dalla Microsoft Corporation contro la GNU General Public License.

Tema del discorso di Stallman è la storia e il futuro del movimento del software libero. Significativo il luogo prescelto: meno di un mese addietro, il senior vice president di Microsoft, Craig Mundie, era intervenuto alla vicina Stern School of Business della New York University mirando a distruggere la General Public License, anche nota come GPL, documento legale elaborato da Stallman 16 anni prima. Messa a punto per ribattere alla crescente ondata di segretezza sul software che stava conquistando l'industria informatica – ondata di cui Stallman si rese conto per la prima volta nel corso degli anni '80 con i problemi relativi alla stampante laser della Xerox – la GPL si era evoluta in uno strumento essenziale per la comunità del software libero. In sintesi, la GPL vincola i programmi software a una sorta di proprietà collettiva – quello che oggi gli studiosi di diritto vanno definendo come "bene comune digitale" – facendo leva sul peso legale del copyright. Una volta vincolati, i programmi divengono inamovibili. Ogni loro versione successiva deve contenere la medesima protezione sul copyright – perfino quei programmi derivati che includono soltanto una porzione minima del codice originario. Questo il motivo per cui qualcuno nell'industria del software ha definito la GPL una licenza "virale", nel senso che si autoriproduce all'interno di ogni software con cui viene a contatto[1].

In un'economia dell'informazione sempre più dipendente dal software e sempre più legata ai relativi standard, la GPL è divenuta un punto di riferimento. Perfino quelle aziende che una volta la schernivano definendola il socialismo del software, sono state costrette a riconoscerne i benefici.

Linux, il kernel in stile Unix sviluppato dallo studente finlandese Linus Torvalds nel 1991, viene rilasciato sotto la licenza GPL, al pari di gran parte degli strumenti di programmazione più diffusi al mondo: GNU Emacs, GNU Debugger, GNU C Compiler, ecc. Nel loro insieme, tali strumenti rappresentano i componenti di un sistema operativo libero sviluppato, mantenuto e posseduto dalla comunità hacker mondiale. Anziché considerare quest'ultima come una minaccia, grandi società informatiche quali IBM, Hewlett-Packard e Sun Microsystems hanno deciso di appoggiarla, commercializzando applicazioni e servizi realizzati per operare appositamente all'interno dell'infrastruttura in continua crescita del software libero.

Queste aziende vi fanno affidamento anche come arma strategica nella guerra infinita intrapresa dalla comunità hacker contro la Microsoft, che nel bene o nel male ha dominato il mercato del software per PC fin dagli ultimi anni '80. In quanto proprietaria del popolare sistema operativo Windows, la Microsoft è quella che rischia di più nel caso di un passaggio alla licenza GPL in grande scala. Quasi ogni riga di codice presente nel colosso Windows è protetta da copyright, ad ulteriore conferma della natura privata dei sorgenti sottostanti oppure, in ultima analisi, per riaffermare la possibilità legale di trattarli in quanto tali. Dal punto di vista di Microsoft, l'eventualità di incorporare all'interno di tale colosso un qualche programma protetto dalla "virale" GPL potrebbe essere paragonato a Superman che inghiotte un intero flacone di pillole alla Kryptonite. Le aziende rivali potrebbero immediatamente copiare, modificare e vendere versioni migliorate di Windows, rendendo vulnerabile all'istante la propria posizione di numero uno tra i produttori di software nell'area consumer. Questo spiega le crescenti preoccupazioni della società rispetto al ritmo di adozione della GPL. Da qui il recente intervento di Mundie contro la GPL e l'approccio "open source" allo sviluppo e alla vendita di software e, di conseguenza, la decisione odierna di Stallman di replicare pubblicamente a quel discorso all'interno del medesimo campus universitario.

Per l'industria del software, vent'anni non sono poca cosa. Consideriamo questo: nel 1980, quando Richard Stallman imprecava contro la stampante

laser Xerox nel laboratorio di intelligenza artificiale, la Microsoft, azienda considerata dagli odierni hacker al primo posto nell'industria del software mondiale, non era altro che una start-up privata. La IBM, azienda fino ad allora considerata dagli hacker come la maggior potenza della stessa industria, doveva ancora commercializzare il suo primo personal computer, sulla cui scia sarebbe poi esploso un mercato dai prezzi in costante ribasso. Gran parte delle tecnologie oggi date per scontate – il World Wide Web, la televisione via satellite, le console video a 32 bit – non esistevano neppure. Lo stesso vale per molte di quelle entità oggi ai primi posti nel firmamento della grande imprenditoria statunitense, aziende quali AOL, Sun Microsystems, Amazon.com, Compaq, Dell. E l'elenco potrebbe continuare.

Il fatto che il mercato della tecnologia sia cresciuto in maniera esponenziale in così poco tempo fornisce motivazioni valide per entrambe le parti coinvolte nel dibattito sulla GPL.

I sostenitori di quest'ultima sottolineano la vita relativamente breve della maggior parte delle piattaforme hardware. Davanti al rischio di acquistare un prodotto obsoleto, i consumatori tendono a optare per quelle aziende che sembrano vivere più a lungo. Di conseguenza il mercato è divenuto una sorta di arena in cui chi vince piglia tutto[2]. L'odierno settore del software sotto proprietà privata, affermano i sostenitori della GPL, conduce all'abuso e alla stagnazione tipiche dei monopoli. Le aziende più forti succhiano tutto l'ossigeno dal mercato a danno dei diretti concorrenti e delle start-up innovative.

Chi è contro la GPL sostiene esattamente l'opposto. Vendere software è rischioso tanto quanto acquistarlo, se non di più, ribadiscono costoro. In assenza delle garanzie legali previste nelle licenze private, per non parlare delle prospettive economiche susseguenti a una "killer application" proprietaria (ovvero, una tecnologia dirompente che apre un mercato del tutto nuovo)[3], le aziende perdono incentivo a partecipare. Di nuovo, il mercato andrebbe verso la stagnazione, e l'innovazione verso il declino. Come ha fatto notare lo stesso Mundie nel suo intervento del 3 maggio 2001 nel medesimo campus, la natura "virale" della GPL "pone una minaccia" a qualsiasi azienda

che basa sull'unicità del software il proprio vantaggio competitivo. Così Mundie:

> Fondamentalmente la GPL mette in pericolo anche il settore indipendente del software commerciale poiché rende effettivamente impossibile una distribuzione in cui l'acquirente sia chiamato a pagare per il prodotto anziché soltanto per i costi di distribuzione[4].

Il reciproco successo sia di GNU/Linux, il sistema operativo costruito intorno al kernel Linux e coperto da licenza GPL, sia di Windows nel corso degli ultimi dieci anni rivela la validità di entrambi i punti di vista. Tuttavia, la battaglia appare di estrema importanza per l'intera industria del software. Perfino potenti produttori come Microsoft fanno affidamento sul sostegno di sviluppatori esterni i cui strumenti, programmi e giochi rendono più attraente per il consumatore medio la piattaforma sottostante, in questo caso Windows. Citando la rapida evoluzione del mercato tecnologico nel corso degli ultimi vent'anni, nonché l'ammirevole record di crescita della propria azienda nello stesso periodo, Mundie ha messo in guardia i presenti contro i facili entusiasmi per i recenti successi del movimento del software libero:

> Due decenni di esperienza hanno dimostrato come un modello economico capace di tutelare la proprietà intellettuale e un modello commerciale in grado di recuperare gli stanziamenti per la ricerca e lo sviluppo, siano in grado di creare notevoli benefici economici e di distribuirli in maniera assai ampia[5].

Questi gli ammonimenti che fanno da contraltare all'odierno intervento di Stallman. A meno di un mese di distanza da quel discorso, eccolo pronto a iniziare, in piedi, le spalle a una lavagna, in un'aula della New York University.

Se gli ultimi due decenni hanno portato enormi cambiamenti nel mercato del software, quelli subìti dallo stesso Stallman sono ancora più evidenti. È ormai scomparso l'hacker smilzo e senza barba che una volta trascorreva intere giornate a stretto contatto di gomito con l'amato PDP-10. Al suo posto troviamo un uomo di mezza età, ben piazzato, capelli lunghi e barba rabbinica, un uomo che oggi trascorre gran parte del tempo a scrivere e rispondere a messaggi e-mail, ad arringare i colleghi programmatori, a tenere interventi pubblici come quello odierno. Con indosso una maglietta color carta

da zucchero e un paio di pantaloni marroni di poliestere, Stallman sembra un eremita del deserto che ha appena arraffato qualche vestito in un guardaroba dell'Esercito della Salvezza.

La platea è piena di visitatori che ne condividono abbigliamento e aspetto. Molti girano con computer portatili e modem cellulari, onde poter registrare e trasmettere le parole di Stallman al trepidante pubblico di Internet. Il rapporto tra i presenti è all'incirca di 15 uomini per ogni donna, e una delle 7-8 donne porta un pinguino di pezza, mascotte ufficiale di Linux, mentre un'altra ha con sé un orsacchiotto.



Figura 2.1 Richard Stallman, circa 2000. "Decisi che avrei sviluppato un sistema operativo di software libero oppure sarei morto provandoci… di vecchiaia, naturalmente." Foto ripresa da http://www.stallman.org

Nervoso, Stallman si sposta dal centro della sala e va a sedersi su una sedia in prima fila, a digitare alcuni comandi su un portatile già aperto. Per i dieci minuti successivi egli è del tutto indifferente al crescente numero di studenti, professori e appassionati che gli girano davanti ai piedi del palco dell'auditorium.

Prima che il discorso possa avere inizio, occorre comunque osservare il rituale barocco delle formalità accademiche. La presenza di Stallman merita non una, bensì due introduzioni. A Mike Uretsky, condirettore del Center for Advanced Technology presso la Stern School, spetta la prima.

“Il ruolo di ogni università è quello di stimolare il dibattito e produrre discussioni interessanti”, inizia Uretsky. “Questa presentazione in particolare, questo seminario rientra perfettamente in tale contesto. Trovo di particolare interesse ogni contributo sul tema dell’open source”.

Prima che possa aggiungere un’altra parola, Stallman è in piedi che gesticola vivacemente come qualcuno rimasto in panne sull’autostrada.

“Io mi occupo di software libero”, urla tra un crescendo di risa. “L’open source è un movimento diverso”.

Le risate si trasformano in applausi. La stanza è zeppa di sostenitori di Stallman, gente che conosce bene la sua reputazione di pignoleria verbale, per non parlare del suo distacco, ampiamente pubblicizzato nel 1998, dai sostenitori del software open source. Molti sono soliti attendere con fervore simili uscite di Stallman, divenute una sorta di marchio di fabbrica inconfondibile.

Uretsky chiude rapidamente l’introduzione per passare la parola a Edmond Schonberg, docente del dipartimento d’informatica della New York University. Programmatore e collaboratore del progetto GNU, Schonberg conosce le trappole linguistiche da evitare. Egli riassume per sommi capi la carriera di Stallman secondo il punto di vista di un programmatore dei nostri giorni.

“Richard incarna il perfetto esempio di qualcuno che, operando a livello locale, ha iniziato a pensare globalmente [ai] problemi relativi alla mancata disponibilità del codice sorgente”, dice Schonberg. “Ha così sviluppato una filosofia coerente che ha costretto tutti noi a riesaminare le nostre concezioni sulla produzione del software, sul significato della proprietà intellettuale e sul valore concreto della comunità dei programmatori.”

Schonberg dà poi il benvenuto a Stallman in un crescendo di applausi. In pochi istanti quest’ultimo spegne il portatile, si alza dalla sedia e sale sul palco.

Inizialmente il discorso di Stallman richiama più un repertorio teatrale che un intervento politico. “Vorrei ringraziare la Microsoft per avermi offerto l’opportunità di trovarmi in questa sede”, così rompe il ghiaccio. “Nelle

ultime settimane mi sono sentito come uno scrittore i cui libri siano stati accidentalmente vietati in qualche paese."

Per i non iniziati, Stallman passa poi ad illustrare rapidamente un'analogia per il software libero. Egli è solito paragonare un programma a una ricetta di cucina. Entrambi forniscono istruzioni dettagliate su come portare a compimento un certo compito e possono essere facilmente modificati a seconda di esigenze o circostanze individuali. "Non occorre seguire alla lettera una ricetta", fa notare Stallman. "Si può omettere qualche ingrediente. Aggiungere un po' di funghi perché ci piacciono. Mettere meno sale perché così ci ha consigliato il medico – e via di seguito."

Elemento ancor più rilevante, prosegue Stallman, è la semplicità nel condividere sia ricette che programmi. Nel passare la ricetta a un ospite, il cuoco non ci rimette altro che il tempo e il costo della carta necessari per scrivere la ricetta. Per il software occorre ancora meno, in genere basta qualche clic del mouse e un minimo di elettricità. In entrambi i casi, tuttavia, la persona che trasmette quelle informazioni guadagna due cose: una maggiore amicizia e la possibilità di ottenere in cambio altre ricette interessanti.

"Immaginiamo cosa accadrebbe se le ricette fossero rinchiuse all'interno di scatole nere", prosegue Stallman, cambiando registro. "Sarebbe impossibile vedere gli ingredienti usati, per non parlare della possibilità di modificarli; e chissà quali conseguenze nel farne una copia per un amico. Saremmo bollati come pirati, processati e sbattuti in galera per anni. Uno scenario che susciterebbe accese proteste da parte di quanti amano scambiarsi ricette. Ma questo è esattamente lo scenario dell'odierno software proprietario. Un mondo in cui si proibiscono o impediscono azioni di comune buon senso verso gli altri."

Dopo quest'analogia introduttiva, Stallman si lancia nel racconto dell'episodio della stampante laser Xerox. Come nel caso delle ricette, anche questa vicenda è un utile espediente retorico. Il racconto in forma di parabola evidenzia la rapidità con cui le cose possono cambiare nel mondo del software. Riportando i presenti al periodo precedente all'acquisto con un solo

clic del mouse ideato da Amazon.com, all'esistenza di Windows o dei database di Oracle, invita chi ascolta a esaminare la nozione di proprietà del software, indipendentemente dall'esistenza degli attuali marchi delle grandi corporation.

Stallman presenta il racconto con tutta l'arguzia e la pratica di un pubblico ministero alle prese con l'arringa finale. Quando arriva alla scena del professore alla Carnegie Mellon che rifiuta di cedergli una copia dei sorgenti della stampante, Stallman fa una pausa.

"Ci ha traditi", prosegue. "Non soltanto ha tradito tutti noi, ma anche te."

Pronunciando quel "te" punta l'indice accusatorio contro un insospettabile membro del pubblico. Costui aggrotta le sopracciglia, ma lo sguardo di Stallman è andato già oltre. In maniera lenta e deliberata, sceglie qualcun altro in platea che sta per accennare una risatina nervosa. "E molto probabilmente ha fatto lo stesso con te", insiste indicando stavolta una persona seduta tre file dietro la prima.

Quando Stallman ripete la scena per la terza volta, le risatine si sono trasformate in una risata generale. Il tutto appare un tantino teatrale, così è infatti. E al momento di concludere la storia della stampante, Stallman si esibisce come un attore consumato. "Probabilmente ha tradito la maggior parte delle persone riunite in questa sala – eccetto forse alcuni che nel 1980 non erano ancora nati", conclude Stallman, suscitando risate ulteriori. "[Questo] perché aveva promesso di rifiutare ogni collaborazione praticamente con l'intera popolazione del pianeta Terra."

E dopo una pausa a effetto per lasciar sedimentare i commenti, aggiunge: "Aveva firmato un contratto di non divulgazione."

Il passaggio di Richard Matthew Stallman da accademico frustrato a leader politico nel corso degli ultimi vent'anni è testimonianza di diversi elementi importanti: la sua natura testarda e la volontà prodigiosa, una visione ben articolata dei valori di quel movimento per il software libero che ha aiutato a costruire. Testimonia inoltre dell'alta qualità dei programmi da lui realizzati, programmi che ne hanno consolidato la reputazione come una leggenda nell'ambito della programmazione. E rivela altresì il crescente successo della

GPL, un’innovazione legale che molti osservatori considerano il suo contributo più valido.

Fatto sicuramente più importante, tale passaggio mette in luce il mutamento del potere politico in un mondo sempre più dipendente dalla tecnologia informatica e dal software su cui questa poggia.

Forse è questo il motivo per cui, anche di fronte al declino di buona parte delle stelle high-tech, la fama di Stallman sia andata invece crescendo. Fin dal lancio del progetto GNU nel 1984[6], egli è stato di volta in volta ignorato, ridicolizzato, denigrato e attaccato – sia all’interno sia all’esterno del movimento del software libero. In tutto ciò, quel progetto è riuscito a raggiungere successi importanti, pur se con i notori ritardi, acquistando un ruolo rilevante in un mercato infinitamente più complesso di quello in cui egli era entrato 18 anni fa. Analoga l’evoluzione seguita dall’ideologia del software libero, un’ideologia meticolosamente coltivata dallo stesso Stallman.

Per comprendere le motivazioni che sottendono all’attuale scenario, potrà giovare un’analisi del personaggio sia tramite le sue parole sia tramite quelle di chi ha collaborato e combattuto al suo fianco fino a oggi. Non sembra certo complicato tracciarne un ritratto. Se c’è una persona che può incarnare il vecchio adagio, “quel che vedi è quello che ottieni[7]”, questi è Stallman.

“Credo che per comprendere Richard Stallman in quanto essere umano, bisogna considerarne le varie componenti come un insieme coerente”, mette sull’avviso Eben Moglen, consigliere legale per la Free Software Foundation e professore di legge presso la Columbia University Law School. “Tutte quelle espressioni di eccentricità personale che molta gente percepisce come ostacoli alla conoscenza di Stallman, in realtà ne costituiscono l’essenza: il suo forte senso di frustrazione, l’enorme attaccamento ai principi etici, l’incapacità di scendere a compromessi, soprattutto su questioni che considera fondamentali. Ecco le ragioni concrete per cui Richard ha fatto ciò che ha fatto quando lo ha fatto.”

È tutt’altro che semplice spiegare come mai un viaggio iniziato con una stampante laser sia poi sfociato in uno scontro aperto con le più ricche corporation del mondo. Una spiegazione che richiede un’attenta disamina di

quelle forze le cui spinte hanno reso la proprietà del software un elemento così importante nella società odierna. Impone altresì un'accurata analisi dell'uomo che, al pari di altri leader politici a lui precedenti, sa far propria la malleabilità della memoria umana. Esige inoltre la capacità di interpretare i miti e la terminologia, con le relative valenze politiche, costruite intorno a Stallman con l'andar del tempo. Richiede infine la comprensione della sua genialità in quanto programmatore, di pari passo con fallimenti e successi nel trasferire tale genialità in altri contesti.

Chiamato ad offrire una sintesi soggettiva del proprio cammino, Stallman conferma quella fusione tra personalità e principi già sottolineata da Moglen. "La mia prima caratteristica è la testardaggine", conferma. "La maggior parte di quanti s'imbarcano in progetti di grande difficoltà finiscono per perdere coraggio e abbandonare. Io sono uno che non molla mai."

Né manca di ammettere l'importanza della pura coincidenza. Se non fosse stato per quell'incontro con la stampante Xerox, se non fosse stato per i conflitti politici e personali che gli chiusero carriera alle dipendenze del MIT, se non fosse stato per una mezza dozzina di altri fattori capitati al momento giusto, Stallman riesce facilmente ad immaginarsi lungo un percorso professionale assai diverso. Ciò detto, però, ringrazia le forze e le circostanze che lo hanno portato a ricoprire una posizione in cui può davvero cambiare le cose.

"È capitato che avessi le capacità adatte", insiste Stallman, riassumendo per i presenti la decisione di lanciare il progetto GNU. "Non c'era nessuno eccetto il sottoscritto, allora mi son detto, 'Sono un eletto. Devo lavorare su questo fronte. Se non io, chi altri?"'

[1] In realtà i punti di forza della GPL non sembrano in fondo così potenti. Nella sezione 10 della GNU General Public License, Versione 2 (1991), la natura virale della licenza dipende parecchio dalla volontà della Free Software Foundation di considerare un programma come opera derivata, per non parlare della licenza precedente che la GPL andrebbe a sostituire. Se si vogliono incorporare parti di questo programma in altri programmi liberi con differenti condizioni di distribuzione, occorre chiederne il permesso all'autore. Per quanto concerne software tutelato dal copyright della Free Software Foundation, va contattata quest'ultima; talvolta facciamo delle eccezioni a quanto sopra. La nostra decisione si basa

sui due obiettivi di fondo: preservare la condizione libera di tutti i programmi derivati dal nostro software libero, e promuovere la condivisione e il riutilizzo del software in generale. “Qualsiasi paragone con un virus appare eccessivo”, sostiene Stallman. “Analogia più accurata è quella di una pianta in grado di propagarsi altrove, ma soltanto se qualcuno ne taglia un pezzo e lo trapianta.” Per maggiori informazioni sulla GNU General Public License, si veda: http://www.gnu.org/copyleft/gpl.html.

[2] Si veda Shubha Ghosh, “Revealing the Microsoft Windows Source Code”, *Gigalaw.com* (gennaio 2000). http://www.gigalaw.com/articles/2000-all/ghosh-2000-01-all.html

[3] Le “killer application” non devono essere per forza proprietarie. Basti ricordare, naturalmente, il leggendario browser Mosaic, programma il cui copyright permette derivati noncommerciali con alcune restrizioni. Credo tuttavia che il lettore abbia ben compreso la questione: il mercato del software è come la lotteria. Maggiori i premi in palio, più numerose le persone che vi partecipano. Per un buon riassunto sul fenomeno delle “killer application” si veda Philip Ben-David, “Whatever Happened to the ‘Killer App’?” e-Commerce News (7 dicembre 2000). http://www.ecommercetimes.com/perl/story/5893.html

[4] Si veda Craig Mundie, senior vice president della Microsoft, “The Commercial Software Model”. Stralci ripresi dalla trascrizione online dell’intervento di Mundie tenuto il 3 maggio 2001 alla Stern School of Business della New York University. http://www.microsoft.com-/presspass/exec/craig/05-03sharedsource.asp

[5] http://www.microsoft.com/presspass/exec/craig/05-03sharedsource.asp

[6] L’acronimo GNU sta per “GNU’s not Unix.” In una parte successiva del discorso del 29 maggio 2001 alla New York University, così Stallman riassumeva l’origine dell’acronimo: “A noi hacker piace chiamare i programmi con un nome scherzoso o malizioso, perché trovare il nome adatto è metà del divertimento di scriverlo. Siamo anche soliti seguire la tradizione degli acronimi ricorsivi, per indicare che il programma su cui si lavora è simile ad altri in circolazione… Mentre cercavo un acronimo per Something Is Not UNIX, dopo aver provato tutte le 26 lettere mi resi conto che nessuna funzionava come una parola. Decisi allora di ricorrere a una contrazione, così da ottenere un acronimo di tre lettere. Prova e riprova, venne fuori il termine “GNU”. Pur se amante dei giochi parole, Stallman raccomanda di pronunciare la “g” iniziale dura (“gh-nu”). Ciò per evitare confusione con il termine “gnu (“g” dolce) che indica l’antilope africana, Connochaetes gnou. In inglese ciò evita problemi anche con l’aggettivo “new” (si legge: “niu”), nuovo. Spiega ancora Stallman: “Ci stiamo lavorando su da oltre 17 anni, quindi non si può dire sia proprio qualcosa di nuovo.” Fonte: note dell’autore e trascrizione online di “Free Software: Freedom and Cooperation”, intervento di Richard Stallman alla New York University, 29 maggio 2001. http://www.gnu.org/events/rms-nyu-2001-transcript.txt

[7] In inglese “what you see is what you get” (WYSIWYG) è l’espressione comunemente utilizzata per indicare quei software la cui interfaccia presenta le informazioni in forma realistica, come sarà poi l’output finale [*N.d.T.*]

Un ritratto dell’hacker da giovane

Alice Lippman, madre di Richard Stallman, ricorda ancora il momento in cui si accorse del dono speciale di cui era dotato suo figlio.

“Credo fosse quando aveva otto anni”, rammenta.

Correva l’anno 1961, e dopo il recente divorzio Lippman trascorreva in famiglia un pomeriggio festivo nel minuscolo appartamento situato nell’Upper West Side di Manhattan. Sfogliando una copia di Scientific American, si fermò alla rubrica preferita, quella di Martin Gardner intitolata “Giochi matematici”. Come supplente nei licei artistici, quella pagina le era sempre piaciuta per lo sforzo mentale che richiedevano i giochi proposti. Mentre suo figlio era immerso nella lettura di un libro sul vicino divano, decise di provare a risolvere il quesito della settimana.

“Non ero una cima nel risolvere quei problemi”, ammette. “Ma come artista ho sempre notato che mi aiutavano a superare certe barriere concettuali.”

Alice ricorda che quella volta il tentativo andò ad infrangersi contro un solido muro. Stava per abbandonare con disgusto la rivista, quando si sentì tirare la manica della maglietta.

“Era Richard”, racconta, “mi chiedeva se non avessi bisogno di una mano.”

Osservando prima il quesito e poi suo figlio, inizialmente considerò l’offerta con un certo scetticismo. “Gli chiesi se avesse già letto il giornale”, prosegue. “Lui rispose affermativamente, e anzi aveva già risolto il quesito matematico. E subito prese a spiegarmi in dettaglio la soluzione.”

Mentre seguiva il ragionamento logico del figlio, lo scetticismo lasciò rapidamente posto all’incredulità. “L’avevo sempre considerato ragazzo brillante”, aggiunge, “ma questa era la prima volta che mi trovavo davanti alla conferma concreta delle sue notevoli capacità”.

Trent’anni dopo quell’evento, la madre sottolinea il ricordo con un sorriso. “A dire il vero, non credo di aver mai capito come risolvere quel problema”, spiega. “Tutto ciò che rammento è la mia enorme sorpresa per il fatto che ne conoscesse la risposta.”

Seduta al tavolo del salotto in un altro appartamento a Manhattan – più spazioso, tre camere, lo stesso in cui traslocò con il figlio nel 1967 dopo essersi risposata con Maurice Lippman, ora deceduto – Alice Lippman lascia trasparire un misto di orgoglio e sorpresa, tipico di una madre ebrea, quando ricorda i primi anni di vita del figlio. Sulla vicina credenza spicca una grossa fotografia dove Stallman appare con la barba folta e la toga nera della laurea. Di fianco ecco altre immagini più piccole con i nipoti della Lippman, ma per evitare di dare troppa importanza a quella foto, si assicura di bilanciarne la posizione prominente con un commento ironico.

“Richard insistette perché la tenessi dopo il suo dottorato onorario all’Università di Glasgow”, dice. “Mi fece, ‘Indovina un po’, mamma? È la prima cerimonia di laurea a cui abbia mai preso parte.”’[1]

Simili commenti riflettono il senso dell’umorismo che bisogna avere quando si cresce un bambino prodigio. A scanso di equivoci, però, per ogni storia che sente e legge sulla testardaggine e sui comportamenti insoliti del figlio, può offrirne almeno una dozzina altrettanto significative.

“Il più delle volte aveva un atteggiamento così conservatore”, sostiene, gesticolando per un’ironica esasperazione. “Eravamo soliti tenere le peggiori discussioni proprio qui, a questo tavolo. Io facevo parte del primo gruppo di insegnanti di scuole pubbliche che aveva deciso di aderire al sindacato, e per questo Richard ce l’aveva molto con me. Considerava il sindacato sinonimo di corruzione. Era anche contrario alla pensione statale. Secondo lui, la gente poteva accumulare molto più denaro facendo investimenti in proprio. Chi poteva prevedere che in dieci anni sarebbe diventato talmente idealista? Tutto quello che rammento è che un giorno sua sorella venne da me e fece, ‘Cosa diventerà mai da grande? Un gran fascista?”’

Come unico genitore per circa un decennio – lei e il padre di Richard, Daniel Stallman, si sposarono nel 1948 per divorziare nel 1958, condividendo

poi la custodia del figlio – la madre convalida l'avversione del figlio per l'autorità costituita. Ne conferma altresì la brama di conoscenza. Era quando queste due forze s'intrecciavano, insiste, che lei e il figlio si davano alle battaglie più accese.

"Sembrava non volesse mai mangiare", afferma rammentando quei modelli comportamentali stabilitisi verso gli otto anni e rimasti inalterati fino al diploma di scuola superiore nel 1970. "Gli dicevo che la cena era pronta, e non mi ascoltava. Dovevo chiamarlo nove o dieci volte soltanto per avere la sua attenzione. Era totalmente assorto in qualcosa."

Da parte sua Stallman ricorda situazioni analoghe, ma con un pizzico di politica in più.

"Amavo leggere", dice. "Se avevo voglia di leggere e mia madre mi diceva di venire a tavola o di andare a letto, non mi curavo di darle ascolto. Non vedevo motivo perché non potessi continuare a leggere. Non c'era nessuna ragione per cui lei potesse dirmi cosa fare, punto e basta. In pratica, quello che avevo letto su robe tipo democrazia e libertà individuale, lo applicavo a me stesso. Non mi pareva giusto escludere i bambini da simili principi."

L'assoluta convinzione nella libertà individuale contro ogni autoritarismo tendeva ad estendersi anche all'ambiente scolastico. All'età di 11 anni era due anni avanti ai compagni di classe, costretto a sopportare le solite frustrazioni di uno studente assai dotato. Non era passato molto tempo dalla scena del quesito matematico, che sua madre venne chiamata per il primo di una lunga serie di incontri con gli insegnanti di Richard.

"Si rifiutava categoricamente di scrivere temi o ricerche", dice Lippman a proposito di una questione assai controversa. "Credo che l'ultima cosa che scrisse prima di passare alle medie fosse un riassunto sulla storia del sistema numerico in occidente, per l'insegnante di quarta elementare."

Dotato di talento in qualunque ambito richiedesse un approccio analitico, Stallman era solito gravitare verso matematica e scienze a spese di altre materie. Quel che ad alcuni insegnanti appariva come una sorta di scarsa elasticità, tuttavia, per la madre non era altro che impazienza. Scienza e matematica offrivano semplicemente troppe opportunità per l'apprendimento,

soprattutto in confronto a materie e attività verso le quali il figlio appariva meno predisposto. Verso i 10 o 11 anni, quando i compagni di classe di Stallman partecipavano regolarmente a partite di calcio, lei lo ricorda tornare a casa inviperito. “Aveva una gran voglia di giocare, ma non aveva molta coordinazione”, spiega. “Questo lo faceva andare su tutte le furie.”

Alla fine la rabbia spinse Richard a dedicarsi con ulteriore passione a scienza e matematica. Tuttavia, anche in campo scientifico l’impazienza poteva giocargli brutti scherzi. Immerso nei libri di calcolo all’età di sette anni, Stallman non vedeva alcun bisogno di esprimersi in maniera più semplice per gli adulti. Una volta, erano gli anni delle medie, Lippman assunse uno studente della vicina Columbia University per giocare al fratello maggiore con il figlio. Dopo la prima volta, lo studente lasciò l’appartamento per non farvi mai più ritorno. “Credo che il modo di esprimersi di Richard gli fece girare la testa”, specula oggi la Lippman.

Un altro aneddoto materno risale all’inizio degli anni ’60, poco dopo il caso del gioco matematico. A circa sette anni, due anni dopo il divorzio e il trasloco dal Queens, Richard si diede all’hobby di lanciare modellini di razzi nel vicino Riverside Drive Park. Quello che era iniziato come un innocuo passatempo prese una piega molto seria quando iniziò a tener nota dei risultati di ogni lancio. Come per l’interesse nei problemi matematici, la faccenda attirò poca attenzione fino al giorno in cui, poco prima di un importante lancio spaziale della NASA, Lippman chiese al figlio se non volesse guardarlo in televisione.

“Era furioso”, afferma Lippman. “Per tutta risposta riuscì a dirmi: ‘Ma non ho ancora pubblicato nulla.’ Sembrava proprio che avesse qualcosa di serio da mostrare alla NASA.”

Tali aneddoti costituiscono le prove precoci di quell’intensità che sarebbe divenuta la principale caratteristica di Stallman per tutta la vita. Quando altri bambini si ritrovavano intorno al tavolo, Stallman se ne stava in camera sua a leggere. Quando altri bambini ascoltavano la musica di Johnny Unitas, egli preferiva Werner von Braun. “Ero strano”, sostiene Stallman riassumendo

succintamente i suoi primi anni di vita nel corso di un'intervista datata 1999. "Dopo una certa età, gli unici amici che avevo erano gli insegnanti."[2]

Nonostante ciò comportasse ulteriori impegni scolastici, Lippman decise di assecondare la passione del figlio. All'età di 12 anni, in estate Richard prese a frequentare dei corsi scientifici integrando l'anno scolastico con lezioni private. Quando un insegnante le raccomandò di iscriverlo al Columbia Science Honors Program, un corso sull'era del dopo-Sputnik realizzato per i migliori studenti di scuola media di New York City, Stallman lo aggiunse alle sue attività extra-scolastiche e si trovò presto a fare il pendolare recandosi in centro ogni sabato, al campus della Columbia University.

Dan Chess, suo compagno di classe durante quel corso, ricorda che Stallman appariva un po' strano perfino agli studenti che condividevano la stessa grande passione per la scienza e la matematica. "Eravamo un po' tutti fissati e asociali, ma lui sembrava insolitamente impacciato", ricorda Chess, oggi professore di matematica presso l'Hunter College. "Era anche incredibilmente intelligente. Ho conosciuto parecchie persone in gamba, ma credo che lui fosse il più sveglio di tutti."

Seth Breidbart, anch'egli ex-studente del Columbia Science Honors Program, rincara la dose. Programmatore informatico rimasto in contatto con Stallman grazie alla comune passione per la fantascienza e i relativi eventi pubblici, ricorda che il quindicenne Stallman "incuteva timore", specialmente ad un coetaneo.

"Non è facile descriverlo", aggiunge Breidbart. "Non che fosse inavvicinabile. Solo che era molto sensibile. [Era] molto perspicace ma anche molto testardo alcune volte."

Simili descrizioni si prestano ad alcune speculazioni: aggettivi e giudizi quali "sensibile" e "testardo" vanno forse intesi come modalità descrittive di quei tratti caratteriali oggi pertinenti alla categoria dei disturbi comportamentali dell'adolescenza? Un articolo apparso nel dicembre 2001 sul mensile *Wired* con il titolo "The Geek Syndrome" presenta la descrizione di diversi bambini portati per la scienza, e affetti da autismo ad alta funzionalità o sindrome di Asperger. Per molti versi, i ricordi dei genitori riportati

nell’articolo di *Wired* assomigliano stranamente a quelli raccontati da Alice Lippman. Perfino Stallman di tanto in tanto si è concesso qualche analisi psichiatrica. Nel 2000, durante un’intervista per il *Toronto Star*, Stallman si è autodefinito “sull’orlo dell’autismo”[3] descrizione che la dice lunga sulle ragioni della sua continua tendenza verso l’isolamento sociale ed emotivo, e sullo sforzo parimenti intenso di superare tale tendenza.

Naturalmente queste speculazioni sono facilitate dal tira e molla che oggi si opera sulla maggior parte dei cosiddetti “disturbi comportamentali”. Come fa notare Steve Silberman, autore di “The Geek Syndrome”, soltanto recentemente gli psichiatri americani sono arrivati a considerare la sindrome di Asperger come termine valido per coprire un’ampia serie di tratti comportamentali. Questi vanno da scarse capacità motorie a difficoltà di socializzazione, da un’intelligenza assai vivace a un’affinità quasi ossessiva con numeri, computer e sistemi ordinati[4]. Riflettendo sulla natura variegata di tale classificazione, Stallman ritiene che, se fosse nato 40 anni più tardi, sarebbe probabilmente rientrato in quella diagnosi. Ma, ancora una volta, lo stesso può dirsi per molti suoi colleghi nel mondo informatico.

“È possibile che io sia stato affetto da qualcosa di simile”, dice. “D’altra parte, uno degli aspetti di tale sindrome riguarda la difficoltà a seguire il ritmo. Io so ballare, anzi, adoro seguire i ritmi più complicati. Quindi non è così semplice.”

Da parte sua Chess rifiuta questi tentativi di diagnosi all’indietro. “Non ho mai pensato potesse avere qualcosa di simle”, dice. “Era soltanto poco socievole, ma a quel tempo tutti lo eravamo.”

La Lippman, d’altra parte, lascia aperta questa possibilità. Rammenta comunque un paio di episodi nell’infanzia del figlio che danno adito a ulteriori speculazioni. Uno dei maggiori sintomi dell’autismo riguarda l’ipersensibilità a rumori e colori, e la madre ricorda due aneddoti che sembrano confermarlo. “Quando Richard era piccolo, lo portavamo in spiaggia”, dice. “Cominciava a urlare ancor prima di arrivare. Alla terza volta capimmo qual era il problema: il rumore della risacca era fastidioso per le sue orecchie.” Analoga reazione di

fronte ai colori: “Mia madre aveva i capelli d’un rosso brillante, e ogni volta che faceva per prenderlo in braccio, lui cominciava a lamentarsi.”

Negli ultimi anni la Lippman dice di aver letto libri sull’autismo e ritiene tali episodi qualcosa di più che semplici coincidenze. “Credo che Richard avesse alcune qualità dei bambini autistici”, afferma. “Rimpiango solo il fatto che allora si sapesse così poco dell’autismo.”

Col passar del tempo, tuttavia, suo figlio imparò a controllarsi. A sette anni lo ricorda vantarsi di stare in piedi sulla prima carrozza della metropolitana, memorizzando la mappa dei labirinti che costituivano il sistema delle rotaie sotto la città. Questo passatempo si basava sulla capacità di ignorare il forte rumore tipico di ogni treno. “Pareva essere infastidito soltanto dal frastuono iniziale”, spiega. “Era come se subisse un shock in fase d’avvio del rumore, ma poi i suoi nervi avevano imparato ad adeguarvisi.”

Per la maggior parte, Lippman ricorda che il figlio esprimeva l’eccitamento, l’energia, e le tendenze sociali di qualunque ragazzo normale. Fu solo quando la famiglia Stallman precipitò in una serie di eventi traumatici che divenne introverso ed emotivamente distante.

Il primo di tali eventi fu il divorzio tra Alice e Daniel Stallman, il padre di Richard. Nonostante la madre sostenga che entrambi cercarono di preparare al meglio il ragazzo, il colpo fu comunque devastante. “Quando gli spiegammo cosa stava accadendo, sulle prime non prestò alcuna attenzione”, rammenta Lippman. “Ma la realtà fu come uno schiaffo quando io e lui ci trasferimmo nel nuovo appartamento. La prima cosa che disse fu, ‘Dove sono i mobili di papà?”’

Per il decennio successivo, Stallman avrebbe trascorso la settimana nella casa della madre a Manhattan e i week-end in quella del padre nel Queens. Gli spostamenti avanti e indietro gli diedero la possibilità di confrontare le contrastanti differenze dei genitori al punto che a tutt’oggi rimane decisamente contrario all’idea di avere figli propri. Parlando di suo padre, un veterano della Seconda Guerra Mondiale deceduto nella primavera 2001, Stallman lo fa mescolando rispetto e rabbia. Da una parte c’è l’integrità morale dell’uomo che volle imparare il francese soltanto per poter essere di

maggiore aiuto agli Alleati dopo il loro sbarco. Dall'altra, c'era invece il genitore che sapeva sempre come fare per demoralizzarti in maniera crudele[5].

"Mio padre aveva un carattere terribile", dice Stallman. "Non urlava mai, ma trovava sempre l'occasione per criticarti in modo gelido, apposta per buttarti giù."

Riguardo la vita in casa della madre, Stallman è meno equivoco. "Lì c'era guerra aperta", sostiene. "Nei momenti peggiori, ero solito dirmi, 'Voglio andare a casa,' riferendomi a quel luogo inesistente che non avrò mai."'

Nei primi anni del divorzio, Stallman trovò quella tranquillità che gli mancava nell'abitazione dei nonni paterni. Poi, quando aveva dieci anni, morirono a breve distanza l'uno dall'altra. Per Stallman si trattò di una perdita devastante. "Quando andavo a far loro visita, sentivo di trovarmi un ambiente amorevole, gentile", ricorda. "Era l'unico posto in cui ciò mi accadeva, fino a quando partii per il college."

Lippman elenca la dipartita dei nonni paterni come il secondo evento traumatico. "Ne rimase molto colpito", dice. "Si sentiva molto vicino a entrambi. Prima che morissero, era molto gioviale, quasi il tipo da leader del gruppo quando stava con gli altri ragazzi. Dopo la loro scomparsa, divenne emotivamente molto più introverso."

Dal punto di vista di Stallman, questa introversione non era altro che un modo per affrontare l'agonia dell'adolescenza. Descrivendo quegli anni come "orrore puro", sostiene di essersi sentito non di rado come un sordo in mezzo a una folla che chiacchiera ascoltando musica.

"Spesso avevo la sensazione di non essere in grado di capire quel che gli altri andavano dicendo", spiega Stallman, rammentando la bolla emotiva che lo isolava dal resto del mondo adolescenziale e adulto. "Ne comprendevo le parole, ma c'era qualcosa nelle conversazioni che non riuscivo ad afferrare. Non capivo perché mai la gente fosse interessata a quello che dicevano gli altri."

Pur in tutta quell'agonia, l'adolescenza avrebbe infine prodotto un effetto incoraggiante sul suo senso d'individualità. In un periodo in cui la maggior parte dei compagni di classe si lasciavano crescere i capelli, Stallman

preferiva tenerli corti. In un epoca in cui tutto il mondo giovanile ascoltava il *rock & roll*, egli optava per la musica classica. Convinto appassionato di fantascienza, della rivista *Mad* e dei programmi notturni in televisione, Stallman andava coltivando una personalità decisamente anticonformista che gli valse l'incomprensione di genitori e amici.

"Oh, tutte quelle battute", dice la Lippman, ancora esasperata ricordando il carattere del figlio adolescente. "Non c'era una cosa che potevi dire a cena senza stimolarne una frecciatina."

Fuori casa, Stallman riservava le battute più sagaci a quegli adulti che parevano appagare le sue qualità innate. Una delle prime volte toccò all'assistente di un corso estivo che passò a Stallman, allora dodicenne, la stampa del manuale per il computer IBM 7094. Per un ragazzo affascinato dai numeri e dalla scienza, si trattava di un dono venuto dal cielo[6]. Stallman si mise a scrivere programmi su carta seguendo le specifiche interne del modello 7094, nell'impaziente attesa di avere la possibilità di sperimentarli su una macchina vera.

Il primo personal computer era di là da venire per ancora un decennio, per cui sarebbe stato costretto ad attendere ancora qualche anno prima di poter mettere le mani su un computer. Finalmente ecco presentarsi la prima vera occasione, durante il primo anno delle medie. Lavorando all'IBM New York Scientific Center, reparto ora scomparso nel cuore di Manhattan, Stallman trascorse l'estate scrivendo il suo primo programma in linguaggio PL/I, un pre-processore per il 7094. "Iniziai con il PL/I, poi rifeci tutto da capo in assembler quando il programma risultò troppo grosso per quel computer", rammenta Richard.

Dopo aver lavorato all'IBM Scientific Center, Stallman divenne assistente di laboratorio presso il dipartimento di biologia della Rockefeller University. Pur se già avviato verso una brillante carriera nel campo della matematica o della fisica, la mente analitica di Stallman colpì a tal punto il direttore del laboratorio che qualche anno dopo aver lasciato il collegio, la madre ricevette una telefonata del tutto inattesa. "Era il professore della Rockefeller", ricorda. "Voleva sapere come se la passava Richard. Rimase sorpreso nell'apprendere

che lavorava con i computer. Aveva sempre creduto che potesse avere un grande futuro come biologo."

Le capacità analitiche di Stallman avevano impressionato anche i docenti della Columbia, anche se spesso non mancava di farli infuriare. "In genere una o due volte ogni lezione [Stallman] trovava qualche errore", sostiene Breidbart. "Ed era tutt'altro che timido nel farlo immediatamente presente al professore di turno. Fu così che si guadagnò parecchio rispetto ma poca popolarità."

L'aneddoto di Breidbart suscita un sorriso ironico su volto di Stallman. "È vero, talvolta esageravo un po"', ammette. "Ma tra gli insegnanti ho trovato persone aperte, perché anche a loro piaceva imparare. Non così, in genere, tra gli studenti. Almeno, non nella stessa maniera."

Comunque sia, il fatto di trascorrere il sabato con ragazzi più grandi spinse Stallman a considerare meglio i meriti di una maggiore socializzazione. Con l'approssimarsi del college, al pari di molti studenti del Columbia Science Honors Program aveva ristretto la lista delle possibili scelte a due nomi: Harvard e il MIT. Vista l'opzione limitata a istituti di livello così alto, la Lippman aveva iniziato a preoccuparsi. Come quindicenne di scuola media, Stallman era ancora solito esprimere disaccordo con insegnanti e amministratori. Appena l'anno prima aveva avuto ottimi voti in storia americana, chimica, francese e algebra, ma un voto del tutto insoddisfacente in inglese che rispecchiava il continuo boicottaggio dei compiti scritti. Simili inadempienze potevano passare inosservate al MIT, ma non certo ad Harvard.

Durante l'ultimo anno delle medie, la madre decise di consultare un terapista. Quest'ultimo rimase subito colpito dal rifiuto di Stallman per i compiti scritti e dai frequenti confronti con gli insegnanti. Il figlio era sicuramente dotato dei requisiti intellettuali per eccellere ad Harvard, ma aveva forse la pazienza di seguire quei corsi che alla fine impongono una relazione scritta? Il terapista suggerì una sorta di test. Se Stallman fosse riuscito a frequentare con successo un anno presso una scuola pubblica di New York City, incluso l'esame di una materia come inglese che richiede un compito scritto, probabilmente ce l'avrebbe fatta anche ad Harvard. Finite le

medie, eccolo iscriversi immediatamente ai corsi estivi della Louis D. Brandeis High School, istituto pubblico situato sulla 84-esima, dove iniziò a recuperare in quelle materie artistiche obbligatorie che aveva trascurato alle medie.

In autunno Stallman si era allineato con il resto della popolazione scolastica delle medie a New York City. Non era certo facile starsene seduto a seguire lezioni che parevano raffazzonate rispetto ai corsi del sabato presso la Columbia, ma Lippman ricorda con orgoglio l'impegno del figlio nel voler superare quel traguardo.

"In un certo senso fu costretto a piegarsi, ma riuscì a farcela", spiega Lippman. "Mi chiamarono una sola volta, una specie di miracolo. Fu l'insegnante di calcolo a lamentarsi del fatto che Richard era solito interrompere la lezione. Gli domandai in che modo lo facesse. Rispose che Richard lo accusava di presentare prove fasulle. Gli chiesi, 'Bè, è vero?', e lui fece: 'Sì, ma non posso certo dirlo alla classe. Non capirebbero."'

Al termine del primo semestre alla Brandeis, le cose stavano prendendo la giusta piega. Pienamente recuperato l'inglese, Stallman ebbe ottimi voti in storia americana, calcolo avanzato e microbiologia. Il massimo lo raggiunse in fisica. Pur rimanendo un'asociale, Stallman concluse i suoi 11 mesi alla Brandeis al quarto posto su un totale di 789 studenti. Fuori da quelle aule, Stallman proseguì gli studi con diligenza perfino maggiore, correndo per completare i suoi doveri di assistente di laboratorio alla Rockefeller University durante la settimana e aggregandosi ai cortei contro la guerra in Vietnam lungo la strada verso i corsi del sabato alla Columbia. Fu allora che, mentre gli altri studenti dello Science Honors Program confrontavano le proprie scelte per il college, finalmente Stallman trovò un momento per partecipare a quelle discussioni.

Ricorda Breidbart, "Naturalmente la maggior parte di loro sarebbe andata ad Harvard e al MIT, giusto qualcuno pensava ad altri istituti importanti. Pur nel pieno della conversazione, Richard rimaneva l'unico a non aver detto ancora nulla. Allora qualcuno, non ricordo chi, trovò il coraggio di chiedergli direttamente cosa pensasse di fare."

Son trascorsi trent’anni, ma Breidbart ricorda chiaramente quel momento. Non appena anch’egli rivelò che in autunno si sarebbe iscritto alla Harvard University, un pesante silenzio piombò nella stanza. Al momento opportuno, gli angoli della bocca di Stallman lentamente si mossero verso un sorriso autocompiaciuto.

Spiega Breidbart, “Era il suo modo di dire, in silenzio, ‘Già, non vi siete ancora liberati di me.”’

[1] Si veda Michael Gross, “Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius” (1999). Si tratta di una delle interviste più sincere tuttora disponibili. Vivamente consigliata. http://www.mgross.com/interviews/stallman1.html

[2] http://www.mgross.com/interviews/stallman1.html

[3] Si veda Judy Steed, Toronto Star, BUSINESS, (9 ottobre 2000). La sua visione del software libero e della cooperazione sociale si pone in netto contrasto con l’isolamento della sua vita privata. Eccentrico in maniera analoga a Glenn Gould, anche il pianista canadese appariva sagace, brillante e solitario. Per certi versi, Stallman si considera affetto da autismo: malattia che, sostiene, ne rende difficile l’interazione con gli altri.

[4] Si veda Steve Silberman, “The Geek Syndrome”, *Wired* (dicembre, 2001). http://www.wired.com/wired/archive/9.12/aspergers_pr.html

[5] Purtroppo non sono riuscito a intervistare Daniel Stallman. Durante le prime fasi di stesura del libro, Stallman mi aveva informato che il padre era affetto dal morbo di Alzheimer. Quando ripresi in mano il progetto sul finire del 2001, appresi con tristezza che Daniel Stallman era scomparso poco prima.

[6] Probabilmente l’ateo Stallman avrebbe voluto precisare meglio questa descrizione. Fatto sta che però gli sia piaciuta. Si veda la precedente nota 1: “Non appena sentii parlare dei computer, mi venne voglia di vederne uno per poterci giocare.”

Processiamo Dio

Sebbene il rapporto con la madre fosse carico di tensione, Richard Stallman finirà per ereditare da lei un importante tratto caratteriale: la passione per le posizioni politiche progressiste.

Si tratta però di una caratteristica che impiegherà svariati decenni per emergere in superficie. Durante i primi anni della sua vita, Stallman si trovò in quello che oggi ammette essere un "vuoto politico"[1]. Come la maggioranza degli americani all'epoca di Eisenhower, la famiglia Stallman trascorse gli anni '50 cercando di riconquistare quella normalità perduta negli anni della guerra.

"Io e il padre di Richard ci consideravamo democratici ma senza spingerci oltre", spiega Alice Lippman, ricordando gli anni familiari passati nel Queens. "Non c'interessava granché la politica nazionale o locale."

Tutto ciò prese a cambiare, tuttavia, sul finire degli anni '50 quando Alice divorziò da Daniel Stallman. Il ritorno a Manhattan fu qualcosa di più che un semplice cambio d'indirizzo; rappresentò il sorgere di un'identità nuova e indipendente, oltre alla dolorosa perdita di tranquillità.

"Credo che il mio primo assaggio di attivismo politico fu quando andai alla biblioteca pubblica del Queens e scoprii che in tutti gli scaffali esisteva un solo volume sul divorzio", ricorda la Lippman. "Si respirava la forte presenza della Chiesa Cattolica, almeno nell'area di Elmhurst dove vivevamo noi. Direi che fosse il primo sentore di quei poteri che controllano le nostre vite in maniera silenziosa."

Tornando nel quartiere in cui era cresciuta, l'Upper West Side di Manhattan, rimase assai colpita dai cambiamenti avvenuti da quando se ne era andata, poco più di dieci anni prima, per frequentare l'Hunter College. La

pressante domanda del dopoguerra per nuove abitazioni aveva trasformato il quartiere in un campo di battaglia politico. Da una parte c'erano consiglieri comunali e imprenditori che spingevano per lo sviluppo edilizio nella speranza di poter ricostruire gran parte dei vecchi edifici ove alloggiare il crescente numero di impiegati che inondavano la città. Sul fronte opposto stavano invece gli inquilini irlandesi e portoricani meno abbienti, i quali avevano scelto quel quartiere proprio per via degli affitti contenuti.

Inizialmente Alice Lippman non sapeva bene in quale delle due fazioni riconoscersi. Come nuovo residente aveva certamente bisogno di un alloggio moderno. Come madre sola dal basso reddito, tuttavia, condivideva le preoccupazioni dei residenti più poveri rispetto al crescente numero di progetti edilizi mirati principalmente a residenti benestanti. Indignata, cominciò a cercare i modi adatti per combattere quella macchina politica che stava cercando di trasformare il suo quartiere in un duplicato dell'Upper East Side.

Fu così che nel 1958 visitò per la prima volta la sezione locale del Partito Democratico. Cercando un asilo nido dove portare il figlio mentre lavorava, era rimasta terrificata dalle condizioni riscontrate in uno dei centri gestiti dalla municipalità per i residenti dal basso reddito. "Tutto quello che ricordo è l'odore di latte rancido, le stanze buie, la scarsità di cibo. Avevo insegnato in asili nido privati e c'era una differenza enorme. Demmo appena un'occhiata al posto e ce ne andammo. Ero scandalizzata."

Purtroppo la visita a quella sezione di partito si rivelò deludente. Descrivendola come "la proverbiale stanza del fumo", sostiene di essersi resa conto per la prima volta come fosse proprio la corruzione esistente all'interno del partito il motivo principale dell'ostilità, a malapena celata, delle autorità cittadine nei confronti dei residenti meno abbienti. Senza mai tornare in sezione, decise invece di aggregarsi ad uno dei numerosi gruppi la cui attività mirava a riformare il Partito Democratico onde liberarsi delle ultime vestigia della "Tammany Hall Machine"[2]. Riuniti sotto il Woodrow Wilson/FDR Reform Democratic Club, Lippman e gli altri attivisti iniziarono così a partecipare alle riunioni del consiglio comunale sullo sviluppo edilizio, chiedendo di poter avere voce in capitolo.

“Nostro obiettivo primario era l’opposizione a Tammany Hall, Carmine DeSapio e i loro compari”[3] ricorda Lippman. “Io fungevo da rappresentante presso il consiglio comunale ed ero molto coinvolta nella creazione di un progetto di rinnovo urbanistico sostenibile che andasse oltre la semplice aggiunta di alloggi di lusso nel quartiere.”

Un coinvolgimento che nei successivi anni ’60 sfociò in un attivismo politico di portata assai maggiore. Nel 1965 Lippman lavorò alacremente nelle campagne elettorali per il sostegno a candidati come William Fitts Ryan, parlamentare democratico eletto con l’aiuto dei gruppi di riforma e uno dei primi deputati statunitensi a intervenire pubblicamente contro la guerra in Vietnam.

Non ci volle molto perché anche Lippman manifestasse la propria opposizione all’intervento USA in Indocina. “Mi schierai contro la guerra in Vietnam fin dal momento in cui Kennedy inviò le truppe”, ricorda. “Leggevo gli articoli di reporter e giornalisti inviati a seguire le prime fasi del conflitto. Ritenevo del tutto azzeccate le loro previsioni secondo cui si sarebbe trasformato in un grande pasticcio.”

Un clima di dissenso che si respirava anche nella famiglia Stallman-Lippman. Nel 1967 Alice si risposò. Il suo nuovo marito, Maurice Lippman, maggiore della Air National Guard, rassegnò le sue dimissioni per dimostrare la sua opposizione alla guerra. Il figlio di Maurice Lippman, Andrew, studiava al MIT e in quanto studente poteva rinviare temporaneamente il servizio militare. Tuttavia la minaccia che tale rinvio potesse essere sospeso, come alla fine accadde, rendeva ancora più concreta la possibilità di un ulteriore coinvolgimento bellico degli Stati Uniti. C’era infine Richard che, sebbene più giovane, aveva davanti a sé la prospettiva di optare tra Vietnam o Canada quando la guerra continuò negli anni ’70.

“Il Vietnam era una faccenda importante in famiglia”, sostiene la Lippman. “Ne parlavamo in continuazione: cosa avremmo fatto nel caso la guerra fosse proseguita, quali i passi migliori per Richard o per il fratellastro se fossero stati chiamati alle armi. Eravamo davvero convinti fosse qualcosa d’immorale.”

Per Stallman, la guerra del Vietnam suscitava un miscuglio complesso di emozioni: confusione, orrore e in ultima analisi una profonda sensazione di impotenza politica. Nei panni di un ragazzo appena in grado di far fronte all'universo leggermente autoritario della scuola privata, Stallman tremava al solo pensiero di dover entrare in un campo di addestramento dell'esercito.

"Ero devastato dalla paura, ma non riuscivo a immaginare cosa fare e non avevo il coraggio di unirmi alle dimostrazioni di protesta", rammenta Stallman, il cui compleanno del 18 marzo lo pose paurosamente a rischio nella lotteria per la chiamata di leva, quando nel 1971 il governo decise di cancellare i rinvii per motivi di studio. "Non mi vedevo scappar via in Canada o in Svezia. Ero terrorizzato dall'idea di dovermi alzare e fare qualcosa in prima persona. Avrei mai potuto farcela? Non sapevo vivere e mantenermi da solo. Non ero il tipo che si sente a proprio agio in una situazione simile."

Quando gli altri della famiglia esprimevano le proprie opinioni in pubblico, Stallman ne rimaneva assai colpito ma provava anche vergogna. Ricordando un adesivo sul parafango della macchina del padre che equiparava il massacro di My Lai alle atrocità naziste della Seconda Guerra Mondiale, dice di essersi sentito "eccitato" per quel gesto d'oltraggio del padre. "Lo ammiravo per quell'azione", dice Stallman. "Ma io non riuscivo a far nulla. Avevo paura che il Moloch della leva stesse per annientarmi."

Nonostante queste descrizioni sull'incapacità personale a prendere posizione contengano un certa dose di rimpianto nostalgico, Stallman sostiene che alla fine rimase deluso dal tono e dalla direzione presi dal movimento contro la guerra. Al pari di altri studenti dello Science Honors Program, considerò quelle dimostrazioni di protesta nei weekend alla Columbia poco più di un distratto spettacolo[4]. In conclusione, ribadisce Stallman, divenne impossibile distinguere tra le forze irrazionali che guidavano il movimento contro la guerra e quelle, altrettanto irrazionali, alla guida del resto della cultura giovanile. Anziché adorare i Beatles, improvvisamente le coetanee di Stallman idolatravano dei tipi sputafuoco quali Abbie Hoffman e Jerry Rubin. Per un ragazzo già pieno di problemi nel comprendere i coetanei, slogan disimpegnati come "fate l'amore non la guerra", contenevano messaggi

ambigui. Non soltanto ciò serviva a rammentare che Stallman, l'asociale dai capelli a spazzola che odiava il rock'n'roll, detestava le droghe e non partecipava alle dimostrazioni nel campus, non comprendeva cosa stesse accadendo a livello politico; non lo "capiva" neppure sessualmente.

"La controcultura non mi ha mai colpito granché", ammette oggi Stallman. "Quella musica non m'interessava, tantomeno le droghe. Avevo paura di quelle sostanze. Non mi andavano giù soprattutto l'antiintellettualismo e i pregiudizi contro la tecnologia. In fondo, io amavo il computer. E non mi piaceva lo sciocco anti-americanismo in cui mi capitava spesso d'imbattermi. Qualcuno pensava in maniera talmente semplicistica che quell'opposizione alla condotta statunitense nella guerra in Vietnam dovesse comportare l'automatico appoggio al Vietnam del Nord. Suppongo che quella gente non riuscisse a immaginare una situazione più complessa."

Simili commenti servono ad alleviare la sensazione di timidezza, ponendo altresì in evidenza un tratto che diverrà la chiave della maturazione politica di Stallman. Per quest'ultimo, la dimestichezza politica dev'essere direttamente proporzionale a quella personale. Nel 1970 Stallman aveva confidenza con pochi elementi oltre il regno della matematica e della scienza. Ciò nonostante, la familiarità con i numeri gli offrì le basi per esaminare il movimento contro la guerra in termini puramente logici. Lungo questo processo, Stallman trovò gli strumenti della logica particolarmente adeguati. Per quanto contrario all'intervento bellico in Vietnam, non vedeva motivo per rifiutare la guerra come mezzo atto a difendere la libertà o a correggere le ingiustizie. Tuttavia, anziché allargare il divario tra lui e i colleghi, Stallman decise di tenere per sé questo tipo di analisi.

Nel 1970, con la partenza per Harvard, si lasciò alle spalle le quotidiane conversazioni a cena sulla politica e sulla guerra in Vietnam. Col senno di poi Stallman descrive il passaggio dall'appartamento di sua madre a Manhattan alla vita nel dormitorio di Cambridge come una "fuga". Ma gli amici che lo seguirono in quel passaggio notarono ben poco a sostegno della tesi per cui si trattasse di un'esperienza liberatoria.

“I suoi primi tempi ad Harvard sembravano piuttosto infelici”, ricorda Dan Chess, compagno di classe nello Science Honors Program e anch’egli matricola ad Harvard. “Era evidente come l’interazione umana gli risultasse davvero difficile, ma in quell’ambiente non c’era modo di evitarla. Harvard era un luogo intensamente sociale.”

Per facilitare la transizione, Stallman fece nuovamente affidamento sui suoi punti di forza, scienza e matematica. Come molti studenti provenienti dallo Science Honors Program, anche Stallman superò facilmente l’esame di ammissione per Math 55, il corso riservato alle matricole dotate in matematica, diventato leggendario per la sua difficoltà. Al suo interno, quanti provenivano dal programma della Columbia University formarono subito un’unità di ferro. “Eravamo la mafia della matematica”, dice Dan Chess con una risata. “Harvard valeva ben poco, almeno in paragone allo Science Honors Program.”

Prima di guadagnarsi il diritto a vantarsene, però, Stallman, Chess e gli ex studenti di quel programma, dovevano superare il corso di Math 55. Quest’ultimo condensava quattro anni di lezioni in appena due semestri, e mirava soltanto ai veri adepti. “Era un corso davvero incredibile”, sostiene David Harbater, ex aderente alla “mafia della matematica” e oggi professore della stessa materia presso la University of Pennsylvania. “È plausibile sostenere che non sia mai esistito un corso per matricole talmente intenso e avanzato. Quel che dicevo agli altri giusto per dare loro un’idea, era che tra le altre cose nel secondo trimestre discutevamo la geometria differenziale della varietà di Banach. A quel punto strabuzzavano gli occhi, perché generalmente tale geometria compariva dopo il secondo anno.”

Partito con 75 studenti, il corso si ridusse rapidamente a 20 nel secondo semestre. Di costoro, aggiunge Harbater, “soltanto 10 sapevano veramente cosa stavano facendo”. Otto di questi sarebbero divenuti a loro volta professori di matematica e uno avrebbe insegnato fisica. “L’ultimo rimasto”, sottolinea Harbater, “era Richard Stallman.”

Seth Breidbart, studente di Math 55, ricorda come anche in quell’ambito Stallman riuscisse a distinguersi dagli altri.

“Era pignolo, ma in modo strano”, dice Breidbart. “In matematica c’è una tecnica standard che tutti sbagliano. Si tratta di un uso errato della notazione, quando bisogna definire una funzione e quello che in realtà si fa è definire prima la funzione e poi dimostrare che è definita correttamente. Solo che, la prima volta che si trovò a presentarla, egli definì una relazione dimostrando poi che è una funzione. È la stessa identica dimostrazione, ma in cui Stallman aveva utilizzato la terminologia corretta, cosa che non aveva fatto nessun altro. Questo per dire come era fatto.”

Fu durante il corso di Math 55 che Richard Stallman iniziò a coltivare la reputazione di studente brillante. Breidbart si dichiara d’accordo, ma secondo Chess, le cui ambizioni competitive rifiutarono di cedere il passo, Stallman non venne riconosciuto come il migliore in matematica fino all’anno successivo. “Avvenne durante il corso di Analisi Matematica”, spiega Chess, oggi professore di matematica presso l’Hunter College. “Ricordo una dimostrazione sui divisori a valori complessi dove Richard se ne uscì con un’idea che praticamente era una metafora del calcolo delle varianti. Era la prima volta che vedevo qualcuno risolvere un problema in maniera così brillante e originale.”

Per Chess fu un momento difficile. Come un uccello che, volando, finisce contro il vetro di una finestra lucida, gli ci volle un po’ per rendersi conto che questo livello di intuito era semplicemente fuori dalla sua portata.

“Questo per quanto riguarda la matematica”, sostiene Chess. “Non occorre un matematico di massimo grado per riconoscere un talento eccezionale. Ero cosciente di aver raggiunto un ottimo livello, ma sapevo anche di non potermi considerare un matematico eccelso.”

Per Stallman, i successi scolastici erano inversamente proporzionali a quelli nell’arena sociale. Perfino quando gli aderenti alla mafia della matematica si riunivano per risolvere insieme i problemi, Stallman preferiva lavorare da solo. Lo stesso dicasi per la sistemazione dell’alloggio. Nella richiesta per una stanza ad Harvard, non mancò di specificare le proprie esigenze. “Dissi che preferivo stare con qualcuno invisibile, inascoltabile, intangibile.” In un evento di rara perspicacia burocratica, i responsabili amministrativi

accettarono quella richiesta, assegnando a Stallman una stanza da solo per quell'anno da matricola.

Breidbart, unico aderente alla mafia della matematica che alloggiasse nel dormitorio oltre a Stallman, segnala che quest'ultimo imparò ad interagire con gli altri studenti in maniera graduale ma sicura. Egli ricorda come gli altri residenti del dormitorio, colpiti dall'acume logico di Richard, presero ad apprezzare i suoi suggerimenti ogni volta che si dava vita a qualche tipo di discussione intellettuale in mensa o nelle sale comuni.

"Tenevamo le solite chiacchierate da bulli per risolvere i problemi del mondo o per capire quale sarebbe stato il risultato di una certa premessa", rammenta Breidbart. "Supponiamo che qualcuno abbia scoperto il siero dell'immortalità. Cosa faresti? Quali le conseguenze politiche? Fornendolo a tutti, il mondo diventerebbe sovrappopolato, e moriremmo tutti. Se si impongono delle limitazioni, stabilendo ad esempio di darlo a chi è vivo oggi ma non ai propri figli, allora finiremmo per creare un ceto di persone inferiori, quelle private dell'accesso al siero. Richard era capace, in modo superiore agli altri, di immaginare le circostanze imprevedibili susseguenti a questo tipo di scenari."

Stallman ricorda assai vividamente quelle discussioni. "Ero sempre a favore dell'immortalità", sostiene. "Rimanevo colpito dal fatto che la maggior parte delle persone la considerasse una cosa negativa. In quale altro modo potremmo osservare i cambiamenti del mondo da qui a 200 anni?"

Pur affermandosi come eccelso matematico e sagace oratore, evitava le competizioni pubbliche che avrebbero sancito la sua brillante reputazione. In conclusione di quell'anno da matricola ad Harvard, Breidbart ricorda come Stallman schivò ripetutamente l'esame Putnam, un prestigioso test riservato agli studenti di matematica in USA e Canada. Oltre a fornire loro la possibilità di misurare le proprie conoscenze rispetto ad altri coetanei, quell'esame era considerato uno strumento per ottenere l'assunzione presso i vari dipartimenti accademici di matematica. Stando alle leggende del campus, il primo classificato avrebbe automaticamente ottenuto una borsa di studio per proseguire gli studi presso un istituto di propria scelta, incluso Harvard.

Come per il corso di Math 55, l'esame Putnam era brutalmente basato sulle capacità mentali. La sessione di sei ore divisa in due parti, sembrava esplicitamente progettata per eliminare ogni dubbio sui più bravi. Breidbart, veterano sia dello Science Honors Program sia del corso di Math 55, lo descrive tranquillamente come l'esame più difficile che abbia mai sostenuto. "Giusto per avere un'idea del livello di difficoltà", spiega, "il punteggio massimo era 120, e nel primo anno il mio risultò intorno a 30, dimostrandosi comunque sufficiente per farmi piazzare al 101° posto nella classifica nazionale."

Sorpreso del fatto che Stallman, il migliore della classe, avesse glissato su quel test, Breidbart ricorda che un giorno lui e gli altri lo costrinsero in un angolo e gli chiesero una spiegazione: "Replicò che temeva di non fare bella figura."

Allora Breidbart e gli altri buttarono giù al volo alcuni problemi che ricordavano a memoria da quell'esame e li passarono a Stallman. "Li risolse tutti", sostiene Breidbart, "portandomi a concludere che la sua idea di non fare bella figura significasse arrivare secondo oppure commettere qualche grossolano errore."

Stallman rammenta l'episodio con qualche differenza. "Ricordo che effettivamente mi passarono dei problemi ed è possibile che li abbia risolti, ma sono quasi certo che non ce la feci con tutti", sostiene. In ogni caso, Stallman è d'accordo con Breidbart sul fatto che non volle sostenere quell'esame essenzialmente per paura. Nonostante l'evidente volontà di mettere in luce la debolezza intellettuale di compagni e professori nel corso delle lezioni, Stallman odiava l'idea della competizione testa a testa.

"È lo stesso motivo per cui non mi è mai piaciuto giocare a scacchi", aggiunge Stallman. "Ogni volta che ci provavo, ero talmente consumato dal terrore di sbagliare una sola mossa che cominciavo a fare stupidi errori fin dall'inizio. La paura divenne così una sorta di profezia autoreferenziale."

Se poi la stessa paura vada ritenuta responsabile della decisione di scansare la carriera di matematico, è questione opinabile. Verso la conclusione del suo anno da matricola, Stallman scoprì altri interessi che finirono per allontanarlo

da quell’ambiente. La programmazione al computer, il cui fascino latente lo aveva accompagnato negli anni delle medie, andava trasformandosi in una vera e propria passione. Laddove gli altri studenti di matematica trovavano occasionale rifugio in materie quali arte e storia, Stallman si rintanava nei laboratori d’informatica.

Il suo primo assaggio di programmazione, all’IBM Scientific Center di New York, ne aveva stimolato il desiderio di saperne di più. “Mentre si chiudeva il mio primo anno ad Harvard, cominciai a sentirmi abbastanza coraggioso da visitare i laboratori d’informatica per vedere cosa avevano. Chiesi in prestito copie di qualunque manuale disponibile.”

Portatosi a casa quei manuali, Stallman avrebbe poi esaminato le specifiche delle macchine, paragonandole a quelle di modelli già conosciuti e buttando giù qualche programma sperimentale, che avrebbe infine riportato al laboratorio insieme ai manuali presi in prestito. Anche se qualche assistente storceva il naso all’idea che un ragazzo qualunque potesse lavorare sulle macchine interne, gran parte di loro finiva per riconoscerne le competenze tecniche, consentendogli così di far girare sul computer i programmi che aveva creato.

Un giorno, ormai al termine dell’anno da matricola, Stallman venne a sapere dell’esistenza di un laboratorio speciale nei pressi del Massachussetts Institute of Technology (MIT). Si trovava al nono piano di un edificio appena fuori dal campus, su Tech Square, nel complesso appena costruito e riservato alla ricerca avanzata. Secondo le voci che giravano, quel laboratorio era dedicato alle tecniche d’avanguardia nel campo dell’intelligenza artificiale e vantava la presenza di macchine e software tra i più sofisticati.

Intrigato, Stallman decise di farvi un salto.

Il percorso era breve, circa tre chilometri a piedi, meno di dieci minuti in tram, ma come Stallman avrebbe scoperto quanto prima, il MIT e Harvard incarnano i poli opposti di uno stesso pianeta. Con il labirinto di corridoi che univa una serie di edifici interconnessi, il campus del MIT presentava un’estetica minimalista rispetto allo spazioso villaggio coloniale di Harvard. Analoga l’impressione riguardo agli iscritti del MIT, una variegata raccolta di

studenti un po’ svitati, noti più per la predilezione alle birichinate che per la brillante carriera politica.

Contrasti che non mancavano di estendersi al laboratorio di intelligenza artificiale. Diversamente dal laboratorio informatico di Harvard, qui non esistevano laureandi-custodi, nessuna lista d’attesa per l’acceso ai terminali, nessuna esplicita atmosfera di “si guarda ma non si tocca”. Stallman trovò soltanto una serie di terminali aperti e braccia robot, probabilmente serviti a qualche esperimento di intelligenza artificiale.

Nonostante le voci dicessero che chiunque poteva sedersi davanti a un computer, Stallman decise di seguire il solito piano iniziale. Quando incontrò uno degli addetti, chiese se non avesse per caso qualche manuale in più da prestare a uno studente curioso. “Ce n’era qualcuno, ma un sacco di cose non erano documentate”, ricorda Stallman. “In fondo erano degli hacker.”

Stallman se andò con qualcosa di meglio di un semplice manuale: un lavoro. Anche se non rammenta bene la natura di quel primo progetto, ricorda di esser tornato al laboratorio la settimana successiva, di essersi seduto davanti a un terminale libero e aver iniziato a scrivere un programma.

Ripensando a quell’evento, Stallman non vede nulla di insolito nel fatto che al laboratorio accettassero il primo venuto. “All’epoca si faceva così”, dice. “E funziona anche adesso. Se mi accorgo che qualcuno è bravo, lo ingaggio appena lo incontro. Perché aspettare? Sbaglia davvero chi si ostina ad attenersi sempre a rigide procedure burocratiche. Se una persona è in gamba, non è necessario che segua un processo lungo e dettagliato prima di essere assunto; meglio metterlo subito davanti a un computer a scrivere codice.”

Per avere un assaggio di rigidità burocratica era sufficiente che Stallman visitasse i laboratori informatici di Harvard. Qui l’accesso ai terminali era subordinato al grado accademico. In quanto studente non ancora laureato, generalmente Stallman doveva mettersi in lista e attendere fino a mezzanotte, quando la maggior parte dei professori e dei laureati finivano i compiti giornalieri. Non era difficile starsene ad aspettare, ma quanto meno frustrante. Dover attendere un terminale pubblico sapendo che nel frattempo una mezza dozzina di macchine ugualmente utilizzabili se ne stavano spente negli uffici

chiusi dei professori, pareva il massimo dell'illogicità. Nonostante Stallman visitasse occasionalmente i laboratori informatici di Harvard, preferiva le procedure più egualitarie in vigore in quello di intelligenza artificiale al MIT. "Era una boccata d'aria fresca", aggiunge. "Qui la gente si preoccupava più del lavoro che della posizione accademica."

Stallman imparò rapidamente che in quell'ambito la pratica del "chi prima arriva, meglio alloggia" si doveva in gran parte agli sforzi di un pugno di persone. Molti erano reduci dall'epoca del progetto MAC, il programma di ricerca sostenuto dal Ministero della Difesa che aveva dato i natali ai primi sistemi operativi a condivisione di tempo (time-sharing). Alcuni erano già leggende del mondo informatico. C'era Richard Greenblatt, l'esperto locale di Lisp e autore di MacHack, programma per giocare a scacchi che una volta aveva umiliato Hubert Dreyfus, critico feroce dell'intelligenza artificiale. C'era Gerald Sussman, creatore originale del programma per funzioni robotiche denominato HACKER. E ancora, Bill Gosper, il mago della matematica già nel bel mezzo di un lavoro di hacking durato 18 mesi, messo in moto dalle implicazioni filosofiche del computer game LIFE[5].

Gli aderenti a questo gruppo ristretto si autodefinivano "hacker". Col passare del tempo allargarono tale definizione allo stesso Stallman. In questo passaggo, egli venne messo al corrente delle tradizioni morali condensate nella "etica hacker". Essere un hacker significava qualcosa di più che sviluppare semplicemente dei programmi, imparò presto Stallman. Voleva dire scrivere il miglior codice possibile e stare seduti davanti a un terminale anche per 36 ore consecutive, se per riuscirci occorreva tutto quel tempo. Fatto ancor più importante, significava aver continuamente accesso alle migliori macchine esistenti e alle informazioni più utili. Gli hacker dicevano apertamente di voler cambiare il mondo tramite il software, e Stallman imparò che l'hacker istintivo supera ogni ostacolo pur di raggiungere un tale nobile obiettivo. Tra questi ostacoli, i maggiori erano rappresentati dal software scadente, dalla burocrazia accademica e dai comportamenti egoistici.

Stallman venne anche a conoscenza del folklore tramandato nel tempo, i racconti di come certi hacker, imbattutisi in qualche intoppo, fossero riusciti a

superarlo in maniera creativa. Apprese il cosiddetto “lock hacking”, l’arte di irrompere negli uffici dei professori per “liberare” i terminali sequestrati. Al contrario della viziata controparte di Harvard, i membri di facoltà al MIT avevano meglio da fare che trattare i computer del laboratorio di intelligenza artificiale come proprietà privata. Se uno di loro aveva commesso l’errore di chiudere in una stanza un terminale per l’intera nottata, gli hacker erano lesti a correggerne lo sbaglio. Altrettanto rapidamente mandavano un chiaro messaggio nel caso l’errore si fosse ripetuto. “Mi venne mostrato un carrello con sopra un pesante cilindro di metallo, usato per rompere la porta dell’ufficio di un professore”[6], ricorda Stallman.

Simili metodi, per quanto privi di finezza, puntavano a un obiettivo preciso. Nonostante professori e amministratori del laboratorio fossero in numero doppio rispetto agli hacker, era l’etica di questi ultimi a prevalere. Non a caso quando arrivò Stallman, gli hacker e gli amministratori del laboratorio avevano sviluppato una sorta di relazione simbiotica. In cambio di attività quali la riparazione delle macchine e il corretto funzionamento del software, gli hacker si erano guadagnati il diritto a lavorare sui progetti preferiti. Che spesso riguardavano l’ulteriore miglioramento delle macchine e dei programmi. Come uno scooter per un adolescente, la miglior forma di divertimento per la maggioranza degli hacker consisteva nel continuare a smanettare con i computer.

Un’attività questa che trova piena rispondenza nel sistema operativo su cui girava il mini-computer centrale del laboratorio, il PDP-6. Soprannominato ITS, acronimo per Incompatible Time Sharing System, questo sistema operativo incorporava l’etica hacker nel cuore del suo stesso progetto. Gli hacker lo avevano realizzato come protesta nei confronti del sistema operativo originale del progetto MAC, noto come Compatible Time Sharing System, CTSS, e anche il nome fu scelto di conseguenza. A quel tempo gli hacker consideravano quest’ultimo progetto troppo restrittivo, poiché limitava le possibilità dei programmatori nell’eventuale modifica e miglioria dell’architettura interna. Secondo una leggenda tramandata da un hacker all’altro, la decisione di costruire il sistema ITS aveva anche connotazioni

politiche. Al contrario del CTSS, realizzato per la serie IBM 7094, l'ITS era stato specificamente progettato per il PDP-6. Nel consentire agli hacker di scrivere quei sistemi in piena autonomia, gli amministratori si erano garantiti il fatto che solo questi ultimi erano poi in grado di usare il PDP-6. All'interno del mondo feudale della ricerca accademica, lo stratagemma funzionò. Nonostante il PDP-6 fosse in comproprietà con altri dipartimenti, ben presto rimase ad esclusivo utilizzo dei ricercatori del laboratorio di intelligenza artificiale[7].

L'ITS includeva funzioni che la maggior parte dei sistemi operativi commerciali non avrebbe offerto per anni, tra cui multitasking, debugging e capacità di editing a tutto schermo. Poggiando sulle fondamenta garantite da queste funzionalità e dal PDP-6, il laboratorio era riuscito a dichiarare la propria indipendenza dal progetto MAC poco prima dell'arrivo di Stallman.

In qualità di apprendista hacker, egli si appassionò immediatamente al sistema ITS. Il sistema presentava numerose opzioni considerate vere e proprie lezioni nello sviluppo del software per un'apprendista come Stallman, anche se tali opzioni erano proibitive per la gran parte dei nuovi venuti.

"L'ITS aveva un meccanismo interno molto elegante che consentiva a un programma di esaminarne un altro", ricorda Stallman. "Si poteva analizzare lo stato di un programma in modo molto pulito e dettagliato."

Fu grazie a questa caratteristica che Stallman notò come i programmi scritti dagli hacker elaboravano le istruzioni man mano che queste comparivano. Un'altra funzione importante consentiva al programma di controllo di bloccare il programma monitorato mentre scorrevano le istruzioni. In altri sistemi operativi ciò avrebbe prodotto semi-elaborati incomprensibili oppure il blocco automatico del sistema. Nell'ITS, quella procedura offriva invece un'ulteriore possibilità di verificarne le prestazioni passo dopo passo.

"Se davi un'istruzione tipo 'Ferma il processo', il sistema si sarebbe sempre fermato nella modalità utente. Si sarebbe bloccato tra due istruzioni di tale modalità, e da quel momento in poi qualsiasi altro comando sarebbe stato conseguente." spiega Stallman. "Se dicevi, 'Continua il processo,' avrebbe proseguito in maniera appropriata. Non solo: se si modificava lo stato del

processo per poi riportarlo allo stato precedente, tutto si sarebbe svolto in maniera coerente. Non esisteva da nessuna parte uno stato nascosto."

Entro la fine del 1970, l'attività di hacking al laboratorio di intelligenza artificiale aveva assunto per Stallman cadenza settimanale. Dal lunedì al giovedì seguiva i suoi corsi ad Harvard. Ma non appena arrivava il venerdì pomeriggio, eccolo prendere il tram T per recarsi giù al MIT, dove avrebbe trascorso l'intero week-end. Generalmente faceva coincidere il proprio arrivo con la rituale caccia a qualcosa da mettere sotto i denti. Unirsi ad altri cinque o sei hacker alla ricerca notturna di cibo cinese significava saltare in una macchina scassata e attraversare l'Harvard Bridge per raggiungere la vicina Boston. Nelle due ore successive il gruppo avrebbe chiacchierato di tutto un po', dal sistema ITS alla logica interna della lingua cinese e al relativo sistema pittografico. Dopo cena, sarebbero tornati al MIT per lavorare al codice fino all'alba.

Per l'asociale che si univa raramente ai compagni delle medie, si trattava di un'esperienza non certo da poco, andarsene tutt'a un tratto in giro con persone che condividevano la sua stessa predilezione per i computer, la fantascienza e il cibo cinese. "Ricordo d'aver visto molte volte l'alba tornando in macchina da Chinatown", disse Stallman con una punta di nostalgia 15 anni dopo, in un discorso tenuto allo Swedish Royal Technical Institute. "L'alba era davvero un spettacolo meraviglioso, perché è un momento così calmo della giornata. Periodo ideale per prepararsi ad andare a letto. È così bello camminare verso casa tra le prime luci del giorno con gli uccelli che iniziano a cinguettare; si prova una concreta sensazione di soddisfazione e tranquillità rispetto al lavoro svolto durante la notte."[8]

Continuando a frequentare quegli hacker, Stallman finì con l'adottarne la visione del mondo. Già sostenitore della libertà individuale, prese a infondere un senso di responsabilità comune in ogni sua azione. Quando gli altri violavano delle norme condivise, Stallman saltava subito su a evidenziarlo. A un anno dalla sua prima visita, eccolo irrompere negli uffici chiusi a chiave per recuperare i terminali sequestrati che appartenevano all'intera comunità del laboratorio. Nella tipica attitudine hacker, non rinunciò a offrire i propri

contributi all’arte del cosiddetto “lock hacking” (come forzare una serratura chiusa). Uno dei trucchi più creativi per aprire quelle porte, comunemente attribuito a Greenblatt, consisteva nel curvare un pezzo di fil di ferro e attaccarvi del nastro adesivo all’estremità più lunga. Dopo aver fatto scivolare sotto la porta il cavetto, lo si poteva ruotare in modo che l’estremità con il nastro adesivo arrivasse a toccare la maniglia interna. Ammesso che il nastro tenesse, un paio di strattoni e la porta si sarebbe aperta.

Dopo averlo provato, Stallman trovò il trucchetto buono ma inadeguato sotto alcuni punti di vista. Far aderire il nastro al fil di ferro era piuttosto difficile, e anche riuscire a girare la maniglia con uno strattone non era facile. Stallman ricordò che il soffitto dei corridoi aveva dei pannelli rimovibili. In realtà più di un hacker era ricorso allo stratagemma del controsoffitto per penetrare negli uffici chiusi, strategia che in genere funzionava, anche se ne uscivi coperto di fibra di vetro.

Stallman considerò allora un approccio alternativo. Invece di far scivolare un cavo sotto la porta, perché non cercare di spostare uno di quei pannelli e operare dallo stipite della porta?

Decise di provarci da solo. Anziché ricorrere al cavo metallico, preparò un nastro magnetico lungo a forma di U, alla cui base sistemò un ovale di nastro adesivo. Da sopra lo stipite, fece cadere quell’aggeggio finché non s’incastrò al di sotto del pomello. Dopo averlo spostato in modo da far aderire il nastro adesivo al pomello stesso, prese a tirare finché non riuscì a farlo ruotare. Ed ecco che la porta si aprì. Stallman aveva aggiunto un trucco nuovo all’arte del lock hacking.

“Talvolta bisognava dare un calcio alla porta dopo aver girato la maniglia”, dice Stallman rammentando le difficoltà del nuovo metodo. “E ci voleva un po’ di equilibrio nel tirare.”

Simili attività riflettevano la sua crescente volontà di parlare e agire in difesa del proprio credo politico. Lo spirito del laboratorio a sostegno dell’azione diretta lo aveva ispirato a spezzare la timida impotenza dei suoi anni giovanili. Forzare una porta per liberare un terminale non era lo stesso

che partecipare ad una manifestazione di dissenso, ma risultava efficace forse più di una protesta. Risolveva il problema di quel momento.

Negli ultimi anni trascorsi ad Harvard, Stallman prese ad applicare anche in quell'ambito le bizzarre e irriverenti lezioni apprese al laboratorio di intelligenza artificiale.

"Ti ha raccontato del serpente?", mi chiede sua madre durante un'intervista. "Insieme con il suo compagno di stanza, avevano presentato un serpente come candidato al consiglio studentesco. Pare che avesse ottenuto un considerevole numero di voti."

Stallman conferma la candidatura del serpente con alcune precisazioni. Si trattava di elezioni limitate alla Currier House, il dormitorio di Stallman, non per le cariche di consiglio dell'intero campus.

Sì, il serpente ottenne un risultato significativo, in buona parte grazie al fatto che sia il serpente sia il proprietario condividevano il medesimo cognome. "La gente lo votò pensando di votare per quella persona", precisa Stallman. "I manifesti della campagna riportavano che il serpente avrebbe 'strisciato' pur di arrivare a quella carica. Specificammo inoltre che si trattava di un candidato 'latitante', perché qualche settimana prima si era arrampicato sul muro per poi sparire nel condotto di ventilazione e nessuno l'aveva più visto."

La presentazione di un serpente come candidato al consiglio del dormitorio non era altro che una delle numerose burle collegate alle elezioni locali. In un'ulteriore circostanza, Stallman e altri studenti proposero il figlio di un dipendente universitario. "La sua piattaforma prevedeva il pensionamento obbligatorio all'età di sette anni", ricorda Stallman.

Tuttavia simili scherzi impallidivano al confronto di quelli analoghi organizzati nel campus del MIT. Uno dei più riusciti riguardava la candidatura di un gatto chiamato Woodstock, che in realtà ottenne più voti di gran parte dei candidati umani in un'elezione che riguardava l'intero campus. "Il numero dei voti ricevuti da Woodstock non venne mai reso noto, li considerarono nulli", rammenta Stallman. "Ma l'elevata quantità di voti nulli sembrò suggerire che in realtà avesse vinto. Un paio d'anni dopo, il gatto venne

investito da un macchina in circostanze sospette. Nessuno è mai riuscito a sapere se l'autista lavorasse per l'amministrazione del MIT." Stallman sostiene di non aver avuto nulla a che far con la candidatura di Woodstock, "pur considerandomi un suo ammiratore."[9]

Al laboratorio di intelligenza artificiale, le attività politiche di Stallman andavano assumendo connotazioni decisamente più serie. Durante gli anni '70, gli hacker dovevano fronteggiare continuamente membri di facoltà e amministratori decisi a costruire una sorta di cordone intorno all'ITS e ai relativi progetti su misura per quegli hacker. Uno dei primi tentativi in tal senso avvenne verso il 1975, quando un crescente numero di professori iniziò a richiedere un sistema di sicurezza a tutela dei dati delle ricerche. La gran parte degli altri laboratori informatici aveva installato sistemi analoghi sul finire degli anni '60, ma il laboratorio di intelligenza artificiale, grazie all'insistenza di Stallman e di altri hacker, rimaneva una zona franca.

Per Stallman, l'opposizione ai sistemi di sicurezza era di natura sia pratica che etica. Da quest'ultimo punto di vista, egli non mancava di sottolineare come l'intera arte dell'hacking fosse basata sull'apertura e sulla fiducia intellettuale. Rispetto al lato pratico, ribadiva come la struttura interna dell'ITS fosse aderente a questo spirito di apertura, e ogni tentativo di re-impostare il tutto si sarebbe rivelata un'operazione assai complessa.

"Gli hacker cui si deve l'Incompatible Time Sharing System si resero conto di come la protezione dei file venisse solitamente usata dal gestore di un sistema per guadagnare potere rispetto a tutti gli altri", così recita la successiva spiegazione di Stallman. "Non volevano che qualcuno potesse arrivare a tanto, perciò decisero di non implementare quel tipo di funzione. Come risultato, ogni qual volta nel sistema si verificavano dei problemi era sempre possibile risolverli."[10]

Fu grazie a questo tipo di vigilanza che gli hacker consentirono alle macchine del laboratorio di intelligenza artificiale di rimanere immuni da ogni funzione di sicurezza. Al contrario, però, di quanto accadde nel vicino laboratorio d'informatica, sulla spinta dei membri di facoltà: qui il primo sistema protetto da password venne installato nel 1977. Ancora una volta fu

Stallman ad assumersi la responsabilità di correggere quel che considerava una sorta di lassismo etico. Avuto accesso al codice del software che controllava il sistema delle password, vi introdusse un comando che inviava un messaggio agli utenti del laboratorio d'informatica quando uno di questi si apprestava a creare una password. Se per esempio qualcuno sceglieva "starfish", veniva generato un messaggio che diceva all'incirca:

> Vedo che hai scelto "starfish" come password. Ti suggerisco di modificarla in "carriage return." E' molto più facile da digitare e aderisce al principio che nega l'esistenza di alcuna password.[11]

Gli utenti che optavano per "carriage return" – quelli cioè che semplicemente premevano il tasto Invio, lasciando in bianco la stringa della password invece di digitarne una personale – consentivano l'accesso al mondo intero, tramite il proprio account. Una pratica che, per quanto preoccupasse qualcuno, rinforzava il concetto secondo cui i computer del MIT, e perfino i file contenuti, appartenevano al pubblico anziché ai singoli individui. Nel corso di un'intervista per il libro del 1984 *Hackers*, Stallman fece notare con orgoglio come un quinto dello staff del laboratorio d'informatica aderì a quella posizione, lasciando in bianco la stringa per la password.[12]

In definitiva però la crociata di Stallman si rivelò inutile. All'inizio degli anni '80 anche le macchine del laboratorio di intelligenza artificiale finirono per dotarsi di sistemi di sicurezza basati sulle password. E tuttavia l'episodio rappresentò una pietra miliare lungo il percorso della maturazione personale e politica di Stallman. Osservando con occhio attento le sue vicende successive, quell'evento si pone come punto di passaggio tra il timido adolescente terrorizzato a intervenire in pubblico persino su questioni d'importanza vitale e l'attivista adulto che avrebbe presto trasformato quell'attività di provocatore in occupazione a tempo pieno.

Nell'opporsi a piena voce ai sistemi di sicurezza, Stallman non fece altro che riflettere quelle forze su cui si era formato da bambino: sete di conoscenza, disgusto per l'autorità, frustrazione per procedure e regole nascoste che emarginavano quanti le ignoravano. In tal modo venivano inoltre evidenziati quei capisaldi morali che avrebbero dato successivamente forma alla sua vita adulta: la condivisione di responsabilità, la fiducia, lo spirito

hacker mirato all’azione diretta. Ricorrendo alla terminologia informatica, si può dire che quella stringa vuota rappresentava la versione 1.0 della concezione politica globale di Richard Stallman – incompleta in alcuni punti, ma per la gran parte giunta a piena maturità.

Col senno di poi, egli appare esitante nell’attribuire un significato eccessivo ad un evento avvenuto praticamente all’inizio della carriera di hacker. “A quei tempi i miei sentimenti erano condivisi da parecchia gente”, sostiene Richard. “L’ampio numero di persone che adottarono una stringa vuota come password è la testimonianza del fatto che per molti ciò fosse una scelta giusta. Io ero semplicemente incline a sostenere il ruolo dell’attivista.”

Stallman riconosce invece al laboratorio d’intelligenza artificiale il merito di aver risvegliato in lui quel ruolo di attivista. Da ragazzo aveva osservato gli eventi politici senza comprendere in che modo un singolo individuo potesse fare o dire qualcosa d’importante. Divenuto adulto, si era fatto avanti su questioni in cui si sentiva a proprio agio, temi quali la progettazione del software, la condivisione di responsabilità e la libertà individuale. “Ero entrato a far parte di questa comunità che professava il pieno rispetto della libertà reciproca”, sostiene. “Non ci misi molto a rendermi conto di quanto questo fosse positivo. Mi ci volle di più per capire come si trattasse in realtà di una questione morale.”

L’attività di hacking al laboratorio d’intelligenza artificiale non era l’unico contributo alla crescita della fiducia in se stesso. Verso la metà dell’anno da matricola ad Harvard, Stallman aveva aderito a un gruppo specializzato in danze popolari. Quel che era iniziato come un semplice tentativo di incontrare qualche ragazza e ampliare i propri orizzonti sociali, si tramutò presto in una nuova passione parallela a quella dell’hacking. Mentre si esibiva di fronte al pubblico con il costume di un contadino dei Balcani, Stallman appariva ben diverso dal ragazzino scoordinato, frustrato per i tentativi falliti di giocare a calcio. Si sentiva sicuro, agile e vivo. Per qualche breve istante, provò perfino la sensazione di un legame emotivo. Si rese subito conto di quanto fosse piacevole danzare davanti a degli spettatori, e finì presto per bramare tanto lo spettacolo in sé quanto il suo aspetto sociale.

Pur se la danza e l'hacking non riuscirono a migliorare di molto la situazione sociale di Stallman, lo aiutarono a superare quella sensazione di stranezza che ne aveva contrassegnato l'esistenza nel periodo pre-Harvard. Anziché doversi lagnare dei propri difetti caratteriali, trovò il modo per celebrarli. Nel 1977, mentre partecipava a un incontro sulla fantascienza, s'imbatté in una ragazza che vendeva spillette con slogan personalizzati. Eccitato, Stallman gliene ordinò una con la scritta "Processiamo Dio."

Per Stallman si trattava di un messaggio che operava a diversi livelli. Ateo fin dalla prima giovinezza, quella frase rappresentava il tentativo di implementare un "secondo fronte" nel dibattito aperto sulla religione. "A quel tempo tutti parevano chiedersi se Dio fosse vivo o morto", rammenta Stallman. "Quel 'Processiamo Dio' proponeva un approccio completamente diverso al problema. Se Dio era davvero così potente da aver creato il mondo senza tuttavia far nulla per correggerne i problemi, perché mai avremmo dovuto adorare un tale Dio? Non sarebbe stato invece meglio metterlo sotto processo?"

Contemporaneamente lo slogan andava inteso come una battuta ironica contro l'America e il relativo sistema politico. Negli anni '70 Stallman era rimasto molto colpito dallo scandalo Watergate. Fin da piccolo aveva imparato a non aver fiducia nell'autorità costituita. Ora, da adulto, quella sfiducia era stata rafforzata dalla cultura in vigore nella comunità hacker del laboratorio di intelligenza artificiale. Per gli hacker lo scandalo Watergate non era altro che un adattamento shakespeariano di quella lotta quotidiana per il potere colpevole di rendere misera l'esistenza di coloro senza privilegi. Era una parabola amplificata di quel che accade quando la gente rinuncia alla libertà e all'apertura in cambio della sicurezza e della convenienza.

Sostenuto da una crescente fiducia in se stesso, Stallman prese a indossare la spilletta con orgoglio. Ai più curiosi che s'azzardavano a chiedergli spiegazioni, impartiva un'identica litania ben preparata: "Il mio nome è Jehovah", replicava Stallman. "Ho un progetto speciale per la salvezza dell'universo, ma per motivi di sicurezza a livello del paradiso non posso rivelare nulla di più. Devi soltanto aver fiducia in me, perché soltanto io sono

in grado di vedere come stanno le cose. Tu sai che io rappresento il bene supremo perché così ti ho insegnato. Se non avrai fede in me, ti metterò sulla lista dei nemici per gettarti nell'abisso dove l'Ufficio Infernale delle Imposte passerà al vaglio le tue dichiarazioni dei redditi da qui all'eternità."

Coloro che interpretavano la litania come una parodia letterale delle udienze sullo scandalo Watergate afferravano soltanto metà del messaggio. Stallman voleva anche riferirsi a un ambito comprensibile soltanto ai colleghi hacker. Un secolo dopo l'avvertimento di Lord Acton sull'equazione tra potere assoluto e corruzione altrettanto assoluta, gli americani sembravano aver dimenticato la prima parte del truismo di Acton: è il potere stesso a corrompere. Anziché porre in evidenza i numerosi esempi di flagrante corruzione, Stallman parve soddisfatto di esprimere in tal modo il proprio oltraggio nei confronti di un intero sistema che venerava il potere al di sopra di tutto.

"Mi chiedevo come mai ci si fermasse ai pesci piccoli", spiega Stallman, ricordando quella spilla e il relativo messaggio. "Se ce la prendevamo con Nixon, perché non fare lo stesso con il Sig. Grande? Dal mio punto di vista, chiunque detenesse il potere e ne avesse abusato, meritava di esserne spogliato."

[1] Si veda Michael Gross, "Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius" (1999).

[2] Tammany Hall, sede del Partito Democratico a New York, è diventato sinonimo di corruzione politica negli USA a causa di episodi svoltisi nella prima metà del Novecento. [*N.d.R.*]

[3] Carmine DeSapio detiene l'equivoco primato di essere stato il primo boss italoamericano di Tammany Hall, la macchina politica di New York City. Per maggiori dettagli su DeSapio e le politiche del dopoguerra a New York, si veda John Davenport, "Skinning the Tiger: Carmine DeSapio and the End of the Tammany Era", *New York Affairs* (1975), 3, 1.

[4] Chess, altro ex-studente del Columbia Science Honors Program, descrive le proteste come "rumore di sottofondo". E aggiunge: "Eravamo tutti politici, ma quel corso era troppo importante per noi. Non avremmo mai saltato una lezione per partecipare a quelle manifestazioni."

[5] Si veda Steven Levy, *Hackers*, Penguin USA, 1984, p. 144. Levy dedica circa cinque pagine alla descrizione del fascino esercitato su Gosper da LIFE, gioco informatico basato sul calcolo creato dal

matematico britannico John Conway. Raccomando caldamente il libro di Levy come supplemento, fors'anche come prerequisto, a questo volume.

[6] Gerald Sussman, membro di facoltà al MIT e hacker il cui lavoro al laboratorio di intelligenza artificiale fece seguito a quello di Stallman, contesta questo ricordo. Secondo Sussman, gli hacker non avrebbero mai fatto irruzione in alcun ufficio per liberarne i terminali.

[7] Mi scuso per il veloce riassunto sulla genesi dell'ITS, sistema operativo che numerosi hacker considerano tuttora l'epitome dell'etica hacker. Per maggiori dettagli sul significato politico del programma, si veda Simson Garfinkel, *Architects of the Information Society: Thirty-Five Years of the Laboratory for Computer Science at MIT*, MIT Press, 1999.

[8] Si veda Richard Stallman, "RMS lecture at KTH (Svezia)", (30 ottobre 1986). http://www.gnu.org/philosophy/stallman-kth.html

[9] In una e-mail che Stallman mi ha inviato poco dopo l'ultima stesura del libro, aggiunge di aver tratto ispirazione anche dal campus di Harvard. "Nel mio primo anno ad Harvard, nel corso di Storia della Cina, studiai le vicende della prima rivolta contro la dinastia Chin", scrive. "Non si tratta di eventi storicamente attendibili, ma ne rimasi molto colpito."

[10] Si veda Richard Stallman (1986).

[11] Si veda Steven Levy, *Hackers*, Penguin USA, 1984, p. 417. Ho modificato la citazione, riportata da Levy anche come estratto parziale, per illustrare più direttamente il modo in cui il programma mettesse a nudo le false sicurezze del sistema.

[12] Ibid.

Una piccola pozzanghera di libertà

Basta interpellare chiunque abbia avuto l'opportunità di trascorrere più di un minuto in presenza di Richard Stallman, e se ne ricaverà la medesima impressione: se tralasci i capelli lunghi e qualche atteggiamento un po' strambo, la prima cosa che noti è il suo sguardo. È sufficiente un'occhiata agli occhi verdi di Stallman per rendersi conto di avere davanti qualcuno che ci crede davvero.

Definire intenso quello sguardo è dire poco. Uno sguardo penetrante, che non si accontenta di osservarti. Anche quando smetti di fissarlo per un gesto di pura cortesia, i suoi occhi rimangono invece bloccati, intenti a trapassarti il cranio come raggi fotonici.

Forse è per questo che nel descriverlo parecchi giornalisti tendono a sottolinearne la visuale religiosa. In un articolo dal titolo "Il santo del software libero", apparso nel 1998 su *Salon.com*, Andrew Leonard ne descrive gli occhi come "irradianti la forza di un profeta del Vecchio Testamento."[1] Un articolo del 1999 sul mensile *Wired* dipinge la barba di Stallman simile a "quella di Rasputin"[2], mentre un profilo del *London Guardian* ne descrive il sorriso come di "un discepolo che ha visto Gesù."[3]

Analogie che, pur avendo le loro ragioni, alla fin fine non colgono il segno. Ciò perché non considerano il lato vulnerabile dell'alter ego di Stallman. Basta infatti osservarne lo sguardo per un certo arco di tempo per notare un sottile cambiamento. Quello che inizialmente appare come un test per intimidire o ipnotizzare l'interlocutore, a una seconda o terza occhiata si rivela come il tentativo frustrato di creare e mantenere il contatto. Se, come lo stesso Stallman ha sospettato di tanto in tanto, la propria personalità è il prodotto dell'autismo o sindrome di Asperger, quello sguardo non fa che confermare

una tale diagnosi. Anche al massimo livello di intensità, i suoi occhi rivelano la tendenza a farsi offuscati e distanti, come quelli di un animale ferito che si prepara a morire.

Il mio primo incontro con il leggendario sguardo di Stallman risale al marzo 1999, in occasione della LinuxWorld Convention & Expo di San Josè, in California. Sorta di grande festa per il debutto della comunità Linux, l'evento riportò Stallman all'attenzione dei media specializzati. Determinato a farsi riconoscere i propri meriti, Stallman approfittò della manifestazione per informare il pubblico e i reporter sulla storia del progetto GNU, puntualizzandone gli obiettivi politici.

In qualità di giornalista inviato a seguire l'evento, mi vidi rimproverare apertamente da Stallman durante la conferenza stampa per il lancio di GNOME 1.0, l'interfaccia grafica libera. Senza volerlo, toccai alcune questioni assai delicate fin dalla prima domanda rivolta allo stesso Stallman: "Ritieni che la maturità di GNOME possa danneggiare la popolarità commerciale raggiunta dal sistema operativo Linux?"

"Ti prego di non chiamare più Linux quel sistema operativo", replicò Stallman, puntandomi immediatamente gli occhi addosso. "Il kernel Linux non è altro che una minima parte del sistema. Molti dei programmi che compongono quel sistema che chiami Linux non sono stati affatto sviluppati da Linus Torvalds. Li hanno invece creati quei volontari cui si deve il progetto GNU, i quali hanno messo a disposizione il proprio tempo per consentire a tutti noi l'impiego di un sistema operativo libero come l'attuale. Il mancato riconoscimento del contributo di questi volontari è una dimostrazione di scarsa educazione oltre che un'errata lettura storica. Ecco perché, riferendosi a quel sistema operativo, ti chiedo di usare il nome corretto, GNU/Linux."

Mentre prendevo nota di quelle sferzate sul mio taccuino, non potei fare a meno di notare il pesante silenzio piombato nella stanza. Quando finalmente alzai la testa, trovai ad attendermi gli occhi impassibili di Stallman. Timidamente un altro giornalista lanciò una domanda, assicurandosi di usare il termine "GNU/Linux" anziché Linux. Stavolta a replicare fu Miguel de Icaza, leader del progetto GNOME. Fu soltanto a metà della sua risposta che lo

sguardo di Stallman smise finalmente di inchiodarmi. Contemporaneamente sentii dei brividi freddi corrermi lungo la schiena. Quando Stallman iniziò una ramanzina a un altro reporter per un presunto errore lessicale, mi sentii più sollevato. Almeno voleva dire che non ce l'aveva proprio con me, pensai.

Per Stallman questi confronti faccia a faccia assumono un significato preciso. Al termine del primo LinuxWorld, la maggioranza dei giornalisti aveva imparato a non usare il termine "Linux" in sua presenza, mentre *wired.com* pubblicava un articolo in cui lo si dipingeva come un rivoluzionario pre-stalinista cancellato dai libri di storia da quegli hacker e imprenditori desiderosi di mettere in ombra gli obiettivi eccessivamente politici del progetto GNU[4]. Altri articoli vennero pubblicati, e sebbene furono pochi i giornalisti della carta stampata a usare il nome corretto "GNU/Linux", la maggior parte riconobbe a Stallman il merito di aver aperto la strada alla realizzazione di un sistema operativo libero già 15 anni prima.

Avrei rivisto Stallman solo 17 mesi dopo. Durante questo periodo, avrebbe fatto ritorno almeno una volta nella Silicon Valley, in occasione del LinuxWorld dell'agosto 1999. Pur se non invitato ufficialmente a intervenire, Stallman riuscì a distinguersi per la migliore battuta dell'evento. Accettando il Linus Torvalds Award for Community Service a nome della Free Software Foundation, dichiarò al microfono: "Offrire il Linus Torvalds Award alla Free Software Foundation è un po' come dare il Premio Han Solo all'alleanza ribelle di Guerre Stellari."

Stavolta però i commenti non trovarono spazio nei media specializzati. A metà settimana faceva il suo ingresso in borsa Red Hat, Inc., maggiore distributore di GNU/Linux. La notizia non fece che confermare quanto già intuito dai giornalisti, incluso il sottoscritto: "Linux" era diventato una parola chiave all'interno di Wall Street, come già accaduto in precedenza per "e-commerce" e "dot-com". Mentre il mercato azionario s'accostava al passaggio del nuovo millennio come un'iperbole vicina al suo asintoto verticale, ogni intervento sul software libero o sull'open source in quanto fenomeni politici finiva velocemente nel dimenticatoio.

Fu forse questo il motivo per cui, quando il LinuxWorld giunse alla sua terza edizione nell'agosto 2000, Stallman era del tutto assente.

Il mio secondo incontro con Stallman e il suo sguardo inconfondibile si svolse poco dopo la terza edizione del LinuxWorld. Avendo saputo della sua presenza a Silicon Valley, ero riuscito a organizzare un'intervista all'ora di pranzo a Palo Alto, in California. Il luogo prescelto sembrava ironico, non soltanto perché si trovava nei pressi dell'evento che Stallman aveva disertato, ma anche per via del contesto più generale. All'infuori di Redmond, poche città offrono una testimonianza maggiore del valore economico del software proprietario. Ero curioso di vedere in che modo Stallman – un uomo che ha dedicato gli anni migliori della sua vita a lottare contro la predilezione della nostra cultura per la cupidigia e l'egocentrismo – riuscisse a cavarsela in una città in cui perfino le villette più piccole costano almeno mezzo milione di dollari. Mi misi quindi al volante in direzione sud, partendo da Oakland, nella Bay Area di San Francisco.

Seguii le indicazioni fornitemi da Stallman finché non raggiunsi la sede di Art.net, un "collettivo di artisti virtuali" nonprofit. Situato in un edificio circondato da alte siepi nell'area nord della città, il quartier generale di Art.net appariva, per fortuna, alquanto modesto. Improvvisamente, l'idea di Stallman appostato nel cuore della Silicon Valley si rivelò tutt'altro che strana.

Lo trovai seduto in una stanza semibuia, intento a digitare sul suo portatile grigio. Appena fui entrato, sollevò gli occhi per trafiggermi con uno sguardo a 200 watt. Al suo semplice "Salve" replicai prontamente, ma i suoi occhi erano già tornati sul monitor del portatile.

"Sto giusto finendo un articolo sullo spirito dell'hacking", disse senza smettere di digitare. "Dai un'occhiata."

Mi avvicinai, nella stanza scarsamente illuminata. Il testo appariva in lettere verde chiaro su sfondo nero, il contrario dei colori usati nella gran parte dei programmi di elaborazione testi, per cui i miei occhi impiegarono qualche istante per adattarsi. Una volta a posto, mi trovai a scorrere il racconto di Stallman su un recente pranzo in un ristorante coreano. Prima di sedersi, fece un'interessante scoperta: sistemando il tavolo dove si stava per sedere, un

cameriere aveva lasciato sei bastoncini anziché i soliti due. Laddove chiunque altro avrebbe ignorato la cosa, Stallman la prese come una sfida: trovare il modo di usare contemporaneamente tutti e sei i bastoncini. Come nel caso dell'hacking, la soluzione si rivelò allo stesso tempo ovvia ma intelligente. Da qui la sua decisione di usarla come illustrazione grafica del pezzo.

Mentre leggevo l'articolo, mi sentii addosso il suo sguardo intenso. Notai un mezzo sorriso, con un punta d'orgoglio ma infantile, aprirsi sul suo volto. Quando gli feci i miei complimenti per il saggio, il mio commento meritò a malapena un'alzata di sopracciglia.

"Sono pronto in un attimo" rispose, riprendendo a scrivere sul suo portatile. Si trattava di un computer grigio e voluminoso, niente a che fare con gli attuali modelli sofisticati così diffusi tra i programmatori al recente LinuxWorld. Sopra la tastiera ve n'era appoggiata un'altra più leggera, a testimonianza dell'invecchiamento delle sue mani. Verso la fine degli anni '80, quando Stallman lavorava 70-80 ore a settimana sui primi programmi e strumenti software per il progetto GNU, il dolore alle mani divenne talmente insopportabile da costringerlo a ingaggiare una dattilografa. Oggi Stallman si affida a una tastiera i cui tasti richiedono una pressione assai minore rispetto a quella delle comuni tastiere.

Quando è al lavoro, Stallman ha la tendenza a bloccare qualsiasi stimolo esterno. Osservandone lo sguardo conficcato nello schermo mentre le dita danzano sulla tastiera, se ne ricava subito l'impressione di due vecchi amici immersi in una profonda conversazione.

Alcuni tasti premuti rumorosamente e la sessione era terminata. Lentamente mise via il portatile.

"Pronto per andare a pranzo?", chiese.

Raggiungemmo la mia macchina. Lamentandosi per una caviglia dolorante, Stallman camminava zoppicando. La causa era una ferita al tendine del piede sinistro risalente a tre anni prima, ma ancora così dolorosa che Stallman, fanatico delle danze popolari, aveva dovuto interrompere ogni attività di danza. "Amo molto le danze popolari", si lamentò Stallman. "Il fatto di aver dovuto smettere è stata un tragedia."

Una tragedia dalle chiare ripercussioni a livello fisico. La mancanza di esercizio lo aveva lasciato con il viso gonfio e una pancia alquanto pronunciata, entrambi assai meno visibili l'anno precedente. Era evidente come avesse messo su parecchi chili, perché quando camminava tendeva a inarcare la schiena come una donna incinta che deve adattarsi a un peso insolito.

La passeggiata venne ulteriormente rallentata perché Stallman decise di fermarsi per sentire il profumo di un cespuglio di rose. Adocchiando un'infiorescenza particolarmente bella, ne solleticò i petali interni con il suo naso prodigioso, inalando profondamente per poi indietreggiando con un sospiro deliziato.

"Mmm, *rhinophytophilia*"[5], dichiarò grattandosi la schiena.

In meno di tre minuti di macchina fummo al ristorante. Seguendo il consiglio di Tim Ney, ex-direttore esecutivo della Free Software Foundation, lasciai scegliere Stallman. Mentre qualche giornalista si divertiva a sparare a zero sul suo stile di vita monastico, la verità è che quando si trattava di mangiare, Stallman si rivelava un vero epicureo. Uno dei vantaggi collaterali di fare il missionario viaggiante per la causa del software libero, stava nella possibilità di assaggiare piatti deliziosi di ogni parte del mondo. "Prendi una della città più note, e molto probabilmente Richard ne conosce i ristoranti migliori" sostiene Ney. "Prova enorme piacere a conoscere le pietanze sul menù e ad ordinare per l'intero tavolo."

Per il pranzo di quel giorno, Stallman optò per un piccolo ristorante in stile cantonese poco distante da University Avenue, la maggiore arteria cittadina di Palo Alto. Una scelta parzialmente ispirata al suo recente viaggio in Cina ove, nella provincia di Guangdong, aveva tenuto anche una lezione, in aggiunta alla sua personale avversione per le cucine più speziate delle regioni Hunan e Szechuan. "Non sono un grande appassionato dei piatti piccanti", ammise.

Arrivammo qualche minuto dopo le 11 del mattino e fummo costretti a un'attesa di 20 minuti. Considerato il fastidio degli hacker per ogni perdita di tempo, trattenni un attimo il respiro temendo qualche scatto d'ira. Ma Stallman, contrariamente alle attese, prese la notizia con tranquillità.

“Peccato non essere riusciti a invitare nessun altro”, mi disse. “Ci si diverte sempre di più a mangiare in gruppo.”

Durante l’attesa Stallman si esercitò in alcuni passi di danza. I suoi movimenti apparvero titubanti ma ben eseguiti. Iniziammo quindi a discutere su eventi di attualità. Disse che l’unico rimpianto per non aver partecipato al LinuxWorld era aver perso la conferenza stampa per il lancio della GNOME Foundation. Nata grazie al supporto di Sun Microsystems e di IBM, la fondazione rappresentava per molti versi una rivincita dello stesso Stallman, il quale sosteneva da tempo come il software libero e l’economia del libero mercato non dovessero ritenersi reciprocamente esclusivi. Ciò nonostante, rimase deluso dal messaggio diffuso. “Visto come è stata presentata l’iniziativa, le aziende hanno parlato di Linux senza mai menzionare il progetto GNU”, disse.

Simili delusioni facevano da contrasto alle calorose risposte ricevute oltreoceano, soprattutto in Asia, aggiunse però Stallman. I suoi itinerari di viaggio del 2000 riflettevano infatti la crescente popolarità del messaggio veicolato dal software libero. Sommando le ultime visite in India, Cina e Brasile, Stallman aveva trascorso sul suolo americano appena 12 degli ultimi 115 giorni. Questi viaggi gli offrirono l’opportunità di osservare da vicino il modo in cui il software libero viene tradotto in altri linguaggi e culture.

“In India riscuote l’interesse di molta gente perché consente di costruire la propria infrastruttura informatica senza dover spendere troppo”, mi spiegò. “In Cina, l’idea si è diffusa con maggior lentezza. Accomunare il software libero alla libertà d’espressione è assai più difficile quando quest’ultima non esiste. Eppure nel corso della mia ultima visita, ho ricevuto ottimi riscontri.”

La conversazione si spostò quindi su Napster, l’azienda di San Mateo, in California, venuta alla ribalta dei media pochi mesi prima. La società commercializzava un programma controverso che consentiva agli utenti di cercare e copiare i file musicali di altri appassionati di musica. Grazie alla forza amplificatrice di Internet, tale programma definito “peer-to-peer” si è evoluto de facto in una sorta di juke-box online, offrendo a tutti la possibilità

di ascoltare file MP3 via computer senza dover pagare diritti o canoni di abbonamento, suscitando così le ire delle etichette discografiche.

Anche se era basato su software proprietario, il sistema Napster traeva ispirazione dalla posizione sostenuta a lungo da Stallman per cui, una volta che un'opera entra nel regno digitale – in altri termini, quando farne una copia diventa più una questione di duplicare informazione e meno di duplicare suoni o atomi – diviene assai difficile limitare l'impulso naturale degli esseri umani a condividere quell'opera. Anziché imporre ulteriori restrizioni, i responsabili di Napster decisero di approfittare di un tale impulso. Offrendo agli ascoltatori un luogo centralizzato destinato allo scambio di file musicali, l'azienda aveva scommesso sulla propria capacità di guidare il risultante traffico d'utenza verso altre opportunità commerciali.

L'improvviso successo del modello Napster terrorizzò l'industria discografica tradizionale, e a ragione. Appena qualche giorno prima del mio incontro con Stallman a Palo Alto, il giudice distrettuale Marilyn Patel accolse la richiesta presentata dalla Recording Industry Association of America (RIAA) approvando un'ingiunzione contro Napster, ingiunzione successivamente sospesa dalla Corte d'Appello. Ma all'inizio del 2001 quest'ultima stabilì anche l'infrazione alle norme sul copyright da parte dell'azienda di San Mateo[6], una decisione che il portavoce della RIAA avrebbe in seguito proclamato come "una netta vittoria a favore di quanti difendono i contenuti creativi nonché del mercato online legale."[7]

Per hacker come Stallman, il modello commerciale di Napster appariva controverso sotto diversi aspetti. La decisione dell'azienda di appropriarsi di principi propri del mondo hacker, quali la condivisione dei file e la proprietà comune dell'informazione, cercando al contempo di vendere un servizio basato su software proprietario, generava un segnale equivoco. Nei panni di qualcuno che deve sudare parecchio per far passare sui media il proprio messaggio attentamente studiato, Stallman appariva comprensibilmente reticente a lasciarsi andare su questo caso. Eppure ammise di aver imparato un paio di cose sull'aspetto sociale innescato dal fenomeno Napster.

“Prima di Napster, ritenevo giusta la libera distribuzione a livello privato di opere di intrattenimento”, spiegò Stallman. “Il numero di persone per cui Napster si è rivelato utile mi dice, tuttavia, che il diritto alla redistribuzione di copie, non solo nell’ambito del vicinato ma a livello di un pubblico più vasto, rimane un fatto essenziale e perciò non può essere cancellato.”

Non fece in tempo a finire la frase che si aprì la porta del ristorante e un cameriere ci invitò a entrare. In pochi secondi eravamo seduti in un tavolo d’angolo vicino a una grande parete a specchio.

Il menù del ristorante si aprì come un modulo d’ordine, mentre Stallman operava rapidamente le sue scelte prima ancora che il cameriere portasse l’acqua in tavola. “Involtini di gamberi e tofu”, lesse Stallman. “Sembra davvero interessante, credo che dovremmo provarlo.”

Il commento ci portò a una rapida escursione sul cibo cinese e la sua recente visita in quelle regioni. “In Cina la cucina è assolutamente squisita”, disse Stallman, la cui voce per la prima volta nella mattinata lasciava trasparire emozione. “Tantissimi piatti diversi che non ho mai visto negli Stati Uniti, ingredienti locali fatti con funghi e vegetali tipici di una certa zona. Ero arrivato al punto di tenere un diario soltanto per conservare il ricordo di quei piatti eccezionali.” La conversazione proseguì affrontando la cucina coreana. Nel corso del suo viaggio in Asia nel giugno 2000 Stallman visitò anche la Corea del Sud. Il suo arrivo scatenò una piccola tempesta tra i media locali grazie a una conferenza sul software, prevista per quella stessa settimana, cui prese parte Bill Gates, fondatore e responsabile della Microsoft. Dopo aver visto la propria foto sopra quella di Gates sulla prima pagina del maggiore quotidiano di Seul, la cosa migliore di quel viaggio fu il cibo, ricordò Stallman. “Una volta assaggiai una tazza di *naeng myun*, una sorta di pasta in brodo fredda”, iniziò a spiegare. “Aveva un sapore veramente intrigante. Generalmente i locali non usano lo stesso tipo di pasta per quel piatto, perciò posso affermare con piena cognizione di causa che si trattò del *naeng myun* più squisito che avessi mai provato.”

Il termine “squisito” rappresentava un grosso complimento da parte di Stallman. Me ne resi conto perché, qualche istante dopo aver tessuto le lodi

del *naeng myun*, sentii il suo sguardo sfiorare la mia spalla destra.

"Appena dietro di te siede una donna davvero squisita", bisbigliò Stallman.

Mi girai dando un'occhiata alle spalle della donna. Era giovane, sui venticinque anni, e indossava un vestito bianco con lustrini. Era insieme a un uomo e si apprestavano a pagare il conto. Quando si alzarono per lasciare il ristorante me ne accorsi senza voltarmi perché improvvisamente la luce negli occhi di Stallman diminuì d'intensità. "Oh, no", disse. "Vanno via. E pensare che probabilmente non mi capiterà di rivederla mai più."

Dopo un breve sospiro, Stallman si riprese. Il momento mi offriva l'opportunità per spostare la discussione sulla sua reputazione nei rapporti faccia a faccia con il gentil sesso. Una reputazione a tratti contraddittoria. Qualche hacker segnalava la predilezione di Stallman nel salutare ogni donna baciandole la mano[8]. Un articolo apparso il 26 maggio 2000 su *Salon.com*, invece, lo descrisse come un hacker libertino. Operando l'associazione software libero-amore libero, la giornalista Annalee Newitz presentò uno Stallman che rifiutava i tradizionali valori familiari, con frasi tipo: "Credo nell'amore, ma non nella monogamia."[9]

Stallman abbassò leggermente il menù non appena sollevai la questione. "Bé, la maggior parte degli uomini sembra volere il sesso eppure tratta le donne in maniera alquanto sprezzante", disse. "Anche quelle con cui hanno una relazione. Non riesco a comprenderne il motivo."

Citai un passaggio dal libro del 1999 *Open Sources* in cui egli confessava di aver pensato di chiamare il kernel GNU mai realizzato con il nome della sua ragazza di allora, Alix, nome che si adattava perfettamente alla convenzione degli sviluppatori di mettere una "x" alla fine del nome di ogni nuovo kernel – è il caso di "Linux." Dato che quella ragazza era amministratore di un sistema Unix, Stallman precisò come la scelta si sarebbe rivelata un tributo ancor più significativo. Purtroppo, aggiunge, lo sviluppatore che alla fine prese il mano il progetto optò per il nome HURD[10].

Anche se in seguito quella relazione s'interruppe, la vicenda suggeriva una domanda: mentre l'immaginario dei media lo dipingeva come un fanatico dallo sguardo stralunato, non era invece Richard Stallman un inguaribile

romantico, un Don Chisciotte vagabondo che lottava contro i mulini a vento delle grandi corporation nel tentativo di incantare qualche Dulcinea ancora da identificare?

“In realtà non stavo cercando di essere romantico”, replicò Stallman sull’episodio di Alix. “Era anche un modo per punzecchiarla. Insomma, romantico sì, ma anche provocatorio. Sarebbe stata una piacevole sorpresa.”

Per la prima volta quella mattina Stallman rise apertamente. Gli ricordai la faccenda del baciamano. “Si, mi piace farlo”, replicò. “Trovo che sia una dimostrazione di simpatia piacevole per molte donne. Un modo per esprimere amicizia ed essere anche apprezzati.”

Il sentimento di amicizia è sempre stato un tema importante nella vita di Richard Stallman, e su questo punto rispose con dolorosa sincerità: “Non è che ne abbia ricevuta granché in vita mia, tranne forse nella mia mente”. La discussione stava velocemente prendendo una strana piega. Dopo qualche replica a monosillabi, Stallman sollevò il menù, bloccando ogni ulteriore domanda.

“Che ne dici di qualche *shimai*?”, mi chiese.

Quando iniziarono a portarci da mangiare, la conversazione si mosse come uno slalom tra le varie portate. Parlammo dell’attrazione, così spesso evidenziata, degli hacker per la cucina cinese; delle incursioni settimanali nel quartiere di Chinatown, a Boston, negli anni in cui lavorava al laboratorio di intelligenza artificiale; della logica alla base della lingua cinese e del relativo sistema di scrittura. Ogni mia battuta stimolava una replica assai informata da parte di Stallman.

“L’ultima volta che sono stato in Cina ho sentito qualcuno parlare la lingua di Shangai”, raccontò Stallman. “È interessante da ascoltare. Tutta un’altra sonorità [dal mandarino]. Mi son fatto dire alcuni termini analoghi in mandarino e shanghainese. In qualche caso la somiglianza è evidente, ma mi chiedevo se l’intonazione della voce sarebbe stata simile. Non lo è. Mi interessava scoprirlo perché secondo una certa teoria tale intonazione deriva dalle sillabe aggiuntive andate perdute e poi sostituite. Il loro effetto sopravvive nel tono della voce. Se ciò è vero, e ho studiato le ipotesi che lo

confermano in determinate epoche storiche, i dialetti devono essersi sviluppati prima della perdita di queste sillabe finali."

Arrivò la prima portata, un piatto di tortine di rape fritte. Ci dedicammo entrambi a quelle torte rettangolari, che avevano l'odore di cavolo bollito e il sapore di patate fritte nella pancetta.

Decisi di tornare nuovamente sulla questione del suo isolamento giovanile, chiedendogli se gli anni dell'adolescenza lo avessero in qualche modo condizionato al punto tale da assumere in seguito posizioni impopolari, in particolare la difficile battaglia avviata fin dal 1994 per convincere utenti e media a sostituire l'ormai diffuso termine "Linux" con "GNU/Linux."

"Credo mi abbia aiutato", replicò Stallman continuando a masticare. "Non ho mai compreso appieno le implicazioni della pressione dei coetanei su qualcuno. Credo il motivo fosse che venivo respinto senza speranza, così non avevo nulla da guadagnare cercando di seguire le mode del momento. Nel mio caso non avrebbe portato ad alcuna differenza. Mi avrebbero rifiutato ugualmente, così non ci provai neppure."

A riprova delle sue tendenze anticonformiste, Stallman citò i propri gusti in fatto di musica. Da adolescente, quando la gran parte dei compagni delle medie ascoltava la Motown e l'acid rock, preferiva la musica classica. Il ricordo portò a uno dei rari episodi divertenti di quegli anni. Nel 1964, subito dopo l'apparizione dei Beatles all'Ed Sullivan Show in televisione, quasi tutti i suoi compagni di classe corsero ad acquistare i loro ultimi album e singoli. Fu proprio in quel momento che, disse Stallman, egli prese la decisione di boicottare i "Fab Four".

"Mi piaceva qualche pezzo popolare dell'era pre-Beatles", rammentò Stallman. "Ma loro non m'interessavano. Mi dava particolarmente fastidio il modo in cui la gente reagiva a quel fenomeno. Era come dire: chi parteciperà all'assemblea sui Beatles per imparare come adularli al meglio?"

Quando la sua proposta di boicotaggio dei Beatles non ottenne seguito, Stallman cercò altre soluzioni per evidenziare la mentalità da gregge dei coetanei. Raccontò di aver considerato brevemente di mettere insieme un

gruppo rock apposta per prendere in giro i quattro di Liverpool. “Avrei voluto chiamarlo Tokyo Rose & Japanese Beetles.”

Data la sua passione per la musica folk internazionale, gli chiesi se avesse mai avuto un’analoga affinità per Bob Dylan e gli altri musicisti folk dei primi anni ’60. Scosse la testa. “Mi piacevano Peter, Paul & Mary”, replicò. “Mi ricordavano alcune grandi parodie musicali[11].”

Quando gli chiesi maggiori dettagli, Stallman mi spiegò che si trattava di canzoni popolari le cui parole originali venivano sostituite da strofe ironiche. Un simile processo di riscrittura è tuttora attività diffusa tra gli hacker e gli appassionati di fantascienza. Tra i grandi classici ci sono “On Top of Spaghetti”, versione rivista di “On Top of Old Smokey”[12], e il capolavoro di “Weird Al” Yankovic “Yoda”, versione sul tema Guerre Stellari di “Lola”, dei Kinks[13].

Stallman mi chiese poi se desiderassi ascoltare una canzone di quel tipo. Al mio cenno positivo, iniziò a cantare, con una voce inaspettatamente chiara:

How much wood could a woodchuck chuck,
If a woodchuck could chuck wood?
How many poles could a polak lock,
If a polak could lock poles?
How many knees could a negro grow,
If a negro could grow knees?
The answer, my dear, is stick it in your ear.
The answer is to stick it in your ear.[14]

Alla fine del pezzo, le labbra di Stallman erano piegate in un mezzo sorriso da bambino. Diedi un’occhiata ai tavoli intorno. Le famigliole asiatiche intente a gustare il pranzo domenicale stavano prestando ben poca attenzione al contralto barbuto lì vicino[15]. Dopo qualche attimo di esitazione, anch’io mi lasciai andare in un sorriso.

“Vuoi quell’ultima di polpetta di mais?”, chiese con gli occhi che gli brillavano. Prima ancora che potessi replicare aveva già afferrato la polpetta con i bastoncini e la teneva sollevata con orgoglio. “Forse me la merito” concluse.

Finite le pietanze, la conversazione assunse le dinamiche di una normale intervista. Stallman si distese sulla sedia cullando una tazza di tè tra le mani. Riprendemmo a parlare di Napster e delle implicazioni per il movimento del software libero. Gli chiesi se i principi alla base di quest'ultimo avrebbero forse dovuto estendersi ad ambiti limitrofi quali la pubblicazione di opere musicali.

"È un errore trasferire le risposte da un contesto all'altro", spiegò Stallman, contrapponendo le canzoni con il software. "L'approccio corretto consiste nel considerare singolarmente ogni tipo di opera e vedere quali conclusioni è possibile raggiungere in quell'ambito."

Secondo Stallman le opere sotto copyright vanno divise in tre diverse categorie. La prima include quelle "funzionali", ovvero programmi informatici, dizionari, libri di testo. La seconda è per le opere definite "testimoniali", cioè relazioni scientifiche e documenti storici. Si tratta di lavori il cui senso verrebbe stravolto qualora lettori e altri autori potessero modificarli a piacimento. L'ultima categoria riguarda le opere di espressione personale, quali diari, resoconti, autobiografie. Modificare questi documenti significherebbe alterarne i ricordi o il punto di vista – azione eticamente ingiustificabile nell'opinione di Stallman.

Delle tre categorie, la prima dovrebbe garantire agli utenti il diritto illimitato alla creazione di versioni modificate, mentre per la seconda e la terza tale diritto andrebbe regolamentato a seconda della volontà dell'autore originale. Indipendentemente dalla categoria di appartenenza, dovrebbe tuttavia rimanere inalterata la libertà di copia e redistribuzione a livello non commerciale, insiste Stallman. Se ciò significa consentire agli utenti Internet di generare un centinaio di copie di un articolo, un'immagine, una canzone o un libro per poi inviarle via email a un centinaio di sconosciuti, ebbene così sia. "È chiaro che la redistribuzione occasionale in ambito privato debba essere permessa, perché soltanto uno stato di polizia potrebbe impedirlo", aggiunge. "È pratica antisociale frapporsi tra una persona e i propri amici. Napster mi ha convinto della necessità di permettere perfino la redistribuzione

non-commerciale al pubblico più ampio anche solo per il divertimento. Perché c'è così tanta gente che la considera un'attività utile."

Quando gli chiesi se i tribunali sarebbero disposti ad accettare uno scenario talmente permissivo, Stallman m'interruppe bruscamente.

"Ecco una domanda sbagliata", replicò. "Hai spostato completamente l'argomento, passando da una questione etica all'interpretazione legislativa. Si tratta di due aspetti totalmente diversi pur nello stesso ambito. È inutile saltare da uno all'altro. I tribunali continueranno a interpretare le norme attuali per lo più in maniera restrittiva, poiché è così che tali norme sono state imposte dagli editori."

Il commento portò a una riflessione sulla filosofia politica di Stallman: solo perché attualmente il sistema legale permette agli imprenditori di trattare il copyright come l'equivalente informatico di un contratto di proprietà terriera, ciò non significa che gli utenti debbano necessariamente conformarsi a queste regole. La libertà è una faccenda etica, non legale. "Guardo al di là delle norme esistenti per considerare come invece dovrebbero essere", spiegò Stallman. "Non sto cercando di stilare una legislazione. Mi sto forse occupando delle applicazioni della legge? Piuttosto, considero le norme che vietano la condivisione di copie col vicino come l'equivalente morale di Jim Crow[16]. Qualcosa che non merita alcun rispetto."

Il richiamo a quelle politiche portò alla domanda successiva. Quanta influenza o ispirazione avevano per Stallman i leader politici del passato? Come il movimento per i diritti civili degli anni '50 e '60, il suo tentativo di guidare il cambiamento sociale poggiava su valori inalterati nel tempo: libertà, giustizia, onestà.

Stallman divideva la sua attenzione tra la mia analogia e una ciocca di capelli particolarmente annodata. Quando allargai la questione fino al punto di paragonarlo a Martin Luther King, Jr., dopo aver dato uno strattone a una doppia punta ed essersela infilata in bocca, Stallman m'interruppe prontamente.

"Non faccio parte di quella squadra, ma gioco la stessa partita" replicò masticando.

Suggerii un altro confronto, quello con Malcolm X. Come nel caso dell'ex-portavoce della Nazione dell'Islam, Stallman si era guadagnato la reputazione di voler creare controversie, alienarsi potenziali alleati e predicare l'autosufficienza contro ogni integrazione culturale.

Masticando un'altra doppia punta, rifiutò anche questo paragone. "Il mio messaggio è più vicino a quello di King", rispose. "Si tratta di un messaggio universale, di una decisa condanna di certe pratiche ingiuste nei confronti degli altri. Non è un messaggio di odio contro chicchessia. Neppure è diretto a un gruppo ristretto di individui. È un invito rivolto a chiunque ad apprezzare e a sperimentare la libertà."

Comunque sia, il sospetto nei confronti di ogni alleanza politica rimane un tratto fondamentale del personaggio Stallman. Considerato il disgusto, ampiamente pubblicizzato, per il termine "open source", diventa comprensibile il rifiuto di prendere parte ai recenti progetti mirati alla costruzione di possibili coalizioni. Nel ruolo di qualcuno che ha passato gli ultimi vent'anni a far sentire la sua voce per sostenere il software libero, il capitale politico di Stallman era profondamente intessuto con quest'ultimo termine. Eppure commenti come quello sul Premio Han Solo all'Alleanza Ribelle di Guerre Stellari del LinuxWorld 1999, non fecero che rinforzare la reputazione, acquisita nell'industria del software, di un conservatore scontroso contrario a scendere a patti con le nuove tendenze della politica o del mercato.

"Ammiro e rispetto Richard per tutto il lavoro svolto finora", afferma Robert Young, presidente di Red Hat, sintetizzando la paradossale natura politica di Stallman. "La mia sola critica è che talvolta finisce per trattare gli amici peggio di quanto faccia con i nemici."

L'opposizione di Stallman alla ricerca di alleanze suscita analoghe perplessità quando si considerano i suoi interessi politici al di fuori del movimento del software libero. Visitando il suo ufficio al MIT, ci si imbatte in una copiosa raccolta di articoli sinistroidi sugli abusi dei diritti civili commessi in ogni paese del mondo. Mentre il sito web personale include discussioni su temi quali il Digital Millennium Copyright Act, la War on Drugs e la World Trade Organization.

Considerate le sue tendenze verso l'attivismo diretto, gli chiesi, perché mai non aveva cercato di dar maggior spazio alla propria voce? Perché non ha usato la visibilità guadagnata nel mondo hacker come piattaforma per amplificare, anziché ridimensionare, il proprio credo politico?

Stallman lasciò andare i cappelli annodati e rifletté per un istante sulla domanda.

"Sì, ho esitato ad ampliare l'importanza di questa piccola pozzanghera di libertà", rispose. "Perché le aree convenzionali e collaudate in cui si lavora per la libertà e per una società migliore sono tremendamente importanti. Non ritengo ugualmente vitale la battaglia a sostegno del software libero. Questa è la responsabilità che mi sono assunto, perché mi è caduta in grembo e ho capito che avrei potuto portare contributi positivi. Ma, ad esempio, porre fine alla brutalità della polizia, alla guerra per droga, alle forme di razzismo tuttora presenti, oppure aiutare qualcuno a vivere meglio, a tutelare il diritto delle donne ad abortire, a proteggerci dalla teocrazia, si tratta di questioni di enorme importanza, ben superiori a ogni mio possibile intervento. Non saprei come offrire contributi efficaci in quei contesti."

Ancora una volta Stallman presentava la propria attività politica come dipendente dalla fiducia in sé stessi. Considerando il tempo che aveva impiegato per sviluppare e affinare i principi cardine del movimento del software libero, appariva esitante a coinvolgersi in ogni tematica o tendenza che potesse trascinarlo in territori inesplorati.

"Vorrei essere in grado di contribuire in maniera significativa alle questioni di maggiore rilevanza, ne sarei tremendamente orgoglioso, ma si tratta di problemi davvero grossi e quanti se ne occupano, ben più in gamba del sottoscritto, non sono ancora riusciti a risolvere granché", aggiunse. "Eppure, per come la vedo io, mentre altre persone ci difendevano contro queste grandi minacce ben visibili, io ne ho individuata un'altra rimasta sguarnita. E così ho deciso di combatterla. Forse non è un minaccia altrettanto grande, ma io sono stato l'unico a farmi avanti."

Masticando un'ultima punta di capelli, Stallman suggerì di pagare il conto. Prima che il cameriere potesse prendere il denaro, però, Stallman aggiunse

una banconota di colore bianco. Appariva così chiaramente falsa che non potei fare a meno di prenderla per osservarla da vicino. Era palesemente contraffatta. Al posto dell'immagine di George Washington o Abramo Lincoln, un lato della banconota presentava la caricatura di un maiale. Invece della scritta United States of America in alto, sopra il maiale si leggeva "United Swines of Avarice" (suini uniti dell'avarizia). La banconota era di zero dollari, e quando il cameriere venne a prendere il denaro, Stallman lo afferrò per la manica. "Ho aggiunto uno zero extra alla mancia", gli disse accennando un mezzo sorriso. Il cameriere, che forse non aveva capito o era rimasto interdetto dall'aspetto della banconota, sorrise e corse via.

"Credo ciò significhi che siamo liberi di andarcene", concluse Stallman.

[1] Si veda Andrew Leonard, "The Saint of Free Software", *Salon.com*, agosto 1998. http://www.salon.com/21st/feature/1998/08/cov_31feature.html

[2] Si veda Leander Kahney, "Linux's Forgotten Man", *Wired News*, 5 marzo 1999. http://www.wired.com/news/print/0,1294,18291,00.html

[3] Si veda "Programmer on moral high ground; Free software is a moral issue for Richard Stallman believes in freedom and free software", *London Guardian*, 6 novembre 1999. Queste sono soltanto alcuni dei paragoni religiosi in circolazione. Fino ad oggi la palma del confronto più estremo spetta a Linus Torvalds, il quale, nella sua autobiografia – si veda Linus Torvalds & David Diamond, *Rivoluzionario per caso*, Garzanti, 2001 – scrive: "Richard Stallman è il Dio del Software Libero." Una menzione d'onore spetta a Larry Lessig, il quale in una nota a descrizione di Stallman inclusa nella sua opera – si veda Larry Lessig, *The Future of Ideas*, Random House, 2001, p. 270 – lo paragona a Mosé: "[…] come Mosé, toccò a un altro leader, Linus Torvalds, guidare finalmente il movimento alla terra promessa, facilitando lo sviluppo della parte finale del sistema operativo. Come Mosé, anche Stallman viene contemporaneamente rispettato e vilipeso dagli stessi alleati all'interno del movimento. Si tratta di un leader implacabile, e quindi d'ispirazione per molti, in un ambito criticamente importante per la cultura moderna. Nutro un profondo rispetto per i principi e l'abnegazione di questo straordinario individuo, sebbene abbia grande rispetto anche per coloro talmente coraggiosi da metterne in dubbio il pensiero e poi sostenerne l'ira funesta."
Durante un'intervista conclusiva con Stallman, gli chiesi la sua opinione su quei raffronti religiosi. "Qualcuno mi paragona a uno dei profeti del Vecchio Testamento, e ciò perché allora costoro andavano sostenendo l'ingiustizia di certe pratiche sociali. Sulle questioni morali non scendevano a compromessi. Non potevano essere comprati e generalmente venivano trattati con disprezzo."

[4] Si veda Leander Kahney, "Linux's Forgotten Man" *Wired News*, 5 marzo 1999. http://www.wired.com/news/print/0,1294,18291,00.html

[5] In quel momento mi sembrò che Stallman citasse il termine scientifico della pianta. Qualche mese dopo avrei scoperto invece che rhinophytophilia era in realtà un riferimento umoristico all'attività in corso, ovvero lo stesso Stallman mentre affondava il naso nel fiore e ne godeva. Per un altro incidente divertente in tema floreale, si veda: http://www.stallman.org/texas.html

[6] Si veda Cecily Barnes & Scott Ard, "Court Grants Stay of Napster Injunction" *News.com*, 28 luglio 2000. http://news.cnet.com/news/0-1005-200-2376465.html

[7] Si veda "A Clear Victory for Recording Industry in Napster Case", comunicato stampa della RIAA (12 febbraio 2001). http://www.riaa.com/PR_story.cfm?id=372

[8] Si veda Mae Ling Mak, "Mae Ling's Story" (17 dicembre 1998). http://www.crackmonkey.org/pipermail/crackmonkey/1998q4/003006.htm. Finora Mak si è dimostrata l'unica persona disposta a parlare apertamente di questa pratica, anche se ne avevo già avuto notizia da altre fonti femminili. Superata una certa repulsione iniziale, in seguito Mak riuscì a mettere da parte ogni preconcetto per cimentarsi in una danza con Stallman durante il LinuxWorld del 1999. http://www.linux.com/interact/potd.phtml?potd_id=44

[9] Si veda Annalee Newitz, "If Code is Free Why Not Me?" *Salon.com*, 26 maggio 2000. http://www.salon.com/tech/feature/2000/05/26/free_love/print.html

[10] Si veda Richard Stallman, "Il progetto GNU" in *Open Sources, Voci dalla rivoluzione Open Source*, Apogeo, 1999, p. 65.

[11] "Filk" nell'originale inglese. Si tratta di un neologismo nato da un errore tipografico sul termine "folk", entrato nell'uso comune per indicare le versioni umoristiche delle canzoni popolari [*N.d.R.*]

[12] Canzone popolare incisa, tra gli altri, anche da Harry Belafonte, "Scarlet Ribbons", Camden. Il testo della versione "filk" con accompagnamento MIDI è disponibile online all'indirizzo http://www.scoutsongs.com/lyrics/ontopofspaghetti.html [*N.d.R.*]

[13] Vedi http://www.com-www.com/weirdal/ per il testo di "Yoda" [*N.d.R.*]

[14] Parodia di "Blowing in the Wind", di Bob Dylan [*N.d.R.*]

[15] Per saperne di più su analoghe parodie di canzoni popolari di Stallman, si veda: http://www.stallman.org/doggerel.html. Per ascoltare Stallman cantare "The Free Software Song": http://www.gnu.org/music/free-software-song.html.

[16] Le politiche di segregazione razziale applicate nel sud degli Stati Uniti all'inizio del XX secolo. [*N.d.T.*]

La comune dell’Emacs

Il laboratorio di intelligenza artificiale degli anni ’70 era un luogo speciale sotto tutti gli aspetti. Il connubio tra progetti d’avanguardia e ricercatori di livello insuperabile, gli conferivano una posizione di primo piano nel mondo dell’informatica. La cultura hacker vigente al suo interno con le sue politiche anarchiche, gli conferiva un’aurea ribelle. Soltanto più avanti, con l’addio dei migliori ricercatori e delle superstar del software, gli hacker avrebbero compreso quanto fosse unico ed effimero il mondo in cui avevano vissuto un tempo.

“Era una sorta di paradiso terrestre”, così Stallman ricorda il laboratorio e l’ethos della condivisione del software, in un articolo del 1998 per la rivista *Forbes*. “Non ci veniva neppure in mente di non collaborare tra noi”[1].

Queste descrizioni mitologiche, per quanto eccessive, sottolineano un fatto importante. Erano in parecchi a ritenere il nono piano dello stabile al 545 di Tech Square come qualcosa di più di un luogo di lavoro. Hacker come Stallman lo consideravano la loro casa.

Nel lessico di quest’ultimo, il termine “casa” riveste un significato del tutto particolare. Con una nota polemica verso i propri genitori, ancora oggi rifiuta di riconoscere come casa propria nessun’altra abitazione precedente alla Currier House, il dormitorio in cui viveva ai tempi di Harvard. Luogo che tra l’altro dovette abbandonare in maniera tragicomica, almeno stando alla sua stessa descrizione. Una volta, parlando dei suoi anni ad Harvard, Stallman disse che il suo unico rimpianto fu di essere stato sbattuto fuori. Non appena gli chiesi cosa avesse provocato quell’episodio, mi resi conto di aver toccato un altro mito della sua vita.

“Ad Harward hanno questa politica: se passi troppi esami, ti chiedono di andartene”, mi disse.

Lasciato il dormitorio e senz’alcuna voglia di tornarsene a New York, Stallman seguì la strada già percorsa da Greenblatt, Gosper, Sussman e numerosi altri hacker prima di lui. Dopo essersi iscritto al MIT e aver preso in affitto un appartamento nella vicina Cambridge, ben presto Stallman si rese conto di come il laboratorio di intelligenza artificiale fosse divenuto di fatto la sua dimora. In un intervento del 1986, così ricorda quel periodo:

> È probabile che io abbia vissuto al laboratorio più di altri, perché quasi ogni anno per un motivo o per l’altro dovevo trovarmi un nuovo appartamento, e nell’attesa finivo per trascorrere lì qualche mese. Mi ci sono trovato sempre bene, in estate era proprio fresco. Capitava di frequente che qualcuno vi si addormentasse, altro effetto dell’entusiasmo; stai alzato davanti al monitor fino allo stremo delle forze perché non vuoi smettere, e quando sei completamente esausto non fai altro che distenderti sulla superficie morbida e orizzontale più vicina. Un’atmosfera davvero informale[2].

Talvolta questa atmosfera casalinga creava dei problemi. Quello che alcuni consideravano alla stregua di dormitorio, per altri non era altro che una specie di ritrovo per “oppiomani elettronici.” Nel libro *Computer Power and Human Reason*, del 1976, Joseph Weizenbaum, ricercatore presso il MIT, non risparmia pesanti critiche ai “vagabondi informatici”, nella sua definizione degli hacker che popolavano quei locali. “Vestiti trasandati, capelli sporchi e spettinati, barba lunga, testimoniavano la loro indifferenza al corpo e al mondo circostante” scrive Weizenbaum. “[I vagabondi informatici] vivono, almeno relativamente a questo contesto, soltanto grazie e per il computer.”[3]

Un quarto di secolo dopo la pubblicazione di quel libro, Stallman rispose alla descrizione dei “vagabondi informatici” di Weizenbaum come se quest’ultimo gli fosse stato di fronte in quel preciso istante. “Vuole che tutti siano dei professionisti, gente che lavora per i soldi e che appena possibile dimentica e abbandona i progetti di cui si occupa”, spiega Stallman. “Quello che a lui appare come uno scenario comune, a me sembra una vera tragedia.”

E tuttavia, anche la vita dell’hacker non era priva di tali tragedie. Stallman descrive la sua transizione da hacker del fine settimana a ricercatore a tempo pieno nel laboratorio di intelligenza artificiale come una serie di dolorose

sventure, che riuscì a superare soltanto grazie all'euforia dell'hacking. Nel suo racconto, il primo di questi episodi risale all'epoca della laurea ad Harvard. Convinto di voler proseguire gli studi in fisica, Stallman si iscrisse subito al MIT. Una scelta naturale che non soltanto gli permetteva di seguire le orme di alcuni grandi studenti del MIT come William Shockley (1936), Richard P. Feynman (1939) e Murray Gell-Mann (1951), ma gli consentiva anche di vivere a meno di tre chilometri dal laboratorio di intelligenza artificiale e dal nuovo computer PDP-10. "Ero attratto dalla programmazione, ma pensavo ancora che avrei potuto coltivare entrambe le cose", ricorda Stallman.

Di giorno a seguire i corsi di laurea e la notte a programmare nei monastici confini del laboratorio di intelligenza artificiale, Stallman era alla ricerca del perfetto equilibrio. Il fulcro di questa frenetica attività stava però nelle uscite settimanali con il gruppo di danze popolari, evento sociale che gli garantiva quantomeno un modico livello di interazione con l'altro sesso. Fu sul finire del primo anno al MIT, tuttavia, che avvenne il disastro. Una ferita al ginocchio lo costrinse a lasciare il gruppo. All'inizio lo considerò un problema temporaneo e dedicò al laboratorio d'intelligenza artificiale l'ulteriore tempo libero distolto alla danza. Ma al termine dell'estate iniziò a preoccuparsi seriamente, col ginocchio ancora dolorante e l'approssimarsi dei nuovi corsi. "Il ginocchio non ne voleva sapere di migliorare", ricorda Stallman, "con la conseguenza che dovetti smettere definitivamente di danzare. Fu un colpo mortale."

Senza né dormitorio né danze, il suo universo sociale finì per implodere. Al pari di un astronauta alle prese con i postumi dell'assenza di gravità, si rese conto di come la sua capacità di interagire con i nonhacker, soprattutto di genere femminile, si fosse atrofizzata in maniera considerevole. Dopo 16 settimane trascorse al laboratorio di intelligenza artificiale, quella fiducia in se stesso lentamente accumulata durante i quattro anni ad Harvard era praticamente scomparsa.

"In pratica mi sentivo svuotato di ogni energia", rammenta Stallman. "Non avevo voglia di far nulla se non quel che mi attirava al momento, non avevo energia per altro. Ero disperato."

Finì presto per ritirarsi ulteriormente dal mondo, dedicandosi unicamente al lavoro nel laboratorio di intelligenza artificiale. Nell'ottobre 1975, abbandonò i corsi al MIT, per non farvi mai più ritorno. Da semplice hobby iniziale, l'attività di hacking era diventata la sua vocazione.

Riflettendo su quel periodo, Stallman considera inevitabile il passaggio da studente a tempo pieno ad hacker a tempo pieno. Prima o poi, così ritiene, quest'ultimo richiamo avrebbe superato ogni altro interesse a livello professionale. "In materie come fisica o matematica, non riuscivo mai a proporre contributi originali", sostiene Stallman riferendosi alle vicissitudini precedenti la ferita al ginocchio. "Sarei stato orgoglioso di fornire contributi allo sviluppo di uno dei due campi, ma non vedevo come avrei potuto farlo. Non sapevo da che parte cominciare. Col software, invece, sapevo già come scrivere cose funzionanti e utili. Il piacere di questa conoscenza mi portò a desiderare di proseguire su questa strada."

Stallman non era certo il primo ad equiparare hacking e piacere. Gran parte di coloro che lavoravano al laboratorio di intelligenza artificiale vantavano un curriculum accademico analogamente incompleto. Molti di loro si erano iscritti a corsi di laurea in matematica o in ingegneria elettrica soltanto per poi abbandonare le proprie ambizioni professionali per la sottile soddisfazione che si ottiene risolvendo problemi mai affrontati prima. Come San Tommaso d'Aquino, noto per aver lavorato talmente a lungo sulle questioni teologiche da avere talvolta delle visioni spirituali, gli hacker raggiungevano stati di trascendenza interiore grazie alla concentrazione mentale e alla spossatezza fisica. Come la maggior parte di loro, Stallman rifiutava l'uso di droghe ma era attirato dallo stato di ebbrezza che si prova dopo 20 ore trascorse a scrivere codice.

Forse l'emozione più piacevole era il senso di appagamento personale. Per Stallman l'arte dell'hacking era qualcosa di naturale. Le lunghe notti passate a studiare quando era bambino, gli permettevano di lavorare a lungo, con poche ore di sonno. Asociale e individualista fin da quando aveva dieci anni, non aveva alcun problema a lavorare in completa solitudine. E in quanto matematico eccellente con capacità innate per la logica e un occhio critico,

Stallman riusciva a superare quelle barriere progettuali che facevano dannare gran parte degli hacker.

“Era davvero speciale”, ricorda Gerald Sussman, membro di facoltà al MIT ed ex ricercatore presso il laboratorio di intelligenza artificiale. Descrivendolo come “pensatore e progettista acuto”, fu Sussman ad assumere Stallman come assistente per un progetto di ricerca a partire dal 1975. Si trattava di un progetto complesso, che richiedeva la creazione di un programma di IA in grado di analizzare i diagrammi dei circuiti. Qualcosa che richiedeva non soltanto un esperto di Lisp, un linguaggio di programmazione realizzato appositamente per applicazioni nel campo dell’intelligenza artificiale, ma anche la piena comprensione di come un essere umano avrebbe affrontato lo stesso problema.

Quando non lavorava a progetti ufficiali come quello di Sussman sul programma automatico di analisi dei circuiti, Stallman si dedicava a progetti personali. Era nel miglior interesse di ciascun hacker contribuire al miglioramento dell’infrastruttura del laboratorio, e uno dei suoi progetti più importanti di quel periodo riguardava l’editor TECO.

Negli anni ’70 il suo impegno in tale progetto appare inestricabilmente connesso alla successiva leadership del movimento del software libero, oltre a rappresentare un passaggio significativo nella storia dello sviluppo dell’informatica, motivo per cui val la pena riassumerlo qui. Nel corso degli anni ’50 e ’60, quando il computer fece la sua prima apparizione nelle aule universitarie, la programmazione costituiva qualcosa di puramente astratto. Per comunicare con la macchina, gli sviluppatori creavano una serie di schede perforate, ognuna delle quali rappresentava un singolo comando software. Tali schede venivano poi passate all’amministratore centrale del sistema, il quale le avrebbe a sua volta introdotte una ad una nella macchina, aspettando che questa ne producesse in uscita una nuova serie, che andava infine decifrata come output dai programmatori. Questa procedura, nota come “elaborazione batch”, era tanto noiosa ed estenuante quanto soggetta ad abusi di autorità. Uno dei fattori a monte dell’innata avversione degli hacker per la

centralizzazione, stava nel potere conferito ai primi amministratori di sistema cui spettava decidere quali lavori avessero priorità più alta.

Nel 1962 i ricercatori informatici e gli hacker riuniti sotto il progetto MAC del MIT, precursore del laboratorio di intelligenza artificiale, presero a darsi da fare per alleviare quel senso di frustrazione. Il “time sharing” (condivisione di tempo) all’inizio detto “time stealing” (furto di tempo), aveva reso possibile a più programmatori l’uso contemporaneo delle capacità operative di una macchina. L’interfaccia a telescrivente consentiva inoltre la comunicazione non più tramite una serie di fori in una scheda, bensì grazie all’interazione testuale. Il programmatore digitava dei comandi e leggeva poi, riga per riga, il risultato prodotto dalla macchina.

Sul finire degli anni ’60, lo studio dell’interfaccia compì altri passi in avanti. Nel 1968 in un famoso intervento Doug Engelbart, allora ricercatore presso lo Stanford Research Institute, illustrò il prototipo di quella che sarebbe diventata l’interfaccia grafica moderna. Con l’aggiunta di un monitor televisivo e di un puntatore chiamato “mouse”, Engelbart mise a punto un sistema ancora più interattivo di quello a condivisione di tempo sviluppato al MIT. Trattando il monitor come una stampante ad alta velocità, tale sistema offriva la possibilità di seguire gli spostamenti del cursore sul video in tempo reale. Tutt’a un tratto l’utente era in grado di posizionare il testo in una parte qualunque dello schermo.

Queste innovazioni non sarebbero arrivate sul mercato prima di altri vent’anni, ma già dagli anni ’70 i monitor stavano iniziando a sostituire le telescriventi come terminali di visualizzazione, dando la possibilità di lavorare a tutto schermo anziché riga per riga.

Uno dei primi programmi a sfruttare tali potenzialità fu appunto TECO. Acronimo per Text Editor and COrrector, si trattava in pratica dell’aggiornamento di un vecchio editor di linea per telescriventi, adattato alla macchina PDP-6 del laboratorio[4].

Pur rappresentando un notevole passo avanti rispetto ai primi programmi analoghi, TECO presentava ancora dei difetti. Per creare e modificare un documento, il programmatore doveva inserire una serie di comandi

specificando ogni singola operazione di modifica. Si trattava di un processo del tutto astratto. Al contrario degli odierni elaboratori di testi, capaci di aggiornare il testo ogni volta che si preme un tasto, TECO richiedeva di inserire una lunga sequenza di istruzioni seguita da un “end of command” (fine del comando) per poter cambiare una sola frase. Col passar del tempo gli hacker divennero sufficientemente abili da scrivere interi documenti in questa modalità, ma come sottolineò in seguito Stallman il processo richiedeva “uno sforzo mentale simile a quello di una partita a scacchi giocata a occhi bendati.”[5]

Onde facilitare la procedura, gli hacker del laboratorio di intelligenza artificiale misero a punto un sistema in grado di visualizzare entrambe le modalità (“source” e “display”) sullo schermo diviso a metà. Nonostante la mossa innovativa, il passaggio da una modalità all’altra rappresentava pur sempre una gran seccatura.

TECO non era l’unico elaboratore testi a tutto schermo che girava nel mondo informatico di allora. Nel 1976 durante una visita al laboratorio di intelligenza artificiale di Stanford, Stallman si imbattè in un editor chiamato E. Il programma comprendeva una funzione interna grazie alla quale l’utente poteva aggiornare il testo visualizzato dopo aver digitato ogni tasto. Nel lessico della programmazione anni ’70, si può dire che E era uno dei primi rudimentali esempi di software WYSIWYG. Abbreviazione di “what you see is what you get” (quello che vedi è quello che ottieni), il termine stava ad indicare la possibilità, da parte dell’utente, di manipolare il file spostando direttamente il testo prescelto, anziché operare tramite una serie ulteriore di istruzioni.[6]

Colpito da quell’idea, al MIT Stallman si mise all’opera per espandere le funzionalità di TECO in maniera analoga. Partì da una funzione nota come Control-R, scritta da Carl Mikkelson e così chiamata per la combinazione dei tasti tramite cui veniva attivata. Tale combinazione consentiva il passaggio dall’abituale modalità di esecuzione astratta a quella più intuitiva operabile tasto dopo tasto. Stallman modificò la funzione in maniera sottile ma significativa, rendendo possibile l’attivazione di altre stringhe di comando,

cioè le cosiddette “macro”, mediante diverse combinazioni di tasti. Laddove in precedenza l’utente doveva digitare una stringa di istruzioni per poi perderla al momento di passare alla riga successiva, la funzione ideata da Stallman permetteva di salvare su file le macro utilizzate così da richiamarle all’occorrenza. La modifica di Mikkelson aveva elevato TECO al livello di un editor WYSIWYG, quella di Stallman al livello di un editor WYSIWYG programmabile dall’utente. “Si trattò di una vera e propria rivoluzione”, sostiene Guy Steele, uno degli hacker allora presenti nel laboratorio di intelligenza artificiale del MIT.

Nel ricordo dello stesso Stallman, l’idea delle macro provocò un’esplosione di innovazioni a catena. “Ognuno costruì la propria serie di comandi per l’editor, ciascuno dei quali attivava le funzioni più comunemte utilizzate”, avrebbe successivamente ricordato. “Le macro venivano passate di mano in mano e migliorate, così da ampliarne la portata e l’utilizzo. Gradualmente le raccolte di macro si trasformarono in veri e propri programmi”. L’innovazione si rivelò utile per così tante persone, che la inserirono nella loro versione di TECO, tanto che quest’ultimo divenne secondario rispetto alla mania delle macro così generata. “Iniziammo a categorizzarlo mentalmente come un linguaggio di programmazione piuttosto che un elaboratore testi”, aggiunge Stallman. “Gli utenti si dilettavano ad affinare il software e a scambiarsi nuove idee.”

Due anni dopo quel boom, il tasso di innovazione cominciò a mostrare pericolosi effetti collaterali. La crescita esplosiva aveva fornito una conferma esaltante della validità dell’approccio collaborativo tra gli hacker, conducendo però anche ad un’eccessiva complessità. “Era l’effetto Torre di Babele”, sostiene Guy Steele.

Questo effetto minacciava di uccidere lo stesso spirito da cui aveva tratto origine. Il sistema ITS (Incompatible Time Sharing) era stato progettato dagli hacker per facilitare la condivisione delle informazioni e migliorare reciprocamente il lavoro degli altri. Ciò significava essere in grado di sedersi davanti al monitor di un altro programmatore per inserire direttamente commenti e correzioni nel software su cui stava lavorando. “Talvolta la

maniera più semplice per mostrare a qualcuno come scrivere o sistemare il codice era semplicemente quella di sedersi al terminale e farlo al posto suo", spiega Steele.

A un certo punto, dopo due anni di applicazioni, le macro iniziarono a mascherare le funzioni stesse del programma. Desiderosi di sfruttare al meglio le nuove funzionalità a tutto schermo, gli hacker avevano personalizzato a tal punto le loro versioni di TECO che, sedendosi davanti al terminale di qualcun altro ci voleva un'ora buona solo per scoprire quali macro eseguissero le diverse operazioni.

Frustrato, Steele si fece carico di risolvere il problema. Raccolse quattro diversi pacchetti di macro e iniziò a compilare un grafico che documentasse i comandi più utili. Fu mentre lavorava all'implementazione dei vari comandi specifici che Steele attrasse l'attenzione di Stallman. "Prese a sbirciare dietro le mie spalle, chiedendomi cosa stessi facendo", rammenta.

Per Steele, hacker tranquillo che raramente si era trovato ad interagire con Stallman, il ricordo è ancora palpabile. Osservare un hacker al lavoro da dietro le spalle costituiva un'attività comune in quel laboratorio. Stallman, incaricato della gestione di TECO, definì "interessante" l'idea di Steele e decise di condurla rapidamente in porto.

"Come mi piace sostenere, io curai il primo 0,001 per cento dell'implementazione e Stallman si occupò del resto", ricorda Steele con un sorriso.

Il nome per il nuovo progetto, Emacs, venne scelto dallo stesso Stallman. Abbreviazione di "editing macros", stava a indicare la trascendenza evolutiva avvenuta nel corso di quei due anni dall'esplosione delle macro. Approfittava inoltre di un vuoto nel lessico della programmazione interna. Notando la mancanza nel sistema ITS di programmi che iniziavano con la lettera "E", la scelta di Emacs ne rese possibile l'identificazione tramite la sola lettera iniziale. Ancora una volta la brama degli hacker per la massima efficienza aveva centrato l'obiettivo[7].

Mentre lavoravano allo sviluppo di un sistema standard per i comandi macro, Stallman e Steele dovettero fare alcune acrobazie politiche. Con la

creazione di un programma standard, Stallman si poneva in chiara violazione di uno dei comandamenti dell'etica hacker: "promuovi la decentralizzazione". E contemporaneamente minacciava di ostacolare la flessibilità che aveva inizialmente alimentato l'esplosiva innovazione di TECO.

"Da una parte stavamo cercando di ricreare una serie uniforme di comandi, dall'altra volevamo mantenerli aperti salvaguardando l'importanza della programmabilità", puntualizza Steele.

Per risolvere il problema, Stallman, Steele e i colleghi David Moon e Dan Weinreib decisero allora di limitare tale standardizzazione alle istruzioni WYSIWYG che controllavano la visualizzazione del testo sul monitor. Il resto del progetto Emacs avrebbe conservato l'estensibilità a piacimento dei comandi.

Stallman si trovava però di fronte a un nuovo rompicapo: se gli utenti avessero apportato modifiche senza comunicarle al resto della comunità, l'effetto Torre di Babele sarebbe semplicemente riemerso altrove. Rifacendosi alla dottrina hacker sulla condivisione delle innovazioni, Stallman inserì nel codice sorgente una dichiarazione che ne stabiliva le modalità d'utilizzo. Ciascuno era libero di modificare e ridistribuire il codice a condizione di informare gli altri sulle ulteriori estensioni ideate. E scelse come soprannome "Emacs Commune" (la comune dell'Emacs). Così come TECO era divenuto qualcosa di più di un semplice elaboratore testi, Emacs era diventato qualcosa di più di un semplice programma. Per Stallman si trattava di un contratto sociale. In uno dei primi appunti relativi al progetto, ne descrisse in dettaglio i termini. "L'EMACS", scrisse, "viene distribuito sulla base della condivisione comune, ovvero ogni miglioramento deve essere ritrasmesso a me che provvederò a incorporarlo nelle successive redistribuzioni."[8]

Non tutti accettarono quel contratto. Il ritmo esplosivo dell'innovazione proseguì per tutto il decennio, dando vita a una miriade di programmi simili all'Emacs con diversi livelli di compatibilità tra loro. Alcuni indicavano la relazione con l'originale progetto di Stallman tramite nomi ironicamente ricorsivi: Sine (Sine is not Emacs), Eine (Eine is not Emacs), Zwei (Zwei was Eine initially). In quanto devoto esponente dell'etica hacker, Stallman non

vide alcun motivo per bloccare questa ventata innovativa ricorrendo a pressioni legali. E tuttavia, il fatto che qualcuno decidesse di appropriarsi di software appartenente all'intera comunità, per poi modificarlo e dargli un nuovo nome, dimostrava una palese mancanza di stile.

Tale comportamento si affiancava ad altri sviluppi negativi in atto nella comunità hacker. La decisione di Brian Reid del 1979 di inserire "bombe a tempo" all'interno di Scribe, consentendo così alla Unilogic di limitare l'accesso al software da parte di utenti non-paganti, rappresentò un oscuro presagio per Stallman. "Lo ritenne l'evento peggiore, addirittura nazista, in cui si fosse mai imbattuto", ricorda Reid. Pur avendo raggiunto successivamente una certa notorietà in quanto coideatore della gerarchia "alt" su Usenet, Reid afferma che deve ancora farsi perdonare quella decisione del 1979, almeno agli occhi di Stallman: "secondo lui, tutto il software dovrebbe essere libero e l'idea di far pagare un programma andrebbe considerata un crimine contro l'umanità."[9]

Pur se impossibilitato a fare alcunché per impedire quella manovra di Reid, Stallman riuscì a ridimensionare altri comportamenti ritenuti contrari all'etica hacker. In qualità di gestore centrale del codice per la "comune" dell'Emacs, iniziò a utilizzare quella posizione di potere a scopo politico. Nella fase finale del conflitto con gli amministratori del laboratorio d'informatica sulla questione delle password, Stallman lanciò uno "sciopero del software"[10], rifiutandosi di fornire allo staff di quel laboratorio le ultime versioni di Emacs fino a quando non avessero eliminato il sistema di sicurezza dai computer del laboratorio. La mossa non migliorò molto la crescente reputazione di estremista acquisita da Stallman, raggiunse però l'obiettivo prefissato: gli aderenti alla comune si sarebbero fatti sentire a difesa dei valori base degli hacker.

"Parecchia gente ce l'aveva con me sostenendo che li stessi tenendo in ostaggio o ricattando, e sotto certi aspetti avevano ragione", avrebbe raccontato in seguito Stallman al giornalista Steven Levy. "Avevo lanciato un'operazione violenta nei loro confronti perché ritenevo fossero costoro ad agire violentemente contro tutti gli altri in senso più generale."[11]

Col passare del tempo, Emacs si trasformò in un veicolo per la diffusione dell'etica hacker. La flessibilità di cui lo avevano dotato Stallman e gli altri hacker non soltanto incoraggiava la collaborazione ma anzi la imponeva. Gli utenti che non si aggiornavano sullo sviluppo del software o che non inviavano a Stallman i propri contributi, correvano il rischio di rimanere tagliati fuori dalle ultime novità. E queste erano tutt'altro che poca cosa. A vent'anni dalla sua creazione, gli utenti avevano modificato Emacs per consentirne l'impiego in molti ambiti – come foglio di calcolo, calcolatrice, database, browser Web – al punto che gli ultimi sviluppatori, per rappresentarne l'ampia versatilità, adottarono l'immagine di un lavandino in cui l'acqua fuoriesce. "Era questa l'idea che volevamo comunicare", spiega Stallman. "La quantità di funzioni al suo interno è allo stesso tempo meravigliosa e terrificante."

I contemporanei di Stallman nel laboratorio di intelligenza artificiale si mostrano più benevoli. Hal Abelson, laureando al MIT, già collaboratore di Stallman negli anni '70 e successivamente membro del direttivo della Free Software Foundation, descrive l'Emacs come "una creazione assolutamente brillante". Consentendo ai programmatori di aggiungere nuove librerie e funzioni senza mettere sottosopra il sistema, secondo Abelson, Stallman aprì la strada verso futuri progetti analoghi su larga scala. "La struttura si dimostrò sufficientemente robusta da permettere l'aperta collaborazione da parte di persone di ogni parte del mondo", aggiunge. "Non mi risulta che questo sia mai stato fatto prima."[12]

Analoga l'ammirazione espressa da Guy Steele. Attualmente ricercatore presso la Sun Microsystems, ricorda Stallman soprattutto come "brillante programmatore con la capacità di generare notevoli quantità di codice sostanzialmente privo di errori." Pur tra le rispettive incompatibilità caratteriali, i due collaborarono abbastanza a lungo da permettere a Steele di apprezzare la profondità dello stile di Stallman nella programmazione. Ne rammenta in particolare un episodio specifico verso la fine degli anni '70, quando lavorarono assieme alla stesura della funzione "pretty print", per la stampa del codice. Inizialmente concepita da Steele, tale funzione veniva

attivata dall'ennesima combinazione di tasti che riformattava il codice sorgente in modo da risultare più leggibile e occupare meno spazio, esaltandone ulteriormente le qualità WYSIWIG. Un'opzione sufficientemente strategica da attrarre l'attivo interesse di Stallman, e non ci volle molto perché i due ne progettassero insieme una versione migliorata.

"Una mattina ci sedemmo davanti al monitor", ricorda Steele. "Io ero alla tastiera e lui di fianco. Non aveva problemi a lasciarmi digitare, ma era lui a suggerirmi cosa fare."

La sessione durò 10 ore. Durante tutto quel tempo, aggiunge Steele, né lui né Stallman fecero alcuna pausa, e nemmeno avviarono qualche breve conversazione. Alla fine erano riusciti a ridurre il codice per la funzione di stampa a meno di 100 righe. "Per tutto quel tempo, le mie dita erano rimaste sulla tastiera", ricorda Steele, "ma erano le idee di entrambi a fluire sullo schermo. Lui mi diceva cosa digitare, e io procedevo."

La lunghezza della sessione si rese evidente quando finalmente Steele uscì dal laboratorio. Una volta fuori dall'edificio al 545 di Tech Square, rimase sorpreso di trovarsi circondato dal buio della notte. Come programmatore Steele era abituato a maratone analoghe. Eppure stavolta c'era qualcosa di diverso. Quel lavoro con Stallman lo aveva costretto a bloccare ogni stimolo esterno per concentrare tutte le sue energie mentali sull'unico obiettivo che avevano davanti. Ripensando a quella circostanza, Steele sostiene di aver considerato la forza mentale di Stallman esilarante e terrificante al tempo stesso. "Il mio primo pensiero subito dopo fu questo: è stata una grande esperienza, molto intensa, ma non avrei mai voluto ripeterla in vita mia."

[1] Si veda Josh McHugh, "For the Love of Hacking", *Forbes*, 10 agosto 1998. http://www.forbes.com/forbes/1998/0810/6203094a.html

[2] Si veda Stallman (1986).

[3] Si veda Joseph Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation*, W. H. Freeman, 1976, p. 116.

[4] Secondo il Jargon File, orginariamente l'acronimo TECO stava per Tape Editor and Corrector. http://www.tuxedo.org/~esr/jargon/html/entry/TECO.html

[5] Si veda Richard Stallman, “EMACS: The Extensible, Customizable, Display Editor”, annotazione interna per il laboratorio di intelligenza artificiale (1979). Una versione HTML aggiornata di questa nota, dalla quale ho tratto la citazione, è disponibile online: http://www.gnu.org/software/emacs/emacs-paper.html.

[6] Si veda Richard Stallman, “Emacs the Full Screen Editor” (1987). http://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt

[7] http://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt

[8] Si veda Stallman (1979): #SEC34.

[9] In un’intervista del 1996 per la rivista online *MEME*, Stallman definì incresciosa la vendita di Scribe, ma esitò nel fare il nome di Reid. “Il problema fu che nessuno censurò o punì quello studente per quanto aveva combinato”, sostenne Stallman. “Come conseguenza, altre persone furono tentate di seguirne l’esempio.” Si veda MEME 2.04. http://memex.org/meme2-04.html

[10] Si veda Steven Levy, *Hackers*, Penguin USA, 1984, p. 419.

[11] Ibid.

[12] In questo capitolo ho scelto di concentrarmi maggiormente sul significato sociale dell’Emacs piuttosto che su quello legato al software. Per saperne di più in tal senso, raccomando la nota di Stallman del 1979. Consiglio in particolare la sezione denominata “Research Through Development of Installed Tools” (#SEC27). Non soltanto risulta comprensibile per il lettore non-tecnico, ma mette altresì in evidenza la stretta interconnessione tra la filosofia politica e quella relativa allo sviluppo del software nella concezione di Stallman. Quanto segue è un breve estratto: “L’EMACS non avrebbe potuto vedere la luce tramite un processo di attenta progettazione, poiché simili procedure raggiungono l’obiettivo preposto soltanto nel caso questo risulti visibile e sia auspicabile fin dall’inizio. Né il sottoscritto né altri potevamo immaginare un elaboratore testi estendibile prima di averlo realizzato, e nessuno avrebbe potuto apprezzarne il valore prima di averlo utilizzato sul campo. L’EMACS esiste perché mi sentivo libero di inserirvi miglioramenti ridotti e unici lungo una strada la cui direzione era impossibile da intravedere.”

Una difficile scelta morale

Il 27 settembre 1983 i programmatori che seguivano il newsgroup net.unix-wizards su Usenet incontrarono un messaggio insolito. Inserito nelle ore piccole, per l'esattezza mezz'ora dopo mezzanotte e firmato da *rms@mit-oz*, appariva con un oggetto semplice ma accattivante: "Nuova implementazione di UNIX". Tuttavia il paragrafo d'apertura del messaggio, anziché presentare una nuova versione di Unix, sembrava una chiamata alle armi:

> A partire dal prossimo giorno del Ringraziamento inizierò a scrivere un sistema pienamente compatibile con Unix chiamato GNU (Gnu's Not Unix), distribuito gratuitamente a chiunque vorrà usarlo. Urgono contributi in tempo, denaro, programmi e attrezzature[1].

Agli occhi di un esperto programmatore Unix il messaggio appariva come un miscuglio tra idealismo e arroganza. Non soltanto l'autore voleva ricostruire da capo un sistema operativo già maturo, ma si proponeva perfino di migliorarlo. Il nuovo sistema GNU, si spiegava in quel testo, avrebbe incluso tutti i componenti di base – un elaboratore testi, un programma shell per far girare le applicazioni compatibili con Unix, un compilatore e "alcune altre cose."[2]. Né sarebbero mancate diverse opzioni stimolanti, tuttora assenti in Unix: un'interfaccia grafica basata sul linguaggio di programmazione Lisp, un file system a prova di blocco e una serie di protocolli di rete basati sull'infrastruttura utilizzata al MIT.

"GNU potrà far girare programmi Unix, ma non sarà un suo duplicato", spiegava il messaggio. "Vi apporteremo tutti i miglioramenti ritenuti necessari, facendo tesoro delle esperienze avute con altri sistemi operativi."

A prevenire lo scetticismo di qualche lettore, l'annuncio sul nuovo sistema operativo era seguito da una breve biografia dell'autore, intitolata "Chi sono?":

> Mi chiamo Richard Stallman, creatore dell'originale EMACS, un elaboratore testi assai imitato, e sono attualmente nello staff del laboratorio di intelligenza artificiale del MIT. Ho lavorato a lungo su compilatori, editor, debugger e interpreti di comandi, sull'Incompatible Time Sharing System e sul sistema operativo delle Lisp Machines. All'interno dell'ITS ho ideato il supporto per la visualizzazione indipendente dal terminale. Inoltre ho implementato un file system a prova di blocco e due sistemi a finestre per Lisp Machines[3].

Per un gioco del destino, il progetto GNU così delineato mancò la data del lancio fissata per il giorno del Ringraziamento (fine novembre). Ma dal gennaio 1984 Stallman rispettò la promessa, immergendosi totalmente nell'ambiente di sviluppo Unix. Per un architetto del software cresciuto sull'ITS, era un po' come progettare grandi magazzini di periferia anziché palazzi moreschi. Tuttavia anche la realizzazione di un sistema operativo analogo a Unix presentava dei vantaggi nascosti. L'ITS si era rivelato assai potente, ma aveva un tallone d'Achille: gli hacker del MIT l'avevano progettato in modo da trarre il massimo dal modello PDP. Quando quest'ultimo venne eliminato, all'inizio degli anni '80, su decisione degli amministratori del laboratorio, quel sistema operativo, un tempo simile a una vibrante metropoli, diventò improvvisamente una città fantasma. Unix, d'altra parte, era stato ideato tenendo conto delle opzioni di mobilità e funzionalità a lungo termine. Sviluppato originariamente dai giovani ricercatori di AT&T, il programma era sfuggito al radar delle grandi corporation, trovando felice dimora nel mondo squattrinato dell'informatica accademica. Avendo a disposizione risorse inferiori a quelle dei colleghi al MIT, gli sviluppatori Unix si erano curati di personalizzarlo in modo da girare su un vasto assortimento di hardware: qualunque cosa compresa tra il PDP-11 a 16-bit – macchina adatta solo per piccole attività, secondo la maggioranza degli hacker nel laboratorio di intelligenza artificiale – e i mainframe a 32-bit come il VAX 11/780. A partire dal 1983 alcune aziende, prima fra tutte la Sun Microsystems, avevano iniziato a spingersi anche oltre, mettendo a punto una nuova generazione di microcomputer, definiti "workstation", per sfruttare la crescente trasferibilità di quel sistema operativo.

Per facilitare questo processo, gli sviluppatori responsabili delle maggiori versioni di Unix si assicurarono di mantenere un strato di astrazione in più tra

il software e la macchina. Anziché tagliare su misura il sistema operativo per sfruttare una macchina specifica – come avevano fatto gli hacker del laboratorio di intelligenza artificiale con l'ITS e il PDP-10 – i programmatori Unix optarono per un approccio più generico e meno specializzato. Concentrandosi maggiormente sugli standard e le specifiche che tenevano insieme i numerosi sottocomponenti del sistema, anziché sui componenti veri e propri, riuscirono a creare un sistema che poteva essere modificato velocemente per adattarsi ai gusti di qualunque macchina. Se un utente si lamentava del funzionamento di una parte, l'adozione di questi standard rendeva possibile l'estrazione di singole sotto-componenti e la loro correzione o sostituzione con qualcosa di meglio. In sintesi, quel che mancava a Unix sotto l'aspetto dello stile o dell'estetica veniva più che ricompensato dalla flessibilità ed economicità, che ne garantirono la rapida adozione[4].

La decisione di Stallman di avviare lo sviluppo del sistema GNU venne provocata dalla fine di quel sistema ITS che gli hacker del laboratorio avevano coltivato così a lungo. La sua scomparsa si rivelò un colpo traumatico per Stallman. Avvenuta sulla scia dell'episodio della stampante laser Xerox, tale scomparsa offrì un'ulteriore prova del fatto che la cultura hacker stesse perdendo la propria immunità nei confronti di quelle pratiche commerciali tipiche del mondo esterno.

Come per il codice del software su cui era basato, le radici che portarono all'estinzione dell'ITS andavano cercate nel recente passato. Negli anni successivi alla guerra del Vietnam i fondi a disposizione del Ministero della Difesa, da tempo maggior finanziatore della ricerca informatica, avevano subito drastiche riduzioni. Nella disperata ricerca di nuovi contributi, laboratori e università si erano così rivolti al settore privato. Per il laboratorio di intelligenza artificiale, ottenere il sostegno degli investitori privati si rivelò un gioco da ragazzi. Già sede di alcuni dei progetti più ambiziosi del dopoguerra, il laboratorio divenne rapidamente un'incubatrice di tecnologia. In realtà a partire dal 1980 la maggioranza dello staff interno, compresi numerosi hacker, divideva il proprio tempo tra il MIT e vari progetti commerciali.

Quel che inizialmente pareva un patto vincente per entrambi i fronti – gli hacker lavoravano su progetti d'avanguardia e in cambio potevano utilizzare le tecnologie più aggiornate – si rivelò presto una sorta di patto Faustiano. Più era il tempo dedicato a tali progetti commerciali, e meno ne rimaneva per il mantenimento della barocca infrastruttura del laboratorio. Le aziende iniziarono poi ad assumere direttamente gli hacker nel tentativo di monopolizzarne tempo e attenzione. Con la diminuzione degli hacker presenti, divenne sempre più difficile assicurare la corretta gestione di macchine e programmi. Ancor peggio, sostiene Stallman, il laboratorio iniziò a subire un "mutamento demografico". Gli hacker che una volta costituivano una rumorosa minoranza stavano perdendo aderenti mentre "quei professori e studenti a cui non piaceva granché [il PDP-10] erano numerosi come sempre"[5]. Il punto di rottura si manifestò nel 1982. Fu in quell'anno che l'amministrazione decise di procedere con l'aggiornamento del computer centrale, il PDP-10. La casa produttrice, Digital, aveva messo fuori commercio quel modello. Pur offrendo ancora un mainframe di alta potenza, denominato KL-10, questo richiedeva una drastica operazione di riscrittura dell'ITS, nel caso gli hacker volessero "portare" quel sistema operativo sulla nuova macchina. Temendo che il laboratorio avesse perso la massa critica di programmatori di talento, i membri di facoltà spinsero per Twenex, un sistema operativo commerciale realizzato da Digital. Rimasti in minoranza, agli hacker non rimase altra scelta che adeguarsi.

"Mancando gli hacker capaci di mantenere il sistema, [i membri di facoltà] dissero 'ci aspetta il disastro, meglio dotarci di un software commerciale"', così avrebbe ricordato Stallman dopo qualche anno. "Aggiungevano, 'in tal modo alla manutenzione ci penserà la casa produttrice.' Purtroppo si sbagliavano di grosso, ma presero comunque quella decisione."[6]

All'inizio gli hacker considerarono il sistema Twenex come un ulteriore simbolo autoritario che andava sovvertito. Il nome stesso suscitava un grido di protesta. Ufficialmente chiamato TOPS-20 dalla Digital, si trattava del successore del TOPS-10, il sistema operativo commercializzato per il PDP-10. La Bolt Beranek & Newman ne aveva messo a punto una versione migliorata,

nota come Tenex.[7] Da qui derivò Twenex, termine coniato da Stallman per evitare di usare il nome originale TOPS-20. “Il sistema era ben lungi dall’essere uno dei migliori (“top”), per cui mi rifiutai di chiamarlo in quel modo.” ricorda Stallman. “Decisi perciò di inserire una ‘w’ nel nome e venne fuori Twenex.”

Alla macchina su cui girava il Twenex/TOPS-20 venne affibbiato un diminuitivo altrettanto sarcastico: Oz. Secondo una leggenda hacker, il computer richiedeva una macchina più piccola, il PDP-11, per attivare il proprio terminale. Uno degli hacker, trovandosi davanti per la prima volta alle impostazioni del KL-10-PDP-11, li paragonò alla roboante introduzione de *Il Mago di Oz*. “Io sono il grande e potente Oz”, intonò l’hacker. “Non prestate alcuna attenzione al PDP-11 nascosto dietro quella console.”[8] Tuttavia la risata con cui gli hacker accolsero il KL-10 svanì non appena ebbero dato un’occhiata a Twenex. Non soltanto il sistema vantava opzioni interne di sicurezza, ma queste facevano da sottofondo a ogni strumento e applicazione realizzati dagli ingegneri responsabili della progettazione. Quello che una volta sembrava un gioco tra gatto e topo sulle password nel sistema di sicurezza del laboratorio d’informatica, si era trasformato in una vera e propria battaglia sulla gestione del sistema. Secondo gli amministratori, una volta privato delle opzioni di sicurezza Oz sarebbe risultato più suscettibile a blocchi accidentali. Per gli hacker, invece, la migliore prevenzione di tali blocchi consisteva nel modificare il codice originale. Purtroppo il numero di hacker dotati di tempo e capacità per farlo si era ridotto a tal punto che fu la tesi degli amministratori a prevalere.

Scroccando le password altrui e provocando deliberatamente il blocco del sistema come prova dei problemi che ne derivavano, Stallman riuscì a frustrare i tentativi degli amministratori di esercitare pieno controllo sul sistema. Dopo un fallito “colpo di stato”, Stallman diffuse un messaggio di allerta all’intero staff del laboratorio[9]:

“C’è stato un altro tentativo di conquistare il potere”, si leggeva nel testo. “Per ora gli assalti dell’aristocrazia sono stati respinti.” A tutela della propria identità, Stallman firmò il messaggio come “Radio Free OZ.”

Un camuffamento piuttosto debole. Già dal 1982 la sua avversione per ogni password e segretezza era talmente nota che gli utenti esterni al laboratorio ne usavano l'account come ponte per raggiungere ARPAnet, la rete informatica per la ricerca cui si devono le fondamenta dell'odierna Internet. All'inizio degli anni '80 tra questi "turisti" c'era anche Don Hopkins, programmatore californiano informato dal giro hacker che per accedere al mitico sistema ITS del MIT bastava inserire le iniziali RMS come login e lo stesso monogramma per la relativa password.

"Sarò eternamente grato al MIT per aver consentito al sottoscritto e a molta altra gente il libero accesso a quelle macchine", afferma Hopkins. "Per parecchi di noi ciò è stato davvero importante."

Questa politica riservata ai "turisti", apertamente tollerata dagli amministratori del MIT durante gli anni del sistema ITS[10], subì un duro contraccolpo quando Oz divenne il collegamento primario tra il laboratorio e ARPAnet. Inizialmente Stallman continuò nella pratica di ripetere i medesimi login e password in modo che gli utenti esterni potessero imitarlo. Tuttavia, con l'andar del tempo la fragilità di Oz costrinse gli amministratori a impedire l'accesso agli estranei, i quali, sia per puro caso sia con intenti dolosi, potevano bloccare il sistema. Quando finalmente gli stessi amministratori imposero a Stallman di smetterla di diffondere la propria password, egli rifiutò di aderire sulla base dell'etica personale e smise completamente di usare Oz[11].

"[Quando] le password fecero la loro comparsa al laboratorio di intelligenza artificiale del MIT [decisi] di seguire le mie convinzioni contrarie all'esistenza di password", avrebbe spiegato in seguito Stallman. "Poiché non credo sia auspicabile dotare un computer delle opzioni di sicurezza, non ho intenzione di avvallare l'esistenza di un siffatto regime."[12]

Quel rifiuto di inchinarsi di fronte al grande e potente Oz simbolizzava la crescente tensione tra gli hacker e la direzione del laboratorio nei primi anni '80. Una tensione che tuttavia impallidiva in raffronto al conflitto in corso all'interno della stessa comunità hacker. Con l'arrivo del KL-10, questa appariva già spaccata su due fronti contrapposti. Il primo ruotava intorno a un'azienda di software nota come Symbolics, Inc. Il secondo invece ne

appoggiava il diretto rivale, Lisp Machines, Inc. (LMI). Entrambe le società erano in lotta per lanciare sul mercato la Lisp Machine, dispositivo costruito per sfruttare al meglio il linguaggio di programmazione Lisp.

Creato dal pioniere degli studi sull'intelligenza artificiale John McCarthy, ricercatore presso il MIT sul finire degli anni '50, il Lisp è un linguaggio elegante e appropriato per programmi finalizzati a poderosi lavori di classificazione e smistamento. Il nome è la versione contratta di LISt Processing. Dopo il passaggio di McCarthy al laboratorio di intelligenza artificiale di Stanford, gli hacker del MIT lo perfezionarono in un dialetto locale soprannominato MACLISP. Il prefisso "MAC" si riferiva al progetto MAC finanziato dal DARPA, da cui nacquero poi il laboratorio di intelligenza artificiale e quello di computer science. Sotto la guida dell'abile hacker Richard Greenblatt, nel corso degli anni '70 i programmatori del laboratorio di intelligenza artificiale misero a punto un intero sistema operativo basato su Lisp, chiamato in seguito sistema operativo Lisp Machine. Nel 1980, il progetto si era già diviso in due derivazioni commerciali: alla direzione della Symbolics c'era Russell Noftsker, già amministratore del laboratorio, mentre la Lisp Machines, Inc. (LMI) era guidata da Greenblatt.

Il software per la Lisp Machine era dovuto a quei primi hacker, ovvero la proprietà spettava al MIT pur rimanendo a disposizione di chiunque volesse copiarlo, nel pieno rispetto della tradizione hacker. Una pratica questa che limitava l'attività commerciale di qualunque impresa volesse ottenere la licenza di un programma del MIT per poi lanciarlo sul mercato come prodotto specifico. Per assicurarsi qualche vantaggio, ed esaltare gli aspetti del sistema operativo che potessero meglio attrarre i consumatori, le aziende ingaggiarono diversi hacker del laboratorio mettendoli all'opera su particolari componenti del sistema operativo per la Lisp Machine al di fuori degli auspici del laboratorio stesso.

La più aggressiva in tal senso si dimostrò la Symbolics. Entro la fine del 1980, questa aveva assunto 14 addetti del laboratorio come consulenti part-time per sviluppare la sua versione della Lisp Machine. Tranne Stallman, gli altri firmarono invece con la LMI[13].

Inizialmente Stallman sembrò accettare i tentativi commerciali di entrambe le aziende, anche se ciò comportava una maggiore mole di lavoro per lui. Entrambe ottennero dal MIT la licenza per i sorgenti, ed era compito di Stallman aggiornare la Lisp Machine del laboratorio per tener dietro alle innovazioni più recenti. Nonostante la licenza ottenuta dalla Symbolics consentisse a Stallman il diritto di rivederne il codice, ma non di copiarlo, egli sostiene che un "gentleman's agreement" tra la direzione della Symbolics e il laboratorio gli permetteva di prendere in prestito i frammenti più interessanti nel tipico stile hacker.

Il 16 marzo 1982, una data che Stallman rammenta bene perché era il suo compleanno, i dirigenti della Symbolics decisero di porre fine a quell'accordo tra gentiluomini. Si trattava di una manovra chiaramente strategica. La LMI, diretta concorrente per il mercato della Lisp Machine, sostanzialmente utilizzava la copia del sistema in dotazione al laboratorio di intelligenza artificiale. Anziché foraggiare la crescita del diretto concorrente, la Symbolics impose la stretta osservanza del testo della licenza. Se il laboratorio voleva tenere aggiornato il proprio sistema operativo con quello della Symbolics doveva utilizzare una macchina di quest'ultima e tagliare ogni rapporto con la LMI.

Stallman, responsabile del mantenimento della Lisp Machine del laboratorio, era furibondo. Considerando la decisione come un "ultimatum", replicò con la disconnessione del canale di comunicazione a microonde tra la Symbolics e il laboratorio stesso. Poi giurò a se stesso che non avrebbe mai lavorato su una macchina della Symbolics e si schierò apertamente a favore della LMI. "Dal mio punto di vista il laboratorio era un paese neutrale, come il Belgio durante la prima guerra mondiale", spiega Stallman. "Ma se la Germania invade il Belgio, allora questo dichiara guerra alla Germania e si allea con Francia e Inghilterra."

Le circostanze della cosiddetta "guerra della Symbolics" del 1982-1983 variano parecchio a seconda delle fonti interpellate. Quando i dirigenti dell'azienda si accorsero che le funzioni più recenti continuavano ad apparire nella Lisp Machine del laboratorio, per poi estendersi alla versione della LMI,

decisero di installare un “programma-spia” nel computer di Stallman. Questi sostiene di aver riscritto ogni funzione da zero, approfittando della clausola relativa alla revisione del codice, ma avendo poi cura di riscriverlo in maniera differente per quanto possibile. I dirigenti della Symbolics la pensarono in maniera opposta e portarono il caso all’attenzione degli amministratori del MIT. Stando al libro *The Brain Makers: Genius, Ego, and Greed, and the Quest for Machines That Think*, scritto nel 1984 da Harvey Newquist, l’amministrazione rispose invitando Stallman a “tenersi alla larga” dal progetto della Lisp Machine[14]. Secondo Stallman, invece, gli amministratori del MIT gli diedero man forte. “Non subii alcuna pressione”, sostiene. “Decisi però di apportare delle modifiche a quella pratica. Per essere ultra-sicuro, non leggevo più il loro codice. Usavo soltanto la documentazione e procedevo a partire da quella.”

Comunque sia, la lite non fece altro che rafforzare la posizione di Stallman. Senza rivedere il codice, colmò ogni lacuna seguendo i gusti personali e coinvolse gli altri del laboratorio nella tempestiva segnalazione di eventuali problemi. Si assicurò, inoltre, che i programmatori della LMI avessero accesso diretto ai cambiamenti apportati. “Avevo deciso di punire la Symbolics, fosse stata l’ultima azione della mia vita”, dice Stallman.

Si tratta di un’affermazione rivelatrice, non soltanto perché getta luce sull’indole non-pacifista di Stallman, ma anche perché riflette l’intenso livello emotivo suscitato dal conflitto. Secondo un altro articolo citato da Newquist, ad un certo punto Stallman sembrò talmente fuori di sé che diffuse una e-mail in cui minacciava di “imbottirsi di dinamite per poi entrare negli uffici della Symbolics.”[15] Pur negando alcun ricordo di tale e-mail e descrivendone l’eventuale esistenza come una “voce maligna”, Stallman riconosce tuttavia che un simile pensiero gli attraversò la mente. “Pensavo veramente di uccidermi e distruggere così la loro sede”, ricorda. “Credevo che la mia vita fosse finita.”[16]

Un simile livello di disperazione è dovuto a quello che Stallman interpretava come la “distruzione della propria casa”, ovvero la fine della cultura hacker tipica del laboratorio di intelligenza artificiale. In una

successiva intervista via e-mail con Levy, Stallman si paragonò alla figura storica di Ishi, ultimo sopravvissuto della tribù Yahi, spazzata via dalle guerre contro gli indiani d'America in California nel periodo 1860-1870. L'analogia dipinge la posizione di Stallman in termini epici, quasi mitologici. In realtà tende però a mascherare la tensione tra lo stesso Stallman e gli altri hacker del laboratorio, una tensione preesistente allo scisma Symbolics-LMI. Anziché vedere la Symbolics come una potenza di sterminio, molti colleghi consideravano appropriata quell'iniziativa. Con il lancio sul mercato della Lisp Machine, l'azienda spinse i principi hacker sulla progettazione ingegneristica del software al di fuori della torre d'avorio del laboratorio, per raggiungere un mercato imprenditoriale in cui vigeva la progettazione manageriale. Lungi dal considerare Stallman un bastione etico, non pochi hacker lo giudicavano una figura inutile e anacronistica.

Stallman non contesta questa diversa spiegazione degli eventi. Aggiunge anzi un ulteriore motivo a spiegazione del clima di ostilità suscitato dall'"ultimatum" della Symbolics. Ancor prima che quest'ultima assumesse gran parte dello staff del laboratorio, secondo Stallman molti di loro avevano già iniziato ad emarginarlo. "Non venivo più invitato ad andare a Chinatown", ricorda. "Stando all'usanza lanciata da Greenblatt, prima di andare fuori a cena si chiedeva o si mandava un messaggio di invito a tutti nel laboratorio. A un certo punto verso il 1980-1981, la mia presenza non venne più richiesta. Non soltanto avevano smesso d'invitarmi, ma in seguito qualcuno mi confessò di aver ricevuto pressioni affinché mentisse pur di tenermi all'oscuro di tutto."

Nonostante l'ira di Stallman nei confronti degli hacker che avevano orchestrato questa forma di meschino ostracismo, la controversia con la Symbolics sembrò suscitargli una rabbia di tipo nuovo, quella di una persona prossima a vedersi privata del proprio tetto. Quando la Symbolics smise di inviare i cambiamenti apportati man mano al codice, per tutta risposta Stallman si rinchiuse in ufficio per riscrivere da capo ogni nuova funzione e strumento. Per quanto frustrante, ciò garantiva ai futuri utenti della Lisp Machine il pieno accesso alle medesime funzioni già disponibili a quanti seguivano la Symbolics.

Quel lavoro assicurava altresì il consolidamento della fama leggendaria guadagnata da Stallman all'interno della comunità hacker. Già assai nota per i contributi con l'Emacs, la sua abilità di tener testa all'intero gruppo di programmatori della Symbolics – tra i quali erano presenti non pochi hacker già leggendari – rimane tuttora uno dei maggiori successi umani dell'era dell'informazione, o di ogni altra epoca, per quel che possa valere. Definendolo "maestro dell'hacking" e "il John Henry virtuale del codice", lo scrittore Steven Levy fa notare come molti dei suoi rivali assunti dalla Symbolics non ebbero altra scelta che rendere omaggio al loro ex-collega idealista. Levy cita Bill Gosper, un hacker che alla fine andò a lavorare nell'ufficio della Symbolics a Palo Alto, il quale si dichiara colpito dalla qualità dell'operato di Stallman in quel frangente:

> Dopo aver dato un'occhiata a qualcosa scritto da Stallman, potevo anche decidere che fosse scadente (probabilmente non lo era, ma diciamo che qualcuno mi avesse convinto del contrario), eppure non avrei fatto a meno di notare, 'Un momento, però, Stallman non ha nessuno che lo aiuta o con cui discutere per tutta la notte.
> Lavora completamente da solo! È incredibile che qualcuno possa fare tutto ciò in completa solitudine!"[17]

Per Stallman, i mesi trascorsi a rincorrere la Symbolics evocano un misto tra orgoglio e profonda tristezza. In quanto persona di tendenze liberali il cui padre combatté nella seconda guerra mondiale, Stallman non può definirsi pacifista. Sotto molti aspetti, la guerra con la Symbolics va considerata un rito di passaggio verso il quale andava dirigendosi fin dal suo arrivo al laboratorio un decennio prima. Allo stesso tempo, tuttavia, l'episodio coincideva con la traumatica distruzione di quella cultura hacker in cui si era cullato fin dall'adolescenza. Un giorno, durante una pausa da quel lavoro, Stallman visse un'esperienza traumatica nella stanza che ospitava le macchine del laboratorio. Mentre si sgranchiva le gambe, finì per imbattersi nella massiccia struttura inutilizzata del PDP-10. Davanti a quelle spie ormai fuori uso, spie che una volta si accendevano e si spegnevano continuamente ad indicare lo stato del programma interno, Stallman subì un impatto emotivo analogo a quello che si prova di fronte al cadavere ben conservato di una persona cara.

“Sono scoppiato a piangere in quella stanza”, ricorda. “Vedere lì quella macchina, morta, senza nessuno a prendersene cura, una scena che rifletteva la totale distruzione della mia comunità.”

Stallman avrebbe avuto poco tempo per il rimpianto. La Lisp Machine, nonostante tutto il furore suscitato e il gran lavoro richiesto, non costituiva altro che un evento collaterale tra le grandi battaglie in corso nel mercato della tecnologia. Il ritmo frenetico della miniaturizzazione del computer stava dando vita a microprocessori rinnovati e più potenti che avrebbero presto incorporato le capacità hardware e software in un’unica macchina, al pari di una moderna metropoli che riesce ad inghiottire un antico villaggio del deserto.

Su tali microprocessori giravano centinaia, persino migliaia, di programmi commerciali, ciascuno protetto da un mosaico di licenze d’uso e di accordi di non divulgazione, che impedivano agli hacker l’accesso o la condivisione del codice sorgente. Le licenze erano grezze e inadeguate, ma dal 1983 erano diventate abbastanza precise da soddisfare i tribunali e spaventare possibili “malintenzionati”. Il software, una volta considerato una sorta di decorazione aggiuntiva offerta dai produttori di hardware per rendere più appetitosi i loro costosi computer, stava diventando rapidamente il piatto forte.

Sempre più alla ricerca di nuovi giochi e di nuove funzioni, gli utenti andavano dimenticando la tradizionale richiesta di poter dare un’occhiata alla ricetta, una volta concluso il pasto.

Lo scenario si evidenziava con maggior forza nel mondo del personal computer. Aziende come Apple Computer e Commodore stavano creando nuovi milionari vendendo macchine con sistema operativo incorporato. Ignorando la cultura hacker e il suo rifiuto per il software in solo formato binario, buona parte dell’utenza non vedeva alcun motivo per protestare quando tali aziende facevano a meno di allegare i file con il codice sorgente. Alcuni anarchici aderenti all’etica hacker tentarono di trovare un varco per quelle norme etiche all’interno del nuovo mercato in fieri, ma per lo più quest’ultimo ricompensava i programmatori abbastanza veloci da sfornare

programmi sempre nuovi e sufficientemente scaltri da registrarli come opere d'ingegno posti sotto la tutela legale del copyright.

Uno dei più famosi tra questi programmatori era Bill Gates, il quale aveva abbandonato Harvard due anni dopo la laurea di Stallman. A quei tempi i due non si conoscevano, ma già sette anni prima che Stallman inviasse quel messaggio al newsgroup net.unix-wizards, Gates – giovane imprenditore e partner dell'azienda software Micro-Soft, successivamente divenuta Microsoft, con base ad Albuquerque, New Mexico – aveva fatto girare una lettera aperta nella comunità degli sviluppatori. Scritta in replica a quegli utenti di PC che copiavano i programmi della Micro-Soft, la "Lettera aperta agli hobbisti" mirava a colpire l'idea stessa dello sviluppo comunitario del software.

"Chi può permettersi di lavorare a livello professionale senza essere retribuito?", chiedeva Gates. "Quale hobbista è in grado di dedicare tre anni di lavoro per un programma, sistemarne ogni problema, documentarlo adeguatamente per poi distribuirlo gratuitamente?"[18]

Anche se soltanto pochi hacker del laboratorio d'intelligenza artificiale notarono quel testo nel 1976, la lettera di Gates rappresentava il mutato atteggiamento verso il software sia tra le aziende sia tra gli sviluppatori di ambito commerciale. Perché considerare il software come un bene a costo zero quando il mercato stabiliva il contrario? Nel periodo a cavallo tra la fine degli anni '70 e l'inizio del decennio successivo, la vendita di programmi informatici divenne ben più di un modo per recuperare le spese iniziali; si trattava di una dichiarazione politica. In un periodo in cui negli Stati Uniti l'amministrazione Reagan si affrettava ad allentare molte restrizioni federali nonché quei freni alla spesa pubblica imposti durante il mezzo secolo seguito alla Grande Depressione, parecchi programmatori consideravano l'etica hacker contraria alla competizione commerciale e, per estensione, antiamericana. Al massimo veniva considerata come il ritorno a posizioni anti-corporation tipiche di fine anni '60 e primi '70. Al pari di un banchiere di Wall Street che ritrovava una maglietta dai colori psichedelici tra camicie dal colletto inamidato e vestiti a doppio petto, parecchi programmatori trattavano

l’etica hacker come un imbarazzante ricordo di un periodo idealista del passato.

Per qualcuno che aveva trascorso l’intero arco degli anni ’60 come un inquieto ritorno al decennio precedente, Stallman non sembrava affatto preoccupato di vivere al di fuori della cerchia dei colleghi. Da programmatore abituato a lavorare con le macchine e i software più aggiornati, tuttavia, si trovava di fronte quella che può essere definita soltanto come “una difficile scelta morale”: lasciar perdere le obiezioni etiche contro il software “proprietario” – il termine usato da Stallman e dagli altri hacker per descrivere qualunque programma contenesse un copyright privato oppure una licenza d’uso che ne limitasse la copia e la modifica – oppure dedicare la propria esistenza alla costruzione di un sistema alternativo, non proprietario, per il software. Fu questa seconda ipotesi ad attirare maggiormente Stallman, visti anche i recenti mesi di traversie con la Symbolics. “Credo che a quel punto avrei potuto smettere definitivamente di occuparmi di computer”, aggiunge. “Pur mancando di doti particolari, sono certo che avrei potuto fare il cameriere. Non in un ristorante di classe, probabilmente, ma un posto l’avrei trovato.”

Fare il cameriere – o meglio, smettere di lavorare alla programmazione informatica – avrebbe significato abbandonare completamente un’attività che gli aveva dato soddisfazioni enormi. Ripensando alle sue giornate dopo l’arrivo a Cambridge, Stallman identificava senza problemi lunghi periodi in cui la programmazione aveva costituito l’unico suo piacere. Anziché mollare decise perciò d’insistere.

Da ateo convinto, Stallman rifiuta l’idea di destino, di dharma o di un richiamo divino nella vita. Ciò nonostante, considera una scelta naturale quella decisione di lottare contro il software proprietario e di realizzare un sistema operativo per spingere gli altri a seguire il suo esempio. Dopo tutto era stata la combinazione tra testardaggine, lungimiranza e virtuosismo nella programmazione a condurlo davanti a un bivio di cui molta gente non conosceva neppure l’esistenza. Nel descrivere la sua decisione in un capitolo

contenuto nel volume del 1999 *Open Sources*, Stallman cita la spiritualità contenuta nelle parole del sapiente ebraico Hillel[19]:

> Se non sono per me stesso, chi sarà per me?
> E se sono solo per me stesso, che cosa sono?
> E se non ora, quando?

Negli interventi pubblici preferisce però evitare ogni riferimento religioso e illustra quella svolta in termini pragmatici. “Mi son chiesto: cosa posso fare, in quanto sviluppatore di sistemi operativi, per migliorare la situazione? Dopo un’attenta valutazione, mi resi conto del fatto che per risolvere il problema occorreva proprio uno sviluppatore di sistemi operativi.”

Una volta presa quella decisione, aggiunge Stallman, tutto il resto “andò a posto da solo.” Si sarebbe astenuto dall’utilizzare programmi che lo costringevano a scendere a patti con le proprie scelte morali, mentre al contempo si sarebbe dedicato a facilitare le cose per chiunque avesse voluto seguirne l’esempio. Dopo aver giurato di voler costruire un sistema operativo libero e gratuito “oppure di morire provandoci – di vecchiaia, naturalmente”, ironizza Stallman, rassegnò le proprie dimissioni dallo staff del MIT nel gennaio 1984 per dedicarsi interamente al progetto GNU.

Quel gesto portò alla fine della protezione legale del MIT per ogni lavoro di Stallman. Tuttavia egli aveva ancora abbastanza amici al laboratorio da poter mantenere un proprio ufficio senza dover pagare alcun affitto. Riuscì anche ad assicurarsi qualche giro di consulenze per finanziare l’avvio del progetto GNU. Tuttavia quelle dimissioni significavano anche, da parte sua, la negazione di qualsiasi discussione sul conflitto d’interessi sul software o sulla proprietà dello stesso da parte del MIT. Colui che da giovane era caduto sempre più profondamente nelle braccia del laboratorio d’intelligenza artificiale a causa del proprio isolamento sociale, stava ora erigendo un muro legale tra se stesso e quell’ambiente.

Nei mesi immediatamente successivi, Stallman operò in isolamento anche rispetto alla comunità Unix. Nonostante l’annuncio apparso sul newsgroup net.unix-wizards avesse provocato risposte di sostegno, in questa fase iniziale furono pochi i volontari che decisero di aggregarsi alla crociata.

“La reazione della comunità si rivelò alquanto uniforme”, ricorda Rich Morin, allora leader di uno user-group dedicato a Unix. “La gente diceva: ‘Oh, davvero un’ottima idea. Facci vedere il codice. Mostraci che si può fare.”’

Seguendo la tipica tradizione hacker, Stallman iniziò a cercare programmi e strumenti già esistenti da inserire all’interno di GNU. Uno dei primi fu un compilatore chiamato VUCK, che convertiva i programmi scritti nel diffuso linguaggio C in un codice comprensibile per la macchina. Tradotto dall’olandese, l’acronimo del nome stava per Free University Compiler Kit. Ottimista, Stallman chiese all’autore se il programma fosse libero. Quando questi lo informò che “Free University” era solo il riferimento alla Vrije Universiteit di Amsterdam, Stallman rimase deluso.

“Mi rispose in tono canzonatorio, dicendo che l’università era libera ma non il compilatore”, rammenta Stallman. “Decisi allora che il mio primo programma per il progetto GNU sarebbe stato un compilatore valido per più linguaggi e piattaforme.”[20]

Alla fine Stallman trovò il compilatore Pastel[21] scritto dagli sviluppatori del Lawrence Livermore National Lab. Per quanto ne sapesse a quel tempo, tale compilatore era liberamente copiabile e modificabile. Purtroppo aveva un difetto sostanziale: salvava ogni programma nella memoria di base, rubando così spazio prezioso per altre attività. Sui sistemi mainframe il difetto poteva essere perdonabile, ma rappresentava una barriera insormontabile su macchine Unix, troppo piccole per poter gestire i grossi file in tal modo generati. Inizialmente Stallman compì rapidi progressi, grazie alla costruzione di un supporto compatibile con il linguaggio C. Tuttavia con l’arrivo dell’estate fu costretto a concludere che avrebbe dovuto scrivere un nuovo compilatore partendo completamente da zero.

Nel settembre 1984, Stallman mise momentaneamente da parte lo sviluppo del compilatore e iniziò a cercare qualche frutto a portata di mano. Cominciò a sviluppare una versione GNU di Emacs, il programma che aveva curato per oltre un decennio. Si trattava di una decisione strategica. All’interno della comunità Unix, esistevano due elaboratori testi originali: *vi*, scritto dal co-fondatore della Sun Microsystems Bill Joy, e *ed*, realizzato dal ricercatore dei

Bell Labs (e co-autore di Unix) Ken Thompson. Entrambi erano utili e diffusi, ma nessuno dei due offriva la possibilità di espandersi all'infinito propria dell'Emacs. Riscrivendo quest'ultimo per l'utenza Unix, Stallman avrebbe avuto maggiori probabilità di evidenziare le proprie abilità. Sembrava inoltre ragionevole ritenere che gli utenti dell'Emacs fossero in sintonia con la mentalità di Stallman.

Ripensando a quella decisione, Stallman non ci vide nulla di strategico. "Volevo un Emacs, ed avevo un'ottima opportunità per curarne lo sviluppo."

Ancora una volta l'idea di dover reinventare la ruota fa a pugni con la sensibilità dell'hacker efficiente. Lavorando alla versione Unix dell'Emacs, Stallman si trovò presto a seguire le orme di James Gosling, laureando presso la Carnegie Mellon nonché autore di una versione in C chiamata Gosling Emacs o GOSMACS. Questa includeva un interprete basato su un derivato semplificato del linguaggio Lisp noto come MOCKLISP. Deciso ad usufruire di una struttura analoga, Stallman prese copiosamente in prestito le innovazioni di Gosling. Nonostante il GOSMACS fosse sotto copyright e l'autore ne avesse venduto i diritti alla UniPress, azienda privata di software, Stallman prese per buone le assicurazioni fornitegli da un collega che aveva partecipato alle prime lavorazioni di MOCKLISP. Secondo quest'ultimo, Gosling, all'epoca prossimo al dottorato presso la Carnegie Mellon, aveva assicurato i suoi collaboratori che il loro lavoro sarebbe rimasto liberamente accessibile all'esterno. Ma quando la UniPress venne a conoscenza del progetto di Stallman minacciò di imporre il proprio copyright. Ancora un volta Stallman si trovava di fronte alla prospettiva di dover ricominciare tutto da capo.

Mentre lavorava alla ricostruzione dell'interprete di Gosling, Stallman finì col crearne uno nuovo perfettamente funzionale, mettendo così in discussione l'uso di quello originale. Tuttavia, il concetto per cui gli sviluppatori potessero vendere i diritti sul software lo faceva soffrire – o meglio, era soprattutto l'idea secondo cui uno sviluppatore fosse proprietario di diritti da poter rivendere agli altri. In un intervento del 1986 presso lo Swedish Royal

Technical Institute, citò l'incidente della UniPress come ulteriore esempio dei pericoli associati con il software proprietario.

"Talvolta credo che una delle cose migliori che potrei fare in vita mia sarebbe quella di trovare una quantità enorme di software proprietario protetto da segreto commerciale, e iniziare a distribuirne copie agli angoli delle strade, così da infrangere tale segreto", disse Stallman. "Forse questa sarebbe una maniera molto più efficace per offrire alla gente nuovi programmi di software libero, anziché mettermi a scriverli; ma avrebbero tutti troppa paura anche soltanto a prendere quelle copie."[22]

Nonostante l'inevitabile tensione, a lungo andare la disputa sulle innovazioni di Gosling si sarebbe rivelata d'aiuto sia per Stallman sia per il movimento del software libero. Avrebbe intanto costretto il primo a considerare le debolezze della comune dell'Emacs e del sistema di fiducia informale che aveva consentito l'emergere di diramazioni problematiche. Lo avrebbe inoltre obbligato a focalizzare meglio gli obiettivi politici del movimento del software libero. Nel 1985, dopo l'uscita di GNU Emacs, Stallman redasse "Il Manifesto GNU", versione ampliata dell'annuncio originale apparso nel settembre 1983. Il documento conteneva una lunga sezione dedicata alle svariate argomentazioni portate da programmatori commerciali e accademici a giustificazione della proliferazione del software proprietario. Una di queste tesi – "i programmatori non meritano forse una ricompensa per la loro creatività?" – ottenne una risposta che rifletteva la collera di Stallman verso il recente episodio sull'Emacs di Gosling:

"Se c'è qualcosa che merita una ricompensa, questo è il contributo sociale", scrisse Stallman. "La creatività può ritenersi un contributo sociale ma soltanto finché la società sia libera di usarne i risultati. Se i programmatori meritano di essere ricompensati per la creazione di programmi innovativi, secondo lo stesso criterio meritano di essere puniti quando invece ne limitano l'utilizzo."[23]

Con la diffusione del GNU Emacs, finalmente il progetto poteva mostrare del codice ben fatto. Ciò diede inoltre avvio alla tipica attività imprenditoriale basata sul software. Man mano che aumentavano gli sviluppatori Unix che

iniziavano a giocare con il software, ecco arrivare regali, offerte in denaro e richieste di ulteriori copie su nastro. Ad occuparsi del lato commerciale del progetto GNU Stallman chiamò qualche collega, formalizzando così la nascita della Free Software Foundation (FSF), organizzazione nonprofit dedita a velocizzare il raggiungimento degli obiettivi del progetto GNU. Con Stallman in funzione di presidente e vari amici hacker nel direttivo, la FSF contribuì a fornire la necessaria professionalità al progetto GNU.

Robert Chassell, allora programmatore presso la Lisp Machines, Inc., divenne uno dei cinque membri del direttivo dopo una conversazione a cena con Stallman. Chassell operava anche in quanto tesoriere, ruolo inizialmente modesto, ma cresciuto in tutta rapidità.

“Credo che nel 1985 il volume totale tra uscite ed entrate fosse intorno ai 23.000 dollari”, ricorda Chassell. “Richard aveva il suo ufficio, per il resto ci arrangiavamo. Misi tutta la roba, soprattutto i nastri, sotto la mia scrivania. Soltanto più tardi la LMI ci affittò qualche spazio dove riporre nastri e altro materiale necessario.”

Oltre a garantire maggiore rispettabilità all’esterno, la Free Software Foundation si pose come centro di gravità per altri programmatori insoddisfatti. Il mercato Unix, ristretto all’ambito accademico anche all’epoca dell’iniziale annuncio di Stallman per l’avvio di GNU, andava diventando sempre più competitivo. Per tenersi stretti i propri clienti, le aziende stavano chiudendo l’accesso al codice sorgente, tendenza a cui era dovuto il rapido incremento di richieste per il software del progetto GNU. I maghi di Unix che un tempo consideravano Stallman un eccentrico un po’ turbolento, iniziarono a considerarlo una Cassandra del software.

“La gente non se ne rende conto fino a quando non viene coinvolta direttamente, ma è davvero frustrante lavorare per due anni su un programma solo per poi vederselo soffiar via sotto il naso”, così Chassell sintetizza le sensazioni e i commenti contenuti nelle lettere ricevute in quegli anni dalla FSF. “Dopo che ti accade un paio di volte, cominci a dirti, ‘Un momento, così non funziona’.”

Per Chassell la decisione di entrare a far parte della Free Software Foundation si basava su analoghe esperienze personali. Prima di lavorare alla LMI, stava scrivendo un libro introduttivo su Unix per la Cadmus, Inc., azienda di software situata nei pressi di Cambridge. Quando questa chiuse bottega, portò con sé i diritti dell'opera, e il tentativo dello stesso Chassell di riacquistarli non andò in porto.

"A quanto mi risulta, il libro è tuttora in qualche cassetto ad ammuffire, inutilizzabile e non copiabile, semplicemente fuori dal giro", aggiunge. "Era una buona opera introduttiva, se posso dirlo. Basterebbero forse tre o quattro mesi per trasformarlo oggi in un'ottima guida all'odierno GNU/Linux. A parte quanto posso ricordare personalmente, l'intera esperienza è andata perduta."

Costretto a vedere il proprio lavoro impantanarsi mentre il temporaneo datore di lavoro rischiava di finire in bancarotta, Chassell afferma di aver provato qualcosa di analogo alla rabbia che spinse Stallman a due passi da un attacco apoplettico. "Per me la questione centrale stava nella certezza che se vuoi vivere in maniera decente non puoi trovarti davanti a delle porte chiuse", sostiene Chassell. "L'idea stessa di avere la libertà di poter risolvere qualcosa, di modificarla, di qualunque faccenda si tratti, è veramente importante. Ti fa pensare con felicità che, dopo aver vissuto così per un po' di tempo, ciò che hai fatto acquista davvero validità. Perché altrimenti ti viene semplicemente portato via e buttato o abbandonato, oppure quantomeno non hai più alcun rapporto con la tua opera. È come perdere un pezzo della propria esistenza."

[1] Si veda Richard Stallman, "Initial GNU Announcement" (settembre 1983). http://www.gnu.ai.mit.edu/gnu/initial-announcement.html

[2] Ibid.

[3] Ibid.

[4] Si veda Marshall Kirk McKusick, "Vent'anni di Unix a Berkeley" in *Open Sources*, Apogeo, 1999, p. 31.

[5] Si veda Richard Stallman (1986).

[6] Ibid.

[7] Fonti diverse: si veda l'intervista a Richard Stallman, la e-mail di Gerald Sussman, e Jargon File 3.0.0. http://www.clueless.com/jargon3.0.0/TWENEX.html

[8] Si veda http://www.as.cmu.edu/~geek/humor/See_Figure_1.txt

[9] Si veda Richard Stallman (1986).

[10] Si veda “MIT AI Lab Tourist Policy.” http://catalog.com/hopkins/text/tourist-policy.html

[11] Si veda Richard Stallman (1986).

[12] Ibid.

[13] Si veda H. P. Newquist, *The Brain Makers: Genius, Ego, and Greed in the Quest for Machines that Think*, Sams Publishing, 1994, p.172.

[14] Ibid. p. 196.

[15] Ibid. Scrive Newquist, secondo cui quest’aneddoto venne confermato da diversi dirigenti della Symbolics: “Il messaggio provocò una breve folata di eccitazione e speculazione da parte degli impiegati della Symbolics, ma alla fin fine nessuno prese veramente sul serio quella battuta di Stallman.”

[16] Si veda “MIT AI Lab Tourist Policy.” http://catalog.com/hopkins/text/tourist-policy.html

[17] Si veda Steven Levy, *Hackers*, Penguin, USA, 1984 p. 426.

[18] Si veda Bill Gates, “An Open Letter to Hobbyists” (3 febbraio 1976). Copia della lettera è disponibile online: http://www.blinkenlights.com/classiccmp/gateswhine.html.

[19] Si veda Richard Stallman, “Il progetto GNU” in *Open Sources*, Apogeo, p. 60. Stallman aggiunge una propria nota a margine di quella dichiarazione: “Essendo ateo, non seguo alcuna guida religiosa, ma a volte mi trovo ad ammirare qualcosa che qualcuno di loro ha detto.”

[20] Richard Stallman (1986)

[21] Compilatore multipiattaforma per una versione estesa del Pascal. [*N.d.R.*]

[22] Richard Stallman (1986).

[23] Si veda Richard Stallman, “Il Manifesto GNU” (1985). http://www.gnu.org/gnu-/manifesto.it.html

Sant’Ignucius

Il Maui High Performance Computing Center delle Hawaii si trova all’interno di un edificio a un piano situato nelle polverose colline rosse appena sopra la città di Kihei.

Circondato da panorami milionari e dal Silversword Golf Course che vale milioni di dollari, il centro è un posto da sogno per ogni ricercatore. Completamente diverso dai contorni sterili, squadrati di Tech Square o anche dagli agglomerati che ospitano i laboratori di ricerca e che si estendono disordinatamente, come Argonne, Illinois, e Los Alamos, New Mexico, il centro sembra quel tipo di posto dove i ricercatori trascorrono più tempo stesi al sole che a seguire i progetti del post-dottorato.

Un’ipotesi vera soltanto in parte. Nonostante essi tendano ad approfittare delle opportunità ricreative in loco, al contempo prendono seriamente il proprio lavoro. Secondo Top500.org, sito web che segue lo sviluppo dei supercomputer più potenti del mondo, l’IBM SP Power3 ospitato nel Computing Center è in grado di eseguire 837 miliardi di operazioni al secondo, affermandosi come uno dei 25 computer più potenti attivi sul pianeta. In comproprietà tra la University of Hawaii e la U.S. Air Force, la macchina divide i propri cicli operativi tra i compiti relativi alla logistica militare e le indagini fisiche sulle alte temperature.

In poche parole, l’High Performance Computing Center di Maui è un luogo unico, un posto dove l’intelligente cultura della scienza e dell’ingegneria si mescola pacificamente con l’atteggiamento rilassato delle isole hawaiiane. Come sintetizza felicemente uno slogan apparso nel 2000 sul sito web del laboratorio locale: “L’informatica in paradiso”.

Non è esattamente il luogo in cui ci si aspetterebbe di incontrare Richard Stallman, il quale di fronte allo stupendo panorama del vicino Maui Channel, dalla finestra di un ufficio dell'edificio, borbotta con fare critico: "Troppo sole". Comunque sia, nei panni dell'emissario di un altro paradiso informatico, Stallman è venuto a portare il proprio messaggio, anche se ciò comporta l'esposizione della sua carnagione pallida da hacker ai pericoli del sole tropicale.

La sala della conferenza è già metà piena quando arrivo per seguire il suo intervento. Il rapporto tra uomini e donne è leggermente migliore dell'evento di New York, ma neppure troppo: 85% contro 15%. Quasi metà dei presenti indossa pantaloni color kaki e magliette da golf griffate. L'altra metà sembra seguire l'usanza locale: le magliette a fiori così diffuse in questa parte del mondo e i volti abbronzati color ocra. Un pubblico multiforme accomunato comunque dai vari gadget elettronici: telefoni cellulari Nokia, Palm Pilot, portatili Sony VAIO.

Inutile aggiungere come Stallman sembri del tutto fuori luogo, in piedi in fondo alla sala con indosso una maglietta blu, dei pantaloni larghi marroni e calzini bianchi. Le luci al neon della sala ne evidenziano il colore poco salutare della pelle assetata di sole. Barba e capelli sono abbastanza lunghi da far apparire gocce di sudore perfino sul più fresco dei colli hawaiiani. Gli manca solo un tatuaggio sulla fronte con una parola tipo "continentale", e Stallman potrebbe essere scambiato per un marziano.

Mentre Stallman gironzola intorno, alcune persone in platea con la maglietta del Maui FreeBSD Users Group (MFUG) sono indaffarate a sistemare le apparecchiature audio e video. FreeBSD, un software libero frutto della Berkeley Software Distribution, la venerabile versione accademica di Unix sviluppata negli anni '70, tecnicamente si pone come rivale del sistema operativo GNU/Linux. Eppure nel mondo degli hacker gli interventi di Stallman vengono documentati con un fervore analogo a quello riservato ai Grateful Dead e alla leggendaria truppa di archivisti amatoriali. Stavolta spetta agli aderenti del Maui Free-BSD Users Group assicurarsi che i colleghi di

Amburgo, Mombai e Novosibirsk non perdano le ultime perle di saggezza elargite da RMS.

L'analogia con i Grateful Dead è assolutamente pertinente. Nel descrivere le opportunità commerciali inerenti al modello del software libero, non di rado Stallman si è riferito a quell'esempio. Rifiutando di limitare le registrazioni amatoriali dei concerti dal vivo, i Grateful Dead sono diventati molto più di un gruppo rock. Sono divenuti il centro di una comunità tribale dedicata alla loro musica. Con il passare del tempo tale comunità ha assunto proporzioni talmente ampie, confermando al contempo la propria devozione, che il gruppo ha rinunciato ai contratti con le case discografiche per affidarsi unicamente alle entrate dei concerti e dei tour dal vivo. Nel 1994, ultimo anno delle apparizioni sul palco, i Grateful Dead avevano incassato 52 milioni di dollari soltanto con i biglietti venduti ai concerti.[1]

Pur essendo ben poche le aziende di software capaci di imitare simili successi, l'aspetto tribale della comunità del software libero rappresenta un motivo per cui nella seconda metà degli anni '90 molta gente ha iniziato a considerare positiva l'idea di pubblicare i codici sorgenti. Nella speranza di attivare una propria schiera di fedeli appassionati, società come IBM, Sun Microsystems e Hewlett-Packard sembrano aver adottato la lettera, se non esattamente lo spirito, del messaggio di Stallman. Nel definire la GPL come la "Magna Carta" dell'industria tecnologica, il giornalista specializzato di ZDNet, Evan Leibovitch, considera la crescente affezione per tutto ciò che è targato GNU qualcosa di più di una semplice tendenza. "Si tratta di un mutamento sociale che consente agli utenti di riprendere in mano il controllo sul proprio futuro", scrive Leibovitch. "Come la Magna Carta riconosceva i diritti dei cittadini britannici, così la GPL sostiene i diritti e le libertà dei consumatori nell'interesse degli stessi utenti del software."[2]

L'aspetto tribale della comunità del software libero consente inoltre di spiegare perché mai una quarantina di programmatori, che altrimenti se ne starebbero a lavorare su progetti di fisica oppure a navigare sul web per sapere quando arriverà il vento buono per il surf, si trovino invece pigiati in una sala-conferenze ad ascoltare Stallman.

Contrariamente a quanto accaduto a New York, stavolta non c'è nessuno a presentarlo e neppure lui si presenta. Non appena il gruppo del FreeBSD ha finito di approntare le apparecchiature, Stallman si limita a farsi avanti e inizia a parlare, con la voce che copre ogni altro brusio in sala.

"Quando si affronta la questione delle regole concernenti l'utilizzo del software, per lo più sono le stesse aziende a occuparsene, e quindi vedono la cosa dal punto di vista della propria utilità", così Stallman apre l'intervento. "Questa la domanda che sembrano porsi: quali norme possiamo imporre agli altri in modo che debbano pagarci un sacco di soldi? Negli anni '70 ho avuto la buona sorte di far parte di una comunità di programmatori fondata sulla condivisione del software. E grazie a ciò mi è sempre piaciuto considerare la medesima questione da un punto di vista diverso, chiedendomi: quali sono le regole necessarie per rendere la società più giusta e positiva per quanti ne fanno parte? E quindi sono arrivato a conclusioni del tutto opposte."

Ancora una volta Stallman propone rapidamente la parabola della stampante laser Xerox, concedendosi un attimo di pausa prima di ripetere il gesto drammatico di puntare il dito contro qualcuno del pubblico. Passa poi a dedicare un minuto o due alla spiegazione del nome GNU/Linux.

"Qualcuno potrebbe dire, 'Perché mai affaticarsi tanto per guadagnare consensi per questo sistema? In fondo l'importante è che si sia raggiunto lo scopo prefissato, non tanto chi ne abbia il merito.' Be', se questo fosse vero si tratterebbe di un saggio consiglio. In realtà, il progetto non riguarda la realizzazione di un sistema operativo, quanto piuttosto la diffusione dell'idea stessa di libertà per gli utenti informatici. E per riuscirci dobbiamo permettere qualsiasi utilizzo del computer in piena libertà."[3]

E aggiunge: "C'è ancora molto lavoro da fare".

Per qualcuno tra il pubblico si tratta di materiale datato, per altri suona un po' arcano. Quando una persona del contingente in maglietta da golf sembra appisolarsi, Stallman si ferma e chiede al vicino di svegliarlo.

"Una volta qualcuno ha trovato la mia voce così suadente che mi ha chiesto se per caso non fossi un guaritore", dice Stallman, suscitando le risate del pubblico. "Temo che questo significhi che posso aiutarvi a sprofondare

gentilmente in un rilassante pisolino. E forse qualcuno ne ha davvero bisogno. E quindi non solleverò obiezioni. Se qualcuno vuole dormire, non faccia complimenti."

Il discorso si chiude con una breve analisi dei brevetti sul software, questione che suscita sempre maggiore preoccupazione sia nella comunità del software libero sia in ambito industriale. Come nel caso di Napster, tali brevetti riflettono l'ambigua situazione di dover applicare leggi e concetti del mondo fisico all'universo incorporeo delle tecnologie dell'informazione. La differenza tra la tutela di un programma sotto copyright e quella di uno coperto da brevetto è sottile ma significativa. Nel primo caso, chi ha creato il software può vietare la duplicazione del codice ma non quella dell'idea o della funzionalità cui si riferisce quel codice. In altri termini, se uno sviluppatore sceglie di non usare un programma tutelato dai termini imposti dall'autore originale, gli resta comunque la libertà di operare il cosiddetto "reverse-engineering" – ovvero, riprodurne le funzionalità scrivendo nuovamente il codice partendo da zero. Si tratta di una duplicazione delle idee assai comune nell'industria del software commerciale, dove spesso le aziende isolano i gruppi a cui affidano tali operazioni, per evitare l'accusa di spionaggio aziendale o possibili scambi tra gli sviluppatori. Nel gergo dell'odierno sviluppo informatico, ci si riferisce a questa tecnica col termine di ingegneria in "locali senza polvere."

I brevetti sul software operano in maniera diversa. Secondo l'omonimo ufficio statunitense, aziende e individui possono brevettare ogni nuovo algoritmo purché lo sottopongano a una revisione pubblica. In teoria, ciò consente al possessore di quel brevetto di trattare la diffusione dell'invenzione con un monopolio limitato a 20 anni dalla data di presentazione della richiesta. In pratica, tale diffusione riveste un valore minimo, poiché spesso l'operatività del programma si spiega da sé. Contrariamente al copyright, un brevetto offre a chi lo detiene la possibilità di impedire lo sviluppo indipendente di un programma dotato di funzionalità identiche o analoghe.

Nell'industria del software, dove 20 anni possono arrivare a coprire l'intero arco di vita di un mercato, i brevetti assumono connotati strategici. Laddove

aziende quali la Microsoft e la Apple una volta si facevano guerra sul copyright e sull'aspetto tipico di varie tecnologie, le società odierne su Internet ricorrono ai brevetti, poiché sono uno strumento atto a bloccare applicazioni individuali e modelli commerciali, come rivela il noto caso di Amazon.com, che nel 2000 tentò di brevettare il procedimento per effettuare acquisti rapidi online denominato "one-click". Ma per la maggior parte delle società i brevetti sul software sono divenuti uno strumento difensivo, dove gli scambi di licenze servono a pareggiare la quantità di brevetti posseduti da una corporation contro quelli di un'altra. Tuttavia in alcuni importanti casi riguardanti la crittografia e gli algoritmi per immagini grafiche, alcuni produttori di software sono riusciti a bloccare determinate tecnologie concorrenti.

Per Stallman, la questione dei brevetti sul software serve a mettere in evidenza la necessità di un'eterna vigilanza da parte degli hacker. Sottolinea inoltre l'importanza di insistere sui benefici politici dei programmi di software libero rispetto ai vantaggi competitivi. Riferendosi alla capacità dei brevetti di creare zone protette all'interno del mercato, Stallman ritiene che la competizione nelle prestazioni e nei prezzi, due aree in cui i sistemi operativi di software libero quali GNU/Linux e FreeBSD già vantano un discreto vantaggio sulle controparti proprietarie, sia un problema di importanza secondaria rispetto alle questioni più ampie concernenti la libertà degli sviluppatori e degli utenti.

"Non ci manca certo il talento necessario a produrre software migliore", sostiene Stallman. "Ma non abbiamo il diritto di intervenire. Qualcuno ci ha vietato di offrire servizi al pubblico. E allora, che cosa succede quando gli utenti s'imbattono in vuoti di questo tipo nel software libero? Se sono stati convinti dal movimento open source che queste libertà sono positive poiché danno come risultato un software più potente e affidabile, probabilmente diranno, 'Non avete mantenuto la promessa. Questo programma non è affatto più potente, gli manca una certa funzione. Mi avete mentito'. Se invece concordano con il movimento del software libero sul fatto che la libertà è importante per se stessa, allora diranno, 'Come si permette questa gente di

impedirmi di usufruire di questa funzione, e con essa della libertà?’. Grazie a questo tipo di risposta forse riusciremo a sopravvivere alle bordate che ci attendono non appena esploderà la questione dei brevetti.”

Questi commenti hanno effetti piuttosto articolati. Quando si parla di brevetti la maggior parte degli esponenti open source si mostra altrettanto, se non più decisa di Stallman. Eppure la logica sottostante la sua tesi – l’accento che i sostenitori dell’open source pongono sui vantaggi utilitaristici del software libero rispetto a quelli politici – rimane immune da contestazioni. Piuttosto che sottolineare la rilevanza politica di tali programmi, gli esponenti open source hanno scelto di evidenziare l’integrità ingegneristica del modello di sviluppo messo a punto dagli hacker. Citando la forza delle revisioni collaborative, la tesi dell’open source sostiene che programmi come GNU/Linux oppure FreeBSD siano costruiti e verificati in maniera più accurata e di conseguenza risultino più affidabili per l’utente medio.

Ciò non implica la mancanza di valenze politiche per il termine “open source”, il quale mira sostanzialmente a due obiettivi. Primo, elimina la confusione associata alla parola “free”, che in inglese ha il doppio significato di “libero” e “gratuito”, tant’è vero che molti imprenditori tendono a interpretarla nel senso di “costo zero”. Secondo, consente alle aziende di prendere in esame il fenomeno del software libero in un contesto tecnologico anziché etico. Eric Raymond, cofondatore della Open Source Initiative e uno dei primi hacker ad abbracciare quel termine, riassume efficacemente la frustrazione di dover seguire Stallman sul terreno politico in un saggio del 1999 intitolato “Shut Up and Show Them the Code” (Smetti di parlare e fagli vedere il codice):

> La retorica di RMS è molto seducente per gente come noi. Noi hacker ci riteniamo pensatori e idealisti sempre aperti al richiamo dei “principi”, della “libertà”, dei “diritti”. Anche quando ci troviamo in disaccordo sui bit di qualche suo programma, siamo convinti che lo stile retorico di RMS debba funzionare comunque; rimaniamo sconcertati e increduli quando questo non si verifica con il 95% di quanti non sono “wired” come noi.[4]

Questo 95%, scrive Raymond, comprende la vasta maggioranza dei manager, degli investitori, degli utenti non-hacker cui spettano, non foss’altro

per il peso dei numeri, le decisioni sull'andamento generale del mercato commerciale. Se non si trova il modo di convicerli, insiste Raymond, i programmatori sono condannati a perseguire la propria ideologia ai margini della società:

> Quando RMS insiste sul fatto che dovremmo parlare dei "diritti degli utenti", provoca un invito pericolosamente attraente a ripetere i fallimenti del passato. Dobbiamo respingerlo – non perché siano sbagliati i principi di base, ma perché quel tipo di linguaggio, applicato al software, semplicemente non riesce a convincere nessun altro tranne noi. Finisce anzi col confondere e allontana molta gente che non appartiene alla nostra cultura.[5]

Ascoltando Stallman in persona enunciare il proprio credo politico, sembra difficile notare qualcosa di poco chiaro o di fastidioso. Il suo aspetto può apparire poco attraente, ma il messaggio è del tutto logico. Quando una persona tra il pubblico chiede se, nel rifiutare il software proprietario, coloro che propongono il software libero finiscano per perdere la capacità di stare al passo con i più recenti sviluppi tecnologici, Stallman replica in modo coerente alle proprie opinioni. "Credo che la libertà sia più importante del puro avanzamento tecnico", dice. "Sceglierei sempre un programma libero meno aggiornato piuttosto che uno non-libero più recente, perché non voglio rinunciare alla libertà personale. La mia regola è, se non posso condividerlo con gli altri, allora non lo prendo."

Simili risposte, tuttavia, non fanno altro che rinforzare la natura quasi religiosa di quel messaggio. Come un ebreo con i cibi kosher o un mormone che rifiuta di bere alcolici, Stallman dipinge la propria decisione di usare il software libero in luogo di quello proprietario con i colori della tradizione e del credo personale. Come si usa fra gli evangelisti del software, Stallman evita però di forzare i presenti ad accettare le sue opinioni a ogni costo. In ogni caso, è raro il caso in cui, dopo averlo ascoltato, qualcuno se ne vada senza sapere quale sia la vera strada che conduce alla legittimità del software.

Per meglio enfatizzare il messaggio, Stallman inframmezza il discorso con un rituale insolito. A un certo punto tira fuori, da una busta di plastica, una tunica scura e la indossa. Da un'altra busta prende il disco rigido di un

computer, giallo e riflettente, e se lo mette in testa. Dal pubblico iniziano a levarsi le prima risate.

“Io sono Sant’Ignucius della Chiesa dell’Emacs”, recita alzando la mano destra come a dispensare la benedizione. “Sono qui a benedire i vostri computer, figli miei.”

Dopo pochi secondi le risate si trasformano in uno scrosciante applauso. E mentre il pubblico continua ad applaudire, il disco rigido sulla testa assorbe i riflessi della luce sovrastante, creando il preciso effetto di un’aureola. In attimo eccolo trasformarsi da strano personaggio a icona religiosa russa.

“All’inizio l’Emacs non era altro che un elaboratore testi”, così Stallman sintetizza l’evoluzione del programma. “Alla fine divenne un modello di vita per molti e una religione per alcuni. Una religione che definiamo la Chiesa dell’Emacs.”

La sceneggiata rappresenta uno spensierato momento di autoironia, un’umoristica stoccata di rimando alle molte persone che considerano l’ascetismo del software propugnato da Stallman null’altro che un camuffamento del suo fanatismo religioso. Ma è anche un modo per mostrarsi senza difese, apertamente. È come se, con addosso quel travestimento di tunica e aureola, abbia finalmente deciso di concedere qualche libertà ai presenti, proponendo loro: “Potete ridere di me quanto volete. So bene di essere un tipo strano”.



Figura 8.1 Stallman nei panni di Sant'Ignucius. Fotografia di Wouter van Oortmerssen.

Parlando in seguito del personaggio di Sant'Ignucius, Stallman spiega di averlo ideato nel 1996, parecchio tempo dopo la creazione dell'Emacs ma sicuramente prima dell'apparizione del termine "open source" e della susseguente lotta per la leadership nella comunità hacker. Al contempo, sottolinea, voleva trovare un modo per "prendersi in giro", così da ricordare al pubblico che, per quanto testardo, non era poi quel fanatico dipinto da qualcuno. Fu soltanto dopo, aggiunge Stallman, che gli altri si appropriarono di quell'alter ego per danneggiare la sua reputazione personale di ideologo del software, come testimonia un'intervista di Eric Raymond apparsa nel 1999 sul sito web linux.com:

> Quando sostengo che RMS calibri con molta attenzione le proprie azioni, non intendo disprezzarlo né accusarlo di poca sincerità. Voglio dire che, al pari di ogni efficace comunicatore, possiede un buon guizzo teatrale. Talvolta ne è consapevole – l'avete visto nei panni di Sant'Ignucius, benedire il software con un disco in testa? Ma per lo più è qualcosa di inconsapevole; ha imparato a stimolare una certa irritazione che funziona, che tiene inchiodata l'attenzione della gente senza (generalmente) confonderla troppo.[6]

Stallman rifiuta l’analisi di Raymond. “È soltanto il mio modo di prendermi in giro”, risponde. “Il fatto che altri lo giudichino in maniera diversa è solo un riflesso delle loro posizioni, non delle mie.”

Ciò detto, Stallman non nega di sentirsi un attore mancato. “Scherzi?”, fa a un certo punto. “Amo trovarmi al centro dell’attenzione.” Per dare una spinta a questo processo, mi rivela che una volta si era iscritto a un’associazione la cui attività puntava a esaltare la capacità di parlare in pubblico, pratica che raccomanda caldamente a chiunque. Possiede un senso della propria presenza sul palco che farebbe invidia a molti attori teatrali e rimanda ai varietà vaudevilliani del passato. Qualche giorno dopo il discorso al Maui High Performance Computing Center, gli rammento il suo intervento al LinuxWorld 1999 per chiedergli se non sia per caso affetto dal complesso di Groucho Marx – ovvero, il rifiuto di appartenere a qualsiasi gruppo che lo vorrebbe invece tra i propri aderenti. La replica di Stallman è immediata: “No, ma ammiro parecchio Groucho Marx e sicuramente alcune sue uscite mi hanno influenzato parecchio. Ma ho trovato fonte d’ispirazione anche in Harpo Marx”.

L’influenza di Groucho Marx appare evidente nella sua costante ricerca della battuta giusta. E comunque, battute e giochi di parole sono un tratto comune agli hacker. Ma forse l’aspetto di Stallman più vicino a Groucho sta nel modo impassibile con cui lancia le sue frecciatine. In genere queste arrivano così furtive e improvvise – senza neppure un’alzata di sopracciglia o l’accenno di un sorriso – che viene da chiedersi se non sia Stallman a prendersi gioco del pubblico anziché viceversa.

Le risate suscitate dalla parodia di Sant’Ignucius al Maui High Performance Computer Center fanno svanire simili preoccupazioni. Pur senza imporsi come prim’attore, Stallman possiede sicuramente le doti per tenere felicemente testa agli ingegneri assiepati in quella sala. “Per diventare santi della Chiesa dell’Emacs non occorre coltivare il celibato, ma bisogna condurre un’esistenza basata sulla purezza morale”, spiega ai presenti. “Dovete esorcizzare il malefico sistema operativo proprietario dai vostri computer per poi installarne uno completamente libero. E su quello dovrete

poi far girare soltanto software libero. Se sarete in grado di aderire a questi precetti, allora anche voi diventerete santi della Chiesa dell'Emacs, e potrete perfino guadagnarvi un'aureola."

La scena di Sant'Ignucius si chiude con una rapida battuta per addetti ai lavori. Su gran parte dei sistemi Unix e annesse diramazioni, il maggior rivale dell'Emacs è *vi*, un elaboratore di testi sviluppato dall'ex-studente alla University of Berkeley (California) e attuale responsabile della ricerca presso la Sun Microsystems, Bill Joy. Prima di togliersi l'"aureola", Stallman scherza sul programma rivale. "Talvolta qualcuno mi chiede se nella Chiesa dell'Emacs non sia considerato peccato ricorrere al *vi*", dice. "Usarne una versione libera non è peccato, ma è una penitenza. Happy hacking a tutti."

Dopo una breve sessione dedicata a domande e risposte, parte del pubblico gli si stringe intorno. Qualcuno gli chiede un autografo. "Firmo su questo foglio", dice allungando a una donna la stampa della GNU General Public License, "ma soltanto se mi prometti di usare sempre il termine GNU/Linux invece di Linux e chiedi ai tuoi amici di fare altrettanto."

Il commento non fa altro che confermare un'osservazione del tutto personale. Al contrario di altre figure teatrali e politiche, Stallman non riesce mai a "spegnersi". A parte il personaggio di Sant'Ignucius, le posizioni filosofiche estrinsecate sul palco rimangono tali e quali anche dietro le quinte. Più tardi quella stessa sera, durante una chiacchierata a cena, un programmatore manifesta il suo apprezzamento per i programmi "open source"; Stallman alza subito il tono pur tra un boccone e l'altro: "Vuoi dire software libero, è questa la definizione giusta".

Tornando per un attimo alla parte della conferenza riservata alle domande conclusive, Stallman ammette talvolta di giocare a fare il pedagogo. "Alcuni dicono, 'Invitiamo prima la gente nella comunità, e poi insegneremo loro che cosa significa libertà'. Potrebbe anche essere una strategia ragionevole, ma poi succede che quasi tutti ci diamo un gran daffare a invitare gli altri, e alla fine quasi nessuno dice nulla sulla libertà."

Il risultato, aggiunge Stallman, fa pensare a una città del terzo mondo. La gente vi si ammassa nella speranza di far soldi o quantomeno di partecipare a

una cultura ricca e vibrante, tuttavia chi detiene veramente il potere impiega nuovi stratagemmi – ad esempio, i brevetti sul software – per tenere lontane le masse. “Arrivano a milioni e finiscono per insediarsi in quartieri-ghetto, ma nessuno s’impegna per il passo successivo: farli uscire da quei ghetti. Se ritenete che parlare del software libero sia una buona strategia, allora prodighiamoci per la fase successiva. Parecchie persone lavorano per arrivare sul primo scalino, ne servono assai di più per raggiungere il secondo.”

Ciò significa avere ben chiaro che è la libertà, e non l’accettazione, a costituire le fondamenta del movimento del software libero. Quanti sperano di riformare dall’interno l’industria del software proprietario si prendono in giro da soli. “È rischioso cercare il cambiamento dall’interno”, sostiene Stallman. “A meno che non si agisca al livello di un Gorbaciov, finiranno per neutralizzarci.”

Molte le mani alzate in platea. Stallman indica una persona tra quelle con la maglietta da golf. “Senza brevetti, come credi si possa limitare lo spionaggio commerciale?”

“Credo proprio che le due questioni non abbiano nulla in comune”, ribatte Stallman.

“Mi riferisco alla possibilità che qualcuno decida di rubare il software di un’altra azienda.”

Stallman si contorce come se fosse stato colpito da un spray velenoso. “Un momento. Rubare? Mi dispiace, ma c’è tanto pregiudizio in quest’affermazione che non posso far altro che respingerla. Le aziende che sviluppano software non-libero e cose simili detengono un’enormità di segreti commerciali, si tratta di uno scenario che difficilmente potrà cambiare. In passato – diciamo anche negli anni ‘80 – la maggior parte dei programmatori non sapeva neanche dell’esistenza dei brevetti e non vi prestava alcuna attenzione. Succedeva che la gente faceva girare delle idee interessanti e, se non faceva parte del movimento del software libero, ne teneva segreti i dettagli. Adesso invece pongono sotto brevetto quelle stesse idee, e continuano a tenerne segreti i dettagli. Insomma, dalla tua descrizione non direi che i brevetti possano cambiare la situazione in un modo o nell’altro.”

“Ma se ciò non ha conseguenze sulla pubblica circolazione delle idee…”, s’inserisce un altro, la voce tremula non appena inizia a parlare.

“Invece lo ha”, replica Stallman. “È tale circolazione a informarci del fatto che quell’idea rimane intoccabile per il resto della comunità per 20 anni. E che cosa c’è di buono in questo? E poi il linguaggio dei brevetti è così difficile da interpretare, allo scopo sia di offuscare la portata dell’idea stessa sia di renderla più ampia possibile, che è praticamente inutile studiare le informazioni pubblicate per cercare di imparare qualcosa. L’unico motivo per dare un’occhiata ai brevetti è per aggiornarsi su quel che non si può fare.”

Il pubblico rimane in silenzio. L’intervento, iniziato alle 3:15 del pomeriggio, si avvicina al fischio di chiusura previsto per le cinque, e gran parte dei presenti si muove sulla sedia già pregustando il weekend alle porte. Cosciente della stanchezza in sala, Stallman dà un’occhiata intorno e rapidamente chiude l’incontro. “Direi allora che siamo arrivati alla fine”, conclude, e aggiunge come un battitore d’aste “aggiudicato” per prevenire le domande dell’ultimo istante. Visto che nessuno alza più la mano, Stallman saluta con la tradizionale battuta finale: “Happy hacking”.

[1] Si veda “Grateful Dead Time Capsule: 1985-1995 North American Tour Grosses”. http://www.accessplace.com/gdtc/1197.htm

[2] Si veda Evan Leibovitch, “Who’s Afraid of Big Bad Wolves”, *ZDNet Tech Update* (15 dicembre 2000). http://techupdate.zdnet.com/techupdate/stories/main-/0,14179,2664992,00.html

[3] Per non appesantire la narrazione, ho esitato ad approfondire qui la questione della definizione offerta da Stallman sulla “libertà” nel caso del software. Il sito web del progetto GNU ne elenca quattro componenti fondamentali:

- *La libertà di far girare un programma per qualsiasi scopo (libertà 0).*
- *La libertà di studiare il funzionamento di un programma, e adattarlo alle esigenze individuali (libertà 1).*
- *La libertà di ridistribuire copie di un programma per aiutare i vicini (libertà 2).*
- *La libertà di migliorare un programma, ridistribuendo al pubblico le migliorie, così che sia l’intera comunità a poterne beneficiare (libertà 3).*

Per maggiori dettagli si veda “The Free Software Definition”: http://www.gnu.org-/philosophy/free-sw.html.

[4] Si veda Eric Raymond, “Shut Up and Show Them the Code”, saggio online (28 giugno 1999). http://tuxedo.org/~esr/writings/shut-up-and-show-them.html

[5] Ibid.

[6] Si veda “Guest Interview: Eric S. Raymond”, Linux.com (18 maggio 1999). http://www.linux.com/interviews/19990518/8/

La GNU General Public License

Con l’arrivo della primavera 1985 Richard Stallman aveva posto la prima pietra miliare del progetto GNU – la versione libera dell’Emacs basata su Lisp. Un obiettivo raggiunto dopo aver superato due grossi scogli. Prima di tutto aveva dovuto ricostruire l’Emacs in modo da renderlo autonomo da qualsiasi piattaforma. Secondo, si era reso necessario procedere a un’analoga revisione della Comune dell’Emacs.

La disputa con la UniPress aveva evidenziato un difetto nel contratto sociale applicato all’interno di tale Comune. Quando gli utenti facevano pieno affidamento sull’esperienza tecnica di Stallman, vigeva il rispetto delle norme interne. In quelle aree in cui egli non conservava più la posizione di hacker-leader – ad esempio, i sistemi Unix anteriori al 1984 – i singoli e le aziende erano di liberi di creare regole proprie.

La tensione tra la libertà di modifica e quella di imporre i privilegi dell’autore esisteva in epoca antecedente all’arrivo del GOSMACS. Il Copyright Act del 1976 aveva riscritto le norme vigenti negli Stati Uniti, estendendo all’ambito del software la tutela legale del copyright. Secondo la Sezione 102(b) della legislazione, a singoli e aziende veniva garantita la protezione della “espressione” di un programma ma non quella relativa a “processi e metodi concreti utilizzati” nello stesso.[1] In altri termini, programmatori e aziende potevano ora trattare il software al pari di un testo letterario o di una canzone. L’opera avrebbe ispirato altri sviluppatori, ma per produrne una copia diretta o derivati non-satirici occorreva prima ottenere l’autorizzazione del creatore originale. Nonostante la nuova normativa garantisse che anche i programmi privi di apposito avviso fossero comunque

tutelati, i programmatori furono lesti ad affermare i propri diritti, apponendo specifiche postille sul copyright per ogni software realizzato.

Inizialmente Stallman considerò con allarmismo quelle diciture. Era raro il caso di un programma che non prendesse in prestito parti di codice da un qualche software precedente, e tuttavia era bastata la firma del Presidente e la volontà del Congresso per consentire a programmatori e aziende di affermare la proprietà individuale su quella comune nella scrittura dei programmi. Ciò introdusse inoltre una buona dose di formalismo all'interno di un sistema precedentemente del tutto informale. Anche quando gli hacker potevano dimostrare che il codice di un certo programma risaliva in realtà a qualche anno addietro, se non spesso a decenni, le risorse e il denaro necessari per controbattere legalmente la veridicità di ogni dicitura relativa al copyright superavano di gran lunga i mezzi a loro disposizione. Per dirla in maniera semplice, le dispute una volta risolte faccia a faccia tra gli hacker venivano ora passate ai rispettivi avvocati. In questo tipo di sistema erano le aziende, non i singoli programmatori, a trovarsi automaticamente in una posizione di vantaggio.

I sostenitori del copyright per il software avevano dalla loro qualche buon argomento: senza quella tutela, le opere potevano scivolare nel dominio pubblico. L'apposizione delle relative note serviva anche come una sorta di bollino di qualità. I programmatori o le aziende che associavano il proprio nome al copyright mettevano anche in gioco la propria reputazione. Si trattava infine di un contratto e di un'affermazione di proprietà. Usando il copyright come una forma di licenza elastica, l'autore poteva concedere alcuni diritti in cambio di determinati comportamenti imposti all'utente. Ad esempio, un autore poteva decidere di rinunciare al diritto di sopprimere copie non autorizzate del programma se in cambio l'utente si dichiarava d'accordo a non crearne derivati commerciali.

Si deve a quest'ultima posizione un certo alleggerimento finale della resistenza di Stallman alle note sul copyright apposte in calce al software. Ripensando a quegli anni preparatori al progetto GNU, egli sostiene di avere iniziato a percepire la natura benefica del copyright all'incirca nel periodo in

cui venne diffuso l’Emacs 15.0, ultimo aggiornamento significativo del programma precedente al progetto GNU. “Avevo notato dei messaggi e-mail contenenti diciture sul copyright con l’aggiunta di semplici licenze tipo ‘consentita la copia letterale”’ ricorda Stallman. “Ciò mi fu di grande ispirazione.”

Per l’Emacs 15.0, Stallman buttò giù una postilla sul copyright in cui si consentiva agli utenti il diritto di produrre e distribuire copie, comprese le versioni modificate, ma non quello di reclamare la proprietà esclusiva di tali versioni, come nel caso del GOSMACS.

Pur risultando d’aiuto nella stesura del contratto sociale per la Comune dell’Emacs, quel testo si dimostrò eccessivamente “informale” per gli obiettivi del progetto GNU, spiega Stallman. Qualche tempo dopo l’avvio della realizzazione di GNU Emacs, egli iniziò a consultarsi con gli altri membri della Free Software Foundation sul modo di migliorare il linguaggio di quella licenza. Si consigliò anche con i legali che l’avevano assistito nel lancio della stessa FSF.

Mark Fischer, avvocato di Boston specializzato sulle questioni relative alla proprietà intellettuale, ricorda le discussioni con Stallman di quel periodo. “Richard aveva un’opinione assai precisa sulle modalità di funzionamento”, afferma Fischer. “Partiva da due principi di base. Primo, fare in modo che il software rimanesse il più aperto possibile. Secondo, incoraggiare gli altri ad adottare le medesime procedure sulle licenze.”

Quest’ultimo punto significava tappare le falle da cui erano emerse le diverse versioni private dell’Emacs. Per farlo, Stallman e i colleghi del software libero trovarono una soluzione: gli utenti sarebbero stati liberi di modificare il GNU Emacs fin tanto che ne pubblicavano ogni cambiamento. Inoltre, le opere “derivate” avrebbero mantenuto l’identica licenza apposta in calce a quel programma.

Occorrerà tempo prima di rendersi pienamente conto della natura rivoluzionaria insita in quest’ultima norma. Allora, aggiunge Fischer, egli aveva considerato la licenza dell’Emacs per GNU come un semplice contratto. Metteva il cartellino del prezzo sull’utilizzo del programma. Anziché denaro,

Stallman imponeva agli utenti il pieno accesso alle modifiche successive. Ciò detto, però, Fischer non manca di sottolineare l'originalità dei termini contrattuali.

"Credo che il fatto di chiedere agli altri di accettare quel prezzo costituisse qualcosa di veramente insolito, se non proprio unico, per quell'epoca", rammenta.

La licenza fece il suo debutto nel 1985 in occasione della diffusione del GNU Emacs. In quella circostanza Stallman incoraggiò i commenti della comunità hacker in generale su eventuali miglioramenti da apportare al linguaggio della licenza. A farsi avanti fu John Gilmore, hacker allora consulente per la Sun Microsystems e futuro attivista a sostegno del software libero. Come parte del suo lavoro di consulenza, Gilmore aveva adattato l'Emacs per il SunOS, versione locale di Unix. Nel corso del processo, ne aveva pubblicato i cambiamenti seguendo i dettati della relativa licenza. Anziché considerare quest'ultima un'affermazione di responsabilità, la intese come chiara e concisa espressione dell'ethos tipico degli hacker. "Fino ad allora gran parte delle licenze ricorrevano a un linguaggio decisamente informale", spiega Gilmore.

E cita come esempio una nota sul copyright, risalente agli anni '80, relativa a *trn*, una utility Unix scritta da Larry Wall, futuro creatore dell'utlity "patch" per Unix nonché del linguaggio di scripting Perl. Nella speranza di mantenere un certo equilibrio fra la cortesia tipica di un hacker e il suo diritto di creatore a stabilire le modalità ci pubblicazione del software, Wall inserì nell'allegato file README la seguente dicitura sul copyright:

> Copyright (c) 1985, Larry Wall
> Potete copiare il kit trn per intero o in parte, senza però cercare di guadagnarci oppure di fingere di averlo scritto voi.[2]

Simili annotazioni, pur se in sintonia con l'etica hacker, riflettono anche la difficoltà di tradurre la natura agile e informale di tale etica nel linguaggio rigido e legale del copyright. Stilando il testo per la licenza dell'Emacs versione GNU, Stallman fece qualcosa in più che bloccare la fuga verso

derivati proprietari. Riuscì a esprimere l'etica hacker in maniera comprensibile sia per gli avvocati sia per gli stessi hacker.

Non passò molto tempo, aggiunge Gilmore, che altri hacker iniziarono a discutere le modalità per adattare tale licenza ai propri programmi. Stimolato da un'analoga conversazione su Usenet, nel novembre 1986 Gilmore inviò una e-mail a Stallman suggerendo alcune modifiche:

> Forse dovresti eliminare dal testo "EMACS" per sostituirlo con "SOFTWARE" o qualcosa di simile. Presto, speriamo, l'Emacs cesserà di essere la parte più consistente del sistema GNU, e la licenza va invece applicata al sistema per intero.[3]

Gilmore non si mostrò l'unico a suggerire un approccio più generalizzato. Alla fine del 1986, lo stesso Stallman era tutto preso dalla successiva pietra miliare del progetto GNU, un "debugger" per codice sorgente e stava cercando di rivedere la licenza dell'Emacs in modo da poterla applicare a entrambi i programmi. Soluzione: eliminare ogni specifico riferimento all'Emacs per convertirla in un copyright generico che abbracciasse l'intero progetto GNU. Nacque così la GNU General Public License, meglio nota con l'acronimo GPL.

In questo caso Stallman aderì alla convenzione di ricorrere ai numeri decimali per indicare le versioni sperimentali e ai numeri interi per quelle più stabili. La versione 1.0 venne pubblicata nel 1989 e proposta come licenza ufficiale di GNU Emacs e di GNU debugger, il secondo importante successo di Stallman nel regno della programmazione Unix. La licenza si apriva con un preambolo a chiarimento delle prospettive politiche:

> La General Public License è redatta in modo da garantire la libertà di regalare o vendere copie di software libero, di ricevere o ottenere il codice sorgente desiderato, di modificare il software o usarne parti all'interno di nuovi programmi liberi; e di informare l'utente su tutto ciò.
> A tutela dei diritti dell'utente, dobbiamo stabilire delle restrizioni in modo da impedire a chicchessia di negarvi tali diritti o chiedervi di rinunciarvi. Queste restrizioni comportano determinate responsabilità a vostro carico nel caso vogliate distribuire copie del software, oppure apportarvi delle modifiche.[4]

Mentre lavorava alla stesura del testo, Stallman fu costretto a rivedere ulteriormente le linee-guida informali relative alla vecchia Comune dell'Emacs. Laddove una volta chiedeva a ogni aderente di pubblicarne tutti i

cambiamenti apportati, ora la richiesta si limitava soltanto a quei casi in cui i programmatori facevano circolare le versioni derivate all'interno del medesimo ambito pubblico in cui operava Stallman. In altri termini, i programmatori che modificavano l'Emacs puramente a uso privato non dovevano più metterlo a conoscenza del nuovo codice. In quello che sarebbe divenuto un raro compromesso nella dottrina del software libero, decise in tal modo di ridurne drasticamente il prezzo. Gli utenti avrebbero potuto introdurre innovazioni senza il suo consenso fintanto che non ponevano alcun divieto allo stesso Stallman e al resto della comunità hacker per quanto riguardava i successivi scambi del medesimo programma.

Guardando indietro, Stallman ritiene che il compromesso sulla GPL fosse dovuto alla propria insoddisfazione per la connotazione da Grande Fratello del contratto sociale originario della Comune dell'Emacs. Per quanto gli piacesse curiosare nei sistemi altrui, la possibilità che qualcuno incaricato del mantenimento del codice di un programma potesse usare tale potere in maniera distorta lo convinse a moderare i toni della GPL.

"Non mi sembrava giusto imporre la pubblicazione di tutti i cambiamenti", sostiene Stallman. "Era sbagliato richiederne l'inoltro a un unico sviluppatore privilegiato. Quel tipo di centralizzazione e di prerogativa riservata a un singolo individuo faceva a pugni con una società in cui vigono uguali diritti per tutti."

È così che la GPL rimane una delle migliori creature partorite da Stallman. Grazie a essa, venne a crearsi un sistema di proprietà condivisa all'interno dei normali confini della legislazione sul copyright. Fatto ancor più importante, questo mise in evidenza la somiglianza intellettuale tra il codice legale e quello del software. Il preambolo della GPL conteneva un profondo messaggio implicito: anziché considerare con sospetto le norme sul copyright, gli hacker avrebbero dovuto interpretarle come un ulteriore contesto su cui poter intervenire.

"La GPL prese forma in maniera analoga a qualsiasi programma di software libero, facendo affidamento su un'ampia comunità che ne discuteva la struttura, sul rispetto delle posizioni contrarie, sulla necessità di addolcirla e

anche di trovare un compromesso per garantirne la maggiore accettazione possibile", sostiene Jerry Cohen, un altro avvocato che aiutò Stallman nella messa a punto della licenza. "Il procedimento funzionò molto bene e nelle successive versioni della GPL si è passati da una reazione diffusamente scettica e talvolta ostile a un'accettazione assai vasta."

In un'intervista apparsa nel 1986 sulla rivista *Byte*, Stallman sintetizza la GPL in toni coloriti. Oltre a una dichiarazione dei valori hacker, disse, i lettori dovrebbero considerarla anche "come una forma di jujitsu intellettuale, dove il sistema legale impostato dai magnati del software si ritorce contro di loro".[5] Qualche anno dopo, Stallman ne avrebbe descritto la creazione in termini meno duri. "Mi trovai a riflettere su diverse questioni a livello etico, politico e legale", spiegò. "Dovevo cercare di mettere insieme qualcosa che potesse rientrare nell'attuale sistema legale. Lo spirito del lavoro voleva porre le basi normative per una società di tipo nuovo ma, visto che non rappresentavo alcun governo, in realtà non potevo modificare alcuna normativa. Dovevo provarci tenendo conto dei sistemi legali attualmente in vigore, i quali non prevedevano nulla di simile."

Mentre Stallman rifletteva sulle questioni etiche, politiche e legali legate al software libero, un hacker californiano rispondente al nome di Don Hopkins gli rispedì un manuale per il microprocessore 68000. Hopkins, specializzato in Unix e amante della fantascienza, lo aveva ricevuto in prestito da Stallman tempo addietro. In segno di riconoscenza, Hopkins decorò la busta con adesivi presi a un incontro locale sulla fantascienza. Uno di questi catturò l'attenzione di Stallman. Diceva, "Copyleft (L), All Rights Reversed" (Copyleft (L), tutti i diritti rovesciati).[6]

Con la diffusione della prima versione della GPL, egli decise di adattarvi quella dicitura, dando alla licenza sul software libero il soprannome di "Copyleft". Col tempo, tale soprannome e il relativo simbolo, una "C" rovesciata, sarebbero divenuti il sinonimo ufficiale apposto dalla stessa Free Software Foundation a ogni tipo di copyright "che vuole che ogni programma sia 'software libero' e che lo stesso valga per tutte le sue versioni modificate e ampliate".

Una volta il sociologo tedesco Max Weber avanzò il sospetto che tutte le grandi religioni siano fondate sulla “routinizzazione” o l’“istituzionalizzazione” del carisma. Ogni religione di successo, sosteneva Weber, converte il carisma o il messaggio del leader originario in un apparato sociale, politico ed etico più facilmente traducibile nel passaggio tra le varie epoche e culture.

Pur se di natura tutt’altro che religiosa, la GPL si pone certamente come interessante esempio di un analogo “processo di routinizzazione” attivato all’interno dell’ambito moderno e decentralizzato tipico dello sviluppo del software. Fin dalla sua diffusione, i programmatori e le aziende che si erano dimostrati poco fedeli o leali nei confronti di Stallman optarono per l’immediata accettazione dei termini previsti dalla GPL. Per alcuni essa si è addirittura trasformata in un meccanismo preventivo per la tutela dei propri programmi. Anche quanti la rifiutano ritenendola un contratto che comporta obblighi eccessivi non possono fare a meno di confermarne l’importanza.

Uno degli hacker che rientra in quest’ultimo gruppo è Keith Bostic, impiegato presso la University of California all’epoca in cui venne diffusa la GPL 1.0. Il dipartimento in cui lavorava, il Computer Systems Research Group, ha partecipato allo sviluppo di Unix a partire dalla fine degli anni ‘70 ed era responsabile di numerose parti-chiave del sistema, tra cui il protocollo di rete TCP/IP, pietra miliare delle attuali comunicazioni via Internet.

Al tramonto degli anni ’80, la AT&T, in origine proprietaria del marchio Unix, avviò la fase di commercializzazione del sistema e iniziò a considerare la Berkeley Software Distribution, o BSD, versione accademica dello stesso Unix sviluppato da Bostic e dai colleghi di Berkeley, come elemento chiave per la messa a punto della tecnologia commerciale.

Sebbene il codice della BSD venisse condiviso tra ricercatori e programmatori commerciali tramite un’apposita licenza, l’operazione presentava un problema. Il codice della Berkeley Software Distribution era inframmezzato con il codice proprietario della AT&T. Di conseguenza le distribuzioni della prima erano disponibili soltanto per quelle istituzioni già in possesso di una licenza della AT&T. Quando quest’ultima decise di alzare i

prezzi, tale soluzione, inizialmente considerata innocua, apparve sempre più onerosa.

Assunto nel 1986, Bostic si era incaricato di trasportare la BSD sul computer PDP-11 della Digital Equipment Corporation. Fu in questo periodo, spiega lo stesso Bostic, che si trovò in stretto contatto con Stallman durante le occasionali visite di quest'ultimo in California. "Ricordo particolarmente le animate discussioni sul copyright, seduti davanti ai terminali del Computer Systems Research Group", afferma Bostic. "Discussioni che continuavamo poi a cena."

Alla fine le diverse posizioni portarono a qualcosa di concreto, sebbene in una direzione diversa da quella auspicata da Stallman. Nel giugno 1989, la Berkeley Software Distribution separò il proprio codice di rete dal resto del sistema operativo di proprietà della AT&T e iniziò a distribuirlo sotto una licenza rilasciata dalla University of California. I termini del contratto sembravano alquanto aperti. Tutto quello che si richiedeva all'utente era di dare esplicito riconoscimento all'Università nelle eventuali inserzioni pubblicitarie previste per i programmi derivati.[7] Al contrario della GPL, venivano consentite le derivazioni proprietarie. La rapida adozione di tale licenza fu frenata però da un'importante limitazione: la versione della BSD non era un sistema operativo completo. Se ne poteva studiare il codice, ma questo funzionava soltanto se integrato in altri programmi proprietari.

Nel corso degli anni successivi, Bostic e altri programmatori della University of California lavorarono alla sostituzione dei componenti mancanti, trasformando così la BSD in un sistema operativo completo e liberamente distribuibile. Pur se in ritardo per via di una disputa legale avviata dalla Unix Systems Laboratories – la sussidiaria della AT&T che rilevò la proprietà del marchio Unix – quello sforzo avrebbe prodotto frutti all'alba degli anni '90. Ma ancor prima, parecchie utility incluse nella BSD erano già finite nel progetto GNU di Stallman.

"Non credo che saremmo mai riusciti ad andare così forte senza l'influenza di GNU", sostiene Bostic, ripensando a quel periodo. "Era chiaramente un progetto in cui credevano molto e l'idea piacque anche a noi."

Fu verso la fine degli anni ’80 che la GPL iniziò a esercitare un effetto gravitazionale sulla comunità del software libero. Non occorreva che un programma usasse quella licenza per ottenere la qualifica di software libero – lo testimoniano le utility della BSD – ma aderendovi avrebbe diffuso un preciso messaggio. “Direi che l’esistenza stessa della GPL ispirò molta gente a chiedersi se il software su cui stavano lavorando fosse davvero libero e quale fosse la licenza più adeguata sotto cui distribuirlo”, sostiene Bruce Perens, creatore di Electric Fence, nota utility Unix, e futuro leader del gruppo di sviluppo di Debian per GNU/Linux. Qualche anno dopo la diffusione della GPL, Perens rammenta di aver abbandonato la licenza prevista per Electric Fence, preferendo adottare quella più corretta a livello legale messa a punto da Stallman. “In realtà si rivelò un’operazione assai semplice”, ricorda Perens.

Rich Morin, un programmatore che aveva preso l’annuncio iniziale di Stallman con un certo scetticismo, ricorda di essere rimasto impressionato dal software che iniziò a raggrupparsi sotto l’egida della GPL. Nella posizione di leader di uno user group dedicato a SunOS, uno dei compiti primari di Morin negli anni ’80 riguardava la distribuzione di nastri contenenti le migliori utility freeware o free software. Un’attività che lo obbligava a contattare gli autori originari per verificare se i loro programmi fossero protetti da copyright oppure di dominio pubblico. Fu intorno al 1989, afferma Morin, che iniziò a notare come il miglior software rientrasse generalmente sotto la licenza GPL. “Nella mia posizione di distributore di software, non appena vedevo la firma GPL la consideravo una sicura garanzia”, ricorda Morin.

A copertura del lavoro necessario per la distribuzione di quei nastri ai membri del Sun User Group, Morin era solito chiedere loro un rimborso. Ora, con il passaggio dei programmi sotto la GPL, si trovava improvvisamente a mettere insieme quelle raccolte in metà tempo, ottenendo perfino qualche minimo profitto. Intuendo l’opportunità commerciale, Morin trasformò l’hobby in attività imprenditoriale sotto il nome di Prime Time Freeware.

Iniziative commerciali del genere rientravano pienamente nei confini progettuali del software libero. “Quando parliamo di software libero ci riferiamo alla libertà, non al valore monetario”, spiegava Stallman nel

preambolo alla GPL. A partire dagli anni ’90, raffinò il concetto con una battuta più semplice: “Non pensiamo a ‘free’ (libero o gratuito) come in ‘free beer’ (birra gratis), ma piuttosto come in ‘free speech’ (libertà d’espressione)”.

Per lo più le aziende ignorarono le puntualizzazioni di Stallman. Eppure per alcuni imprenditori il termine “freedom” (libertà) associato al software libero rivestiva un significato identico a quello di “free market” (libero mercato). Eliminiamo la proprietà del software dall’equazione commerciale e avremo una situazione in cui perfino la più piccola delle aziende si sentirà libera di competere contro giganti come l’IBM e la Digital.

Uno dei primi imprenditori ad appropriarsi di una simile idea fu Michael Tiemann, programmatore e laureando presso la Stanford University. Durante gli anni ‘80, Tiemann aveva seguito il progetto GNU come un aspirante musicista jazz segue l’artista preferito. Dovette tuttavia attendere la distribuzione del GNU Compiler C (GCC) del 1987 prima di poter afferrare in pieno le potenzialità del software libero. Definendolo “una bomba”, Tiemann ritiene la sola esistenza di quel programma sufficiente a evidenziare la determinazione di Stallman nelle vesti di programmatore.

“Proprio come ogni scrittore sogna di scrivere un grande romanzo, così negli anni ’80 ogni programmatore non parlava d’altro se non di realizzare un grande compilatore”, rammenta Tiemman. “Improvvisamente Stallman ci era riuscito. Fu una cosa umiliante.”

“Se vogliamo isolare un solo punto di svolta, quello fu il GCC”, concorda Bostic. “Prima del suo lancio, nessuno poteva vantarsi di avere un compilatore.”

Anziché competere con Stallman, Tiemann decise di costruire qualcosa a partire dal suo lavoro. La versione originale del compilatore occupava 110.000 righe di codice, ma Tiemann lo ricorda come sorprendentemente semplice da capire. Era talmente semplice che in realtà gli ci vollero meno di cinque giorni per impararlo e un’ulteriore settimana per adattarlo a una nuova piattaforma hardware, il microchip 32032 della National Semiconductor. Nel corso dell’anno successivo, Tiemann iniziò a giocare con il codice, finendo col creare un compilatore nativo per il linguaggio C++. Un giorno, intervenendo a

un evento presso i Bell Labs, Tiemann venne a sapere che alcuni sviluppatori della AT&T erano alle prese con un identico progetto.

“In sala c’erano 40 o 50 persone, e chiesi se qualcuno stesse lavorando a quel compilatore”, ricorda. “Uno di loro rispose che si trattava di un’informazione riservata, aggiungendo però che una semplice occhiata in giro sarebbe stata sufficiente a fornirmi il quadro della situazione.”

Comunque sia, l’idea giusta prese corpo qualche tempo dopo. “Lavoravo a quel progetto ormai da sei mesi. Pensai, ‘non so se dipende da me o dal codice, ma il livello di efficacia raggiunto è tale da potersi aspettare qualche tipo di ricompensa dal libero mercato”’, chiarisce Tiemann.

Un’ulteriore fonte d’ispirazione giunse dal Manifesto GNU il cui testo, pur ferendo la cupidigia di alcuni rivenditori, ne incoraggia altri a considerare i vantaggi del software libero dal punto di vista del consumatore. Eliminando il potere del monopolio dall’ambito del software commerciale, la GPL consente infatti ai distributori più accorti di competere efficacemente nell’ambito dei servizi e della consulenza, le due aree più redditizie di quel mercato.

In un saggio redatto nel 1999, Tiemann ricorda l’impatto del Manifesto di Stallman. “[…] in superficie può apparire polemica socialista, ma io vi ho trovato qualcosa di diverso. Nascosto tra le righe vi ho intravisto un progetto commerciale.”[8]

In collaborazione con John Gilmore, altro entusiasta del progetto GNU, Tiemann lanciò così un servizio di consulenza dedicato alla personalizzazione di programmi GNU. Con il nome di Cygnus Support, la società firmò il suo primo contratto nel febbraio 1990. A fine anno ne aveva raccolti per un valore complessivo pari a 725.000 dollari.

GNU Emacs, GDB e GCC erano i “tre grandi” tra gli strumenti per sviluppatori, ma non furono gli unici messi a punto da Stallman durante i primi cinque anni del progetto GNU. Entro il 1990, Stallman aveva realizzato anche la versione GNU della Bourne Shell (ribattezzata Bourne Again Shell, o BASH), YACC (rinominata Bison), e awk (gawk). Al pari di GCC, ciascun programma GNU era studiato per girare su molteplici sistemi, non unicamente sulla piattaforma di un solo produttore. Con l’intento di fornire maggiore

elasticità a quei programmi, spesso Stallman e i suoi collaboratori finivano per renderli anche più utili.

Riferendosi all'approccio universale di GNU, Morin della Prime Time Freeware sottolinea in particolare un pacchetto, importante pur se di portata ridotta, chiamato *hello*. "Si tratta di un programma composto da cinque righe in C, organizzato come fosse una distribuzione GNU", spiega Morin. "In tal modo riuscì a conquistarsi moduli tipo il 'Texinfo' e il 'configure'. Per consentire al software di adattarsi facilmente ad ambienti diversi, si è cioè avvalso di tutti gli altri accessori utilizzati dal progetto GNU. Si tratta di un'operazione di estrema importanza che finì per estendersi non soltanto a tutto il software [di Stallman] ma anche a quello del progetto GNU."

Secondo Stallman, il miglioramento dei programmi rappresentava un'attività secondaria rispetto alla loro scrittura. "Non ero affatto certo di essere in grado di perfezionare questo o quel programma", spiegò nell'intervista a *Byte*. "Quando ci riuscivo, in un certo senso a beneficiarne era la procedura di re-implementazione, operazione che tende in ogni caso a migliorare le prestazioni di ogni sistema. D'altra parte, ciò può accadere perché è da molto tempo che lavoro su sistemi diversi, di conseguenza mi vengono buone idee per adattarli al meglio."[9]

Comunque sia, mentre sul finire degli anni '80 gli strumenti GNU iniziavano a lasciare il segno, la reputazione acquisita da Stallman all'epoca del laboratorio di intelligenza artificiale, per quanto riguarda l'eccessiva pignoleria progettuale, divenne leggendaria nell'intera comunità degli sviluppatori.

Jeremy Allison, allora utente Sun e programmatore destinato a lanciare un decennio più avanti un proprio progetto di software libero, Samba, rammenta quella reputazione ridendoci sopra. Durante gli ultimi anni '80, Allison iniziò a usare l'Emacs. Ispirato dal modello di sviluppo comunitario, dice di aver inviato a Stallman il proprio contributo solo per vederselo rifiutare.

"Si può riassumere il tutto con un titolo della testata satirica online *Onion*", scherza Allison. "Qualcosa tipo, 'Dio dice no alle preghiere di un fanciullo."'

La crescente statura di Stallman come programmatore veniva tuttavia controbilanciata dai problemi in cui si dibatteva nelle vesti di project manager. Anche se GNU passava di successo in successo nella creazione di strumenti per lo sviluppo, l'incapacità di generare un kernel funzionante – il programma di controllo centrale di tutti i sistemi Unix capace di determinare quali periferiche e applicazioni debbano avere accesso al microprocessore e quando – anche prima della fine degli anni '80 stava provocando qualche mugugno di delusione.

Come per gli altri componenti del progetto GNU, Stallman ne aveva iniziato lo sviluppo cercando un programma già in circolazione da modificare. Secondo la "GNUsletter" della fine degli anni '80, questo approccio, come il tentativo iniziale di costruire GCC partendo da Pastel, era tutt'altro che ottimale. Una GNUsletter del gennaio 1987 riporta che il progetto GNU stava lavorando al rifacimento di TRIX, un kernel Unix realizzato al MIT.

Nel febbraio del 1987, una newsletter segnala che il progetto aveva spostato l'attenzione su Mach, un "micro-kernel" leggero sviluppato presso la Carnegie Mellon. Ciò nonostante, il lavoro effettivo sul kernel del progetto GNU non prese ufficialmente il via prima del 1990.[10]

Un simile ritardo rappresentava soltanto una delle numerose preoccupazioni che incombevano su Stallman in quel periodo. Nel 1989, la Lotus Development Corporation avviò una azione legale contro un'azienda rivale, la Paperback Software International, per aver copiato i comandi del menu presenti nel popolare spreadsheet Lotus 1-2-3. La denuncia della Lotus, in aggiunta alla battaglia Apple-Microsoft sull'aspetto tipico delle applicazioni, provocò una ricaduta problematica per il progetto GNU. Anche se le due vicende giudiziarie esulavano dagli obiettivi perseguiti da tale progetto, entrambe riguardavano sistemi operativi e applicazioni realizzate per il personal computer – non per sistemi compatibili con Unix – minacciavano comunque di propagare un micidiale effetto a catena sull'intera cultura degli sviluppatori. Deciso a far qualcosa, Stallman incaricò alcuni amici programmatori di realizzare un'inserzione pubblicitaria per alcune riviste per opporsi alle due vertenze. Proseguì poi avviando l'organizzazione di un

gruppo di protesta contro simili iniziative delle corporation. Sotto il nome di League of Programming Freedom (Lega per la libertà di programmazione), il gruppo tenne dimostrazioni di protesta davanti agli uffici della Lotus e all'aula giudiziaria di Boston che ospitava il processo.

Proteste che ottennero una certa visibilità,[11] poiché documentavano la natura in continua evoluzione dell'industria del software. Le applicazioni avevano silenziosamente sostituito i sistemi operativi come principale campo di battaglia delle grandi aziende. Nella sua ricerca senza esito di costruire un sistema operativo di software libero, il progetto GNU sembrava però perdere irrimediabilmente il passo con i tempi. Anzi, agli occhi di alcuni osservatori, proprio il fatto che Stallman avesse ritenuto necessario mettere insieme un gruppo completamente nuovo per opporsi alle azioni legali non faceva altro che rafforzare l'idea di una certa obsolescenza in atto.

Nel 1990, la John D. & Catherine T. MacArthur Foundation riconobbe il genio di Stallman assegnandogli un riconoscimento in denaro. Il premio, una ricompensa di 240.000 dollari per il lancio del progetto GNU e per aver dato voce alla filosofia del software libero, risolse alcune preoccupazioni a breve termine. Per prima cosa consentì a Stallman, dipendente non stipendiato della FSF il cui sostentamento era unicamente basato su contratti di consulenza, di dedicare maggior tempo alla scrittura del codice GNU.[12]

Ironicamente, il riconoscimento permise inoltre a Stallman di esercitare il diritto di voto. Qualche mese prima un incendio aveva distrutto i suoi pochi possedimenti terreni. Ufficialmente risultava perciò "senza fissa dimora",[13] residente al 545 di Technology Square. "[L'ufficio elettorale] non volle accettare quell'indirizzo come mio domicilio", avrebbe ricordato successivamente Stallman. "Ma cambiarono idea subito dopo la pubblicazione su un quotidiano di un articolo in cui si parlava del premio assegnatomi dalla MacArthur Foundation."[14]

Fatto ancora più importante, quel denaro gli garantì maggiore libertà di movimento. Già completamente immerso nelle tematiche relative al software libero, poté così incrementare i viaggi a sostegno della missione del progetto GNU.

È interessante notare come il definitivo successo di tale progetto e più in generale del movimento del software libero prese avvio da uno di questi viaggi. Nel 1990 Stallman visitò il Politecnico di Helsinki, in Finlandia. Tra il pubblico sedeva anche il ventunenne Linus Torvalds, futuro sviluppatore del kernel Linux, destinato a colmare il vuoto di maggiori proporzioni presente nel progetto GNU.

Allora studente presso la vicina Università, Torvalds osservò Stallman non senza meraviglia. “Per la prima volta in vita mia, eccomi davanti allo stereotipo dell’hacker con i capelli lunghi e la barba”, scrive Torvalds nella sua autobiografia del 2001, *Rivoluzionario per caso*. “Era difficile imbattersi in gente simile a Helsinki.”[15]

Pur se lontano dalle posizioni “sociopolitiche” di Stallman, Torvalds considerò comunque positiva la logica che lo motivava: nessun programmatore scrive codice privo di errori. Grazie alla condivisione del software, per gli hacker il miglioramento di un programma diviene la molla prioritaria rispetto a motivazioni individuali quali cupidigia o egoismo.

Come molti programmatori della propria generazione, Torvalds non si era formato sui mainframe tipo l’IBM 7094, bensì su un vasto assortimento di sistemi fatti in casa. Seguendo i corsi universitari, era passato dalla programmazione su PC a Unix, usando il MicroVAX dell’università. Questa crescita progressiva gli aveva offerto una diversa prospettiva sugli ostacoli che si frapponevano all’accesso alle macchine. Per Stallman, gli scogli maggiori erano costituiti da burocrazia e privilegi interni. Per Torvalds si trattava invece della geografia e del gelido inverno di Helsinki. Costretto ad attraversare i viali dell’Università di quella città unicamente per avere accesso al proprio account Unix, Torvalds iniziò presto a cercare un modo per farlo dal calduccio del proprio appartamento situato fuori dal campus.

Una ricerca che lo condusse presto al sistema operativo Minix, versione leggera di Unix sviluppata a scopo didattico dal professore universitario olandese Andrew Tanenbaum. Il programma stava nei limiti della memoria di un PC 386, la macchina più potente che Torvalds potesse permettersi, ma difettava ancora di alcune componenti essenziali. Mancava soprattutto

l’emulazione di terminale, funzione che poteva consentirgli di emulare quello universitario, rendendo così possibile il collegamento al MicroVAX direttamente da casa.

Nell’estate 1991 Torvalds iniziò così a riscrivere da capo Minix, aggiungendovi nello stesso tempo nuove funzioni. A fine estate parlava del suo lavoro come “del GNU/Emacs degli emulatori di terminale”.[16] Sentendosi sicuro, chiese al newsgroup Minix copie degli standard POSIX, il software di base che stabiliva la compatibilità di un programma con Unix. E qualche settimana dopo fece girare online un messaggio stranamente analogo al testo originario relativo a GNU, diffuso da Stallman nel 1983:

> Un saluto a tutti coloro che usano minix
> Sto lavorando su un sistema operativo (libero), giusto per hobby, nulla di grande e professionale come gnu, per i cloni AT 386 (486). È in costruzione da aprile, e si può dire quasi pronto. Vorrei ricevere commenti su quello che vi piace o non vi piace in minix, poiché il mio sistema operativo gli assomiglia parecchio (tra le altre cose ha la stessa configurazione fisica del file-system, per motivi pratici).[17]

Il messaggio attirò una valanga di risposte e, nel giro di un mese, Torvalds inseriva su un sito FTP la versione 0.01 del sistema operativo – ovvero, la prima stesura che fosse possibile sottoporre a revisioni esterne. Nel frattempo aveva anche trovato un nome per il nuovo sistema. Sul disco fisso del PC, lo aveva salvato come Linux, in omaggio alla consuetudine di chiamare ogni variazione di Unix con un nome che finisse in X. Ritenendo però quel nome troppo “egocentrico”, Torvalds decise di sostituirlo con Freax, solo per poi scoprire che il gestore del sito FTP l’aveva riportato al nome originario.

Anche se animati dall’intenzione di costruire un vero e proprio sistema operativo, sia Torvalds che altri programmatori erano al corrente della disponibilità di gran parte degli strumenti operativi necessari, grazie all’opera di GNU, BSD e di altri sviluppatori di software libero. Uno dei primi applicativi da cui il gruppo di lavoro su Linux trasse vantaggio fu il GNU C Compiler, strumento che rese possibile l’elaborazione di programmi scritti in C.

L’integrazione del GCC migliorò le prestazioni di Linux. Ma sollevò anche problemi. Nonostante i poteri “virali” della GPL non avessero nulla a che fare

con il nuovo kernel, la decisione di Torvalds di prendere in prestito il GCC per il proprio sistema operativo libero stava a indicare un certo obbligo a permettere che altri utenti facessero lo stesso. Come dice lo stesso Torvalds: “Mi ero arrampicato sulle spalle di giganti.”[18] Inevitabilmente iniziò a pensare a cosa sarebbe accaduto quando qualcun altro gli avesse avanzato una richiesta analoga. Dieci anni dopo quella decisione, Torvalds riecheggia la posizione di Robert Chassel della Free Software Foundation nel riassumere le proprie riflessioni in quella circostanza:

> Dopo aver speso sei mesi della tua vita in quel lavoro, decidi di metterlo a disposizione di tutti, anche per tirarci fuori qualcosa, ma non vuoi che altri se ne approfittino. Volevo che la gente potesse prenderne visione e apportarvi modifiche e migliorie a piacimento. Ma dovevo anche assicurarmi di poter verificare tali modifiche. Volevo garantirmi l’accesso continuato al codice, in modo da poter sempre riprendere le eventuali migliorie altrui.[19]

Quando giunse il momento di distribuire la versione 0.12 di Linux, la prima a contenere una versione pienamente integrata del GCC, Torvalds decise di rendere esplicita la propria lealtà al movimento del software libero. Gettò via la licenza del vecchio kernel per sostituirla con la GPL. La decisione diede vita a una miriade di adattamenti aggiuntivi, con Torvalds e i vari collaboratori alla ricerca di altri programmi GNU da inserire nell’emergente calderone di Linux. Nel giro di tre anni, gli sviluppatori erano pronti a distribuire il primo vero sistema, Linux 1.0, che comprendeva le versioni pienamente modificate di GCC e GDB oltre a un’ampia serie di strumenti BSD.

Entro il 1994, il sistema operativo così amalgamato aveva guadagnato sufficiente considerazione nel mondo hacker, tanto che alcuni osservatori si chiesero se Torvalds non avesse fatto male ad abbracciare la GPL nella fase iniziale del progetto. Nel primo numero della rivista *Linux Journal*, l’editore Robert Young tenne un’intervista a quattr’occhi con Torvalds. Alla domanda se non si fosse pentito di aver rinunciato alla proprietà privata del codice, il programmatore finlandese rispose di no. “Anche con il senno di poi”, questa la replica di Torvalds, considerava l’adesione alla GPL “una delle decisioni migliori” prese durante i primi passi del progetto Linux.[20]

Una decisione, non certo presa per rispetto o deferenza nei confronti di Stallman e della Free Software Foundation, che testimonia la crescente portabilità della GPL. Anche se ci vollero alcuni anni prima che Stallman se ne rendesse conto, l'esplosione dello sviluppo di Linux ricreava l'analogo scenario dell'Emacs. Stavolta, tuttavia, l'innovazione alla base di tale esplosione non era una trovata come il Control-R quanto piuttosto la novità di far girare un sistema simile a Unix sull'architettura di un PC. Ma anche con motivazioni diverse il risultato finale rifletteva in pieno le caratteristiche etiche di base: un sistema operativo pienamente funzionale composto interamente di software libero.

Come indicava quella prima e-mail inviata al newsgroup comp.os.minix, per alcuni mesi Torvalds considerò Linux poco più che una sorta di intermezzo, fino a quando gli sviluppatori GNU non avessero realizzato il kernel HURD. Questa volontà iniziale di non valutare Linux in termini politici rappresenta un grosso colpo inferto alla Free Software Foundation.

A livello personale Torvalds non era altro che l'ultimo di una lunga fila di ragazzi interessati semplicemente a smontare e rimontare le cose per puro divertimento. Ciò nonostante, sintetizzando il successo di un progetto che altrettanto facilmente avrebbe potuto rimanere per sempre sul disco fisso di un computer abbandonato, egli attribuisce alla propria giovinezza il merito di aver avuto la saggezza di lasciar perdere ogni controllo per accettare invece quanto proposto dalla GPL.

"Forse non ho visto la luce", scrive Torvalds riflettendo su quell'intervento di Stallman al Politecnico e sulla successiva decisione di aderire alla GPL. "Ma direi che qualcosa del suo discorso mi era rimasto dentro."[21]

[1] Si veda Hal Abelson, Mike Fischer & Joanne Costello, "Software and Copyright Law", versione aggiornata (1998). http://www.swiss.ai.mit.edu/6805/articles/int-prop/software-copyright.html

[2] Si veda Trn Kit README. http://www.za.debian.org/doc/trn/trn-readme

[3] Si veda John Gilmore, citazione tratta da una sua e-mail all'autore del libro.

[4] Si veda Richard Stallman, et al., "GNU General Public License: Version 1", (febbraio 1989). http://www.gnu.org/copyleft/copying-1.0.html

[5] Si veda David Betz & Jon Edwards, “Richard Stallman discute con i redattori di *Byte* il proprio sistema di pubblico dominio [sic] compatibile con Unix”, *Byte* (luglio 1996). (Ripubblicata sul sito del progetto GNU: http://www.gnu.org/gnu/byte-interview.html.)
L’intervista getta un’interessante, per non dire candida, luce sulle posizioni politiche di Stallman all’alba del progetto GNU. Risulta inoltre d’aiuto nel tracciare l’evoluzione della sua retorica.
Descrivendo l’obiettivo della GPL, Stallman sostiene: “Sto cercando di cambiare l’approccio alla conoscenza e all’informazione in generale. Secondo me, ogni tentativo di rendere proprietaria tale conoscenza, di limitarne o meno l’utilizzo da parte dei singoli, oppure di impedire ad altri di condividerla, va considerato un sabotaggio”.
Confrontiamo simili posizioni con un’affermazione resa dallo stesso Stallman all’autore del libro nell’agosto 2000: “Vi esorto a non usare il termine ‘proprietà intellettuale’ nelle vostre riflessioni. Ciò potrebbe ingenerare equivoci, perché quel termine suggerisce un’eccessiva generalizzazione tra copyright, brevetti e marchi commerciali. Si tratta di elementi dagli effetti talmente diversi tra loro che è del tutto folle discuterne come di un unico insieme. Se qualcuno parla di proprietà intellettuale senza virgolette, vuol dire che difetta di chiarezza, e allora è meglio lasciar perdere”.

[6] Il gioco di parole fra “reserved” (riservati) e “reversed” (rovesciati) è intraducibile. [*N.d.R.*]

[7] Quella “odiosa clausola pubblicitaria” della University of California in seguito si sarebbe rivelata un problema. Alla ricerca di un’alternativa meno restrittiva della GPL, alcuni hacker ricorsero a tale clausola, sostituendo la dicitura “University of California” con il nome della propria istituzione. Risultato: i programmi di software libero che utilizzavano parti prese da decine di altri programmi dovevano citare decine di entità. Nel 1999, dopo circa dieci anni di insistenze da parte di Stallman, la University of California si trovò d’accordo a eliminare quella clausola. Si veda “The BSD License Problem”: http://www.gnu.org/philosophy/bsd.html.

[8] Si veda Michael Tiemann, “Il futuro della Cygnus Solutions: resoconto di un imprenditore”, *Open Sources*, Apogeo, 1999, p. 78.

[9] Si veda Richard Stallman, *Byte* (1986).

[10] Si veda “HURD History”, http://www.gnu.org/software/hurd/history.html

[11] Secondo un comunicato della League of Programming Freedom, le proteste attirarono l’attenzione, perché costituivano il primo esempio di strofe di protesta esadecimali:
1-2-3-4, toss the lawyers out the door; (1-2-3-4, butta fuori l’avvocato)
5-6-7-8, innovate don’t litigate; (5-6-7-8, innovazione non cause legali)
9-A-B-C, 1-2-3 is not for me; (9-A-B-C, 1-2-3 non fa per me)
D-E-F-O, look and feel have got to go (D-E-F-O, l’aspetto tipico può restare)
http://lpf.ai.mit.edu/Links/prep.ai.mit.edu/demo.final.release

[12] Qui il termine “ scrivere” va inteso in senso lato. Più o meno in concomitanza con il premio della MacArthur, Stallman iniziò a soffrire di dolori cronici alle mani e quindi a dettare il suo lavoro ai dattilografi della FSF. Qualcuno pensava che potesse trattarsi dei tipici dolori dovuti a stress ripetitivi (RSI), problema comune tra i programmatori, ma Stallman non ne appare convinto al 100%: “Non era la sindrome del tunnel carpale, poiché il problema era nella mani e non nei polsi”. Da allora ha imparato a fare a meno dei dattilografi, utilizzando una tastiera che richiede una minore pressione delle dita.

[13] Si veda Reuven Lerner, “Stallman wins $240,000 MacArthur award”, MIT, The Tech (18 luglio 1990). http://the-tech.mit.edu/V110/N30/rms.30n.html

[14] Si veda Michael Gross, “Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius” (1999).

[15] Si veda Linus Torvalds & David Diamond, *Rivoluzionario per caso*. Milano, Garzanti, 2001.

[16] Ibid.

[17] Si veda “Linux 10th Anniversary”. http://www.linux10.org/history/

[18] Si veda Linus Torvalds & David Diamond, *Rivoluzionario per caso*. Milano, Garzanti, 2001.

[19] Ibid.

[20] Si veda Robert Young, “Interview with Linus, the Author of Linux”, Linux Journal (1 marzo 1994). http://www.linuxjournal.com/article.php?sid=2736

[21] Si veda Linus Torvalds & David Diamond, *Rivoluzionario per caso*, Milano, Garzanti, 2001.

GNU/Linux

Nel 1993 il movimento del software libero si trovò a un bivio. Per chi era incline all'ottimismo, tutti i segnali sembravano confermare il successo della cultura hacker. *Wired*, nuovo mensile specializzato centrato su tematiche quali la crittografia, Usenet e la libertà nel software, andava letteralmente a ruba. Internet, una volta termine confinato al gergo di hacker e ricercatori, si era fatto strada nel lessico comune. Veniva usato perfino dal presidente Clinton. Il personal computer, un tempo giocattolo per hobbysti, aveva guadagnato una diffusa rispettabilità, garantendo a un'intera generazione di nuovi utenti l'accesso ai programmi realizzati dagli hacker. E anche se il progetto GNU non era ancora riuscito a raggiungere l'obiettivo di un sistema operativo completo di software libero, i più curiosi potevano intanto provare Linux.

Sotto qualunque punto di vista, le notizie erano di segno positivo, o almeno così sembrava. Dopo un decennio di stenti, gli hacker e i loro valori stavano finalmente ottenendo il riconoscimento della società. La gente iniziava a comprenderli.

Ma stavano davvero così le cose? Per i pessimisti, ogni segno di accettazione implicava una contropartita negativa. Certo, improvvisamente essere un hacker faceva moda, ma la cosa poteva considerarsi positiva per una comunità cresciuta nell'alienazione? Sicuro, la Casa Bianca andava tessendo le lodi di Internet, spingendosi al punto di registrare un proprio dominio, whitehouse.gov, ma al contempo prendeva accordi con l'imprenditoria, i fautori della censura e i rappresentanti delle forze dell'ordine nel tentativo di addomesticare la cultura da Far West imperante su Internet. Chiaro, i PC offrivano una potenza sempre maggiore, ma con la trasformazione dei chip in

bene di consumo, Intel aveva creato una situazione in cui il potere si era semplicemente trasferito nelle mani dei rivenditori di software proprietario.

Per ogni nuovo utente conquistato alla causa del software libero tramite Linux, ne esistevano centinaia, forse migliaia, che avviavano Windows per la prima volta.

Occorreva infine fare i conti con la curiosa natura dello stesso Linux. Superati in scioltezza problemi di progetto (come accaduto a GNU) e dispute legali (era il caso della BSD), la rapida evoluzione del sistema risultò talmente imprevista, il suo successo così casuale che gli stessi programmatori più addentro nel codice non sapevano che cosa farne. Più simile a una "compilation" che a un vero e proprio sistema operativo, Linux era composto dai pezzi migliori del repertorio hacker: si andava da GCC, GDB e glibc (la nuova libreria in C realizzata dal progetto GNU) fino a X (interfaccia grafica per gli utenti basata su Unix, sviluppata presso il laboratorio d'informatica del MIT) e ad altri strumenti creati dal giro BSD, quali BIND (il Berkeley Internet Naming Daemon, che consente la sostituzione di domini Internet facili da ricordare al posto degli indirizzi IP numerici) e TCP/IP. La colonna portante della creazione era naturalmente il kernel – a sua volta versione rivisitata e super-potenziata di Minix. Anziché costruire il sistema operativo partendo da zero, Torvalds e l'annesso gruppo di sviluppo in rapida espansione avevano seguito il vecchio adagio di Picasso, "i buoni artisti prendono in prestito, i grandi rubano". Oppure, come Torvalds stesso interpreterà più tardi la frase, descrivendo il segreto del proprio successo: "Sostanzialmente sono una persona molto pigra a cui piace prendersi il merito di quello che in realtà ha fatto qualcun altro".[1]

Una simile pigrizia, pur se ammirabile dal punto di vista dell'efficienza, risultò problematica dal punto di vista politico, poiché sottolineava in primo luogo la mancanza di un programma ideologico da parte di Torvalds. Al contrario degli sviluppatori GNU, egli non si era cimentato nella realizzazione di un sistema operativo spinto dal desiderio di offrire ai colleghi hacker uno strumento per lavorare; l'aveva fatto per giocarci lui stesso. Come Tom Sawyer, che imbiancava uno steccato grazie all'aiuto altrui, il genio di

Torvalds stava meno nella visione complessiva e più nella capacità di coinvolgere altri hacker per accelerare il processo.

Il fatto che Torvalds, assieme ai suoi, fosse riuscito laddove altri avevano fallito suscitava domande scomode: che cosa rappresentava esattamente Linux? Si poteva forse considerare la manifestazione di quella filosofia del software libero articolata per primo da Stallman nel Manifesto GNU? O era semplicemente un amalgama di buoni programmi che qualunque utente, ugualmente motivato, avrebbe potuto assemblare sul proprio sistema casalingo?

Verso la fine del 1993, un numero crescente di utenti Linux iniziavano a propendere verso quest'ultima ipotesi, dando così vita a variazioni private sul tema. Si fecero anche abbastanza coraggiosi da imbottigliare e rivendere le proprie varianti – o "distribuzioni" – ad altri appassionati di Unix. Con risultati a dir poco alterni.

"Non si profilava ancora all'orizzonte l'arrivo di Red Hat e delle altre distribuzioni commerciali", ricorda Ian Murdock, allora studente d'informatica presso la Purdue University. "Sfogliando qualche rivista dedicata al mondo Unix ci si imbatteva in tutti quegli annunci formato biglietto da visita che pubblicizzavano 'Linux'. Gran parte delle aziende si dedicavano a quelle operazioni nottetempo, non vedendo nulla di sbagliato nell'integrazione del prodotto finale con qualche stringa di codice realizzato in proprio."

Murdock, programmatore Unix, rammenta di essere rimasto "travolto" da Linux poco tempo dopo averlo prelevato e installato per la prima volta sul PC di casa. "Era davvero divertente", sostiene. "Decisi subito di farmi coinvolgere." Un entusiasmo che tuttavia iniziò a smorzarsi per via di quell'esplosione di distribuzioni malamente realizzate. Avendo deciso che il modo migliore di farsi coinvolgere consisteva nel costruirne una versione priva di additivi, Murdock buttò giù un elenco dei migliori strumenti di software libero disponibili con l'intento di inserirli nella distribuzione personale. "Volevo qualcosa che fosse davvero all'altezza del nome Linux", spiega Murdock.

Nel tentativo di “suscitare qualche interesse”, illustrò le proprie intenzioni in vari ambiti su Internet, compreso il newsgroup di Usenet comp.os.linux. Uno dei primi messaggi di risposta giunse da rms@ai.mit.edu. Da buon hacker Murdock riconobbe immediatamente quell’indirizzo. Si trattava di Richard M. Stallman, fondatore del progetto GNU, colui che da sempre egli considerava “l’hacker degli hacker”. Notando l’indirizzo nella posta in arrivo, ne rimase sconcertato. Perché mai Stallman, già leader di un progetto personale per un sistema operativo, avrebbe dovuto interessarsi alle idee di Murdock per Linux?

“Il messaggio diceva che la Free Software Foundation seguiva da vicino la crescita di Linux ed era interessata alla possibile realizzazione di un sistema analogo. In pratica, considerava i nostri obiettivi in linea con la loro filosofia.”

Quel messaggio rappresentava un completo dietro-front da parte di Stallman. Fino al 1993 si era accontentato di tenere il naso fuori dagli affari della comunità Linux. Anzi, nel 1991 aveva completamente ignorato il sistema operativo rinnegato in occasione della sua prima apparizione sullo scenario della programmazione Unix. Dopo aver ricevuto la prima notifica di un sistema operativo simile a Unix in grado di girare su PC, Stallman disse di aver delegato a un amico il compito di esaminarlo. Ricorda Stallman: “Mi fu poi spiegato che il software era modellato sulla base del System V, una versione inferiore di Unix, e che non poteva essere adattato ad altre piattaforme”.

Il rapporto dell’amico era corretto. Costruito per girare su macchine basate sul 386, Linux era fermamente radicato in quelle piattaforme a basso costo. Quel che l’amico mancò di segnalare, tuttavia, fu il notevole vantaggio che Linux godeva in quanto unico sistema operativo liberamente modificabile presente sul mercato. In altri termini, mentre Stallman trascorse i tre anni successivi ad ascoltare i problemi riportati dal gruppo che lavorava su HURD, Torvalds andava conquistandosi quei programmatori che in seguito avrebbero rivisitato il sistema così da adattarlo ad altre piattaforme.

Nel 1993, l’incapacità del progetto GNU di produrre un kernel funzionante causò problemi all’interno sia dello stesso progetto sia del movimento del

software libero più in generale. Nel marzo di quell'anno un articolo a firma Simson Garfinkel apparso sul mensile *Wired* descriveva il progetto GNU come "impantanato", nonostante il successo dei numerosi strumenti realizzati fino a quel momento.[2] Quanti operavano all'interno del progetto e dell'annessa struttura non-profit, la Free Software Foundation, ricordano uno stato d'animo ancora più abbattuto di quanto non trasparisse nell'articolo di Garfinkel. "Personalmente in quel periodo ero consapevole dell'esistenza di una finestra di opportunità per l'introduzione di un nuovo sistema operativo", sostiene Chassell. "Ma una volta richiusa tale finestra, la gente avrebbe perso interesse. Fu esattamente quel che accadde."[3]

Molto si è detto sui guai che afflissero il progetto GNU nel periodo 1990-1993. Mentre qualcuno getta ogni colpa su Stallman, secondo Eric Raymond, tra i primi aderenti al gruppo di lavoro sull'Emacs GNU e successivamente critico nei confronti dello stesso Stallman, si trattò chiaramente di un problema istituzionale. "La FSF divenne arrogante", afferma Raymond. "L'obiettivo della sua attività si spostò dalla realizzazione di un sistema operativo pronto per la produzione alle ricerche sui sistemi possibili." Ancor peggio, "credeva che nulla di quanto avveniva al di fuori avrebbe mai potuto toccarla".

Murdock, persona meno interessata alle faccende interne del progetto GNU, offre una prospettiva più giudiziosa. "Credo che in parte del problema fosse che erano diventati un po' troppo ambiziosi e avevano buttato via inutilmente un sacco di soldi", sostiene Murdock. "A cavallo tra gli anni '80 e '90 andavano forte i micro-kernel. Purtroppo questo ha coinciso con l'avvio del loro sistema operativo. Alla fine si trovarono sotto il peso di un bagaglio eccessivamente pesante e sarebbe occorso troppo lavoro per disfarsene."

Per spiegare il ritardo, Stallman cita una serie di questioni. Le azioni legali della Lotus e della Apple si rivelarono delle distrazioni politiche che, unite all'impossibilità di digitare da parte sua, gli resero difficile aiutare coloro che lavoravano su HURD. Menziona inoltre lo scarso livello di comunicazione tra le varie aree del progetto GNU. "Fu davvero duro far funzionare l'ambiente del 'debugging"', ricorda. "E le persone che a quel tempo gestivano il GDB

sembravano restie a cooperare.” Tuttavia Stallman ritiene che per lo più si trattò di un errore di valutazione da parte di tutti i membri del progetto, lui compreso, sulle difficoltà insite nell’espansione del micro-kernel Mach in un kernel Unix a tutto tondo.

“Pensavo: bene, la parte [del Mach] che deve comunicare con la macchina ha già superato la fase di debugging”, così Stallman ricorda in un discorso del 2000 i problemi della squadra di lavoro su HURD. “Grazie a quest’avvio veloce, dovremmo farcela in fretta. E invece successe che il debugging di questi programmi asincroni multithread si dimostrò assai complicato. C’erano bug di tempificazione che massacravano i file; non è stato molto piacevole. Di conseguenza ci vollero molti, molti anni per produrre una versione buona da sottoporre al primo vero test.”[4]

Qualunque sia il motivo, o i motivi, il concomitante successo del gruppo del kernel Linux non fece altro che accentuare una situazione già tesa. Certo, quel kernel era stato diffuso sotto la licenza GPL ma, come fece notare lo stesso Murdock, la volontà di considerare Linux come un sistema operativo di puro software libero era tutt’altro che unanime. Verso la fine del 1993, la popolazione totale degli utenti Linux da una dozzina o poco più di entusiasti del Minix aveva raggiunto una cifra compresa tra le 20.000 e le 100.000 unità.[5] Quello che una volta pareva poco più di un hobby stava trasformandosi in un mercato pronto per essere sfruttato. Al pari di Winston Churchill che osservava le truppe sovietiche entrare a Berlino, Stallman provò un comprensibile miscuglio di emozioni quando giunse il momento di celebrare la “vittoria” di Linux.[6]

Pur se in ritardo per partecipare alla festa, Stallman esercitava ancora parecchia influenza. Non appena la FSF annunciò il prestito di denaro e sostegno morale al progetto di Murdock, piovvero altre offerte di supporto. Murdock chiamò il nuovo progetto Debian – contrazione del suo nome e di quello della moglie, Deborah – e nel giro di un paio di settimane la prima distribuzione era cosa fatta. “Quasi da un giorno all’altro, [il sostegno di Richard] catapultò Debian da quel piccolo progetto interessante a qualcosa cui l’intera comunità doveva prestare attenzione”, spiega Murdock.

Nel gennaio 1994, egli diffuse il “Manifesto Debian”. Redatto nello spirito del “Manifesto GNU” stilato da Stallman un decennio prima, sottolineava l’importanza di lavorare a stretto contatto con la Free Software Foundation. Scriveva Murdock:

> La Free Software Foundation gioca un ruolo estremamente vitale nel futuro di Debian. Per il semplice fatto che la stessa ne curerà la distribuzione, viene trasmesso al mondo il messaggio che Linux non è un prodotto commerciale e mai dovrebbe diventarlo, ma ciò non significa che non sarà mai in grado di competere in ambito commerciale. Invito chi non fosse d’accordo a razionalizzare il successo di GNU Emacs e GCC, i quali, pur non rientrando nell’ambito del software commerciale, hanno comunque avuto un forte impatto sul mercato.
> È giunto il momento di concentrarsi sul futuro di Linux anziché sull’obiettivo distruttivo di arricchirsi alle spese dell’intera comunità Linux e del suo futuro. Può darsi che lo sviluppo e la distribuzione di Debian non siano la risposta ai problemi delineati in questo Manifesto, ma spero che finiranno almeno per attirare sufficiente attenzione su tali problemi in modo che sia possibile risolverli positivamente.[7]

Poco dopo la diffusione del Manifesto, la Free Software Foundation avanzò la prima importante richiesta. Stallman voleva che Murdock chiamasse la sua distribuzione “GNU/Linux”. All’inizio, sostiene Murdock, Stallman aveva proposto il termine “Lignux” – “ossia Linux con GNU al centro” – ma una rapida verifica su Usenet e in diversi ambiti hacker avevano suscitato un tale subisso di fischi da convincere Stallman a optare per il meno bizzarro GNU/Linux.

Anche se qualcuno ha bollato quel tentativo di aggiungere il prefisso “GNU” come una tardiva richiesta di riconoscimento, Murdock lo intese in maniera diversa. Ripensandoci, egli lo vide come uno sforzo per allentare la crescente tensione tra il progetto GNU e gli sviluppatori del kernel Linux. “Si andava delineando una divisione”, rammenta Murdock. “E Richard ne era chiaramente preoccupato.”

La divisione più profonda, aggiunge Murdock, riguardava glibc. Acronimo per GNU C Library, glibc è il pacchetto che consente ai programmatori di fare delle “chiamate di sistema” direttamente al kernel. Nel periodo 1993-1994, glibc si rivelò un problematico collo di bottiglia nello sviluppo di Linux. Considerato l’elevato numero di nuovi utenti decisi ad aggiungere nuove funzioni al kernel, i gestori glibc all’interno del progetto GNU si trovarono

rapidamente sommersi dai loro suggerimenti. Frustrati dai ritardi e dalla crescente reputazione del progetto GNU di essere una palla al piede, alcuni sviluppatori Linux proposero la creazione di un “fork” – ovvero, di una C Library specifica per Linux e parallela a glibc.

Nel mondo hacker, i “fork” costituiscono un fenomeno interessante. Nonostante l’etica hacker consenta a ogni programmatore di fare quel che vuole con il codice di un certo programma, la maggioranza degli hacker preferisce inserire le proprie innovazioni all’interno di un file centrale o “tree” (albero) per assicurarne la compatibilità con i programmi altrui. Operare il “fork” di glibc in questo primo stadio dello sviluppo di Linux avrebbe significato perdere il contributo di centinaia, se non forse migliaia, di sviluppatori. Il che avrebbe inoltre ampliato l’incompatibilità tra Linux e il sistema GNU che Stallman e gli altri del gruppo GNU speravano ancora di realizzare.

In qualità di leader del progetto GNU, nel 1991 Stallman aveva già sperimentato i nefasti effetti di un “fork”. Un gruppo di sviluppatori Emacs, che lavoravano per un’azienda chiamata Lucid, rimasero indispettiti dal suo rifiuto di re-integrare i loro cambiamenti nel codice di GNU Emacs. Il relativo “fork” diede così vita a una versione parallela, Lucid Emacs, provocando pesanti ripercussioni.[8]

Murdock spiega che Debian stava portando a un analogo “fork” nel codice di glibc, così da motivare Stallman a insistere per l’aggiunta del prefisso GNU quando Debian fu pronto per la distribuzione. “Gradatamente la situazione venne ricomposta. Eppure in quel periodo la preoccupazione era che la comunità Linux si considerasse qualcosa di diverso da quella GNU, e ciò poteva sfociare in una netta spaccatura.”

Stallman conferma il ricordo di Murdock. Aggiunge anzi che si andavano profilando dei “fork” tutte le più importanti componenti di GNU. All’inizio sostiene di aver considerato tali “fork” come vino prodotto da uva acerba. Contrariamente alle dinamiche rapide e informali del gruppo al lavoro sul kernel Linux, coloro che mantenevano il codice GNU tendevano a una maggiore lentezza e circospezione nell’inserire cambiamenti che avrebbero

potuto danneggiare le prestazioni del programma a lungo termine. Non dimostravano inoltre alcuna remora a criticare pesantemente il codice altrui. Col passar del tempo, tuttavia, Stallman iniziò a percepire nelle e-mail degli sviluppatori Linux una certa mancanza di considerazione per il progetto GNU e i suoi obiettivi.

"Avevamo scoperto che coloro che si consideravano utenti Linux non erano interessati al progetto GNU", sostiene Stallman. "Dicevano, 'Perché mai dovrei impegnarmi a perseguire queste mete? Non m'importa nulla del progetto GNU. Per me funziona. E se funziona per me e per gli altri utenti Linux, non ci interessa altro'. E questo sorprendeva, considerando che essi usavano una variante del sistema GNU, eppure non se ne preoccupavano. Non avrebbero potuto essere meno interessati."

Mentre c'era chi valutava la definizione di Linux come "variante" del progetto GNU una sorta di appropriazione politica, Murdock, già devoto alla causa del software libero, considerò invece ragionevole la richiesta di Stallman di chiamare GNU/Linux la versione di Debian. "Mirava più al mantenimento dell'unità che a guadagnare dei meriti", sostiene Murdock.

Subito dopo seguirono altre richieste di natura più tecnica. Nonostante Murdock si fosse dimostrato conciliante sulle questioni politiche, rimase fermo sulle proprie posizioni quando si trattò di affrontare il modello di sviluppo del software vero e proprio. Quel che era iniziato come una dimostrazione di solidarietà sfociò presto nella replica delle polemiche interne tipiche di altri progetti GNU.

"Certo mi sono spesso trovato in disaccordo con le sue posizioni", dichiara Murdock con un sorriso. "In tutta onestà, non è affatto facile lavorare con Richard."

Nel 1996, Murdock, dopo aver ottenuto il diploma di laurea alla Purdue, decise di passare ad altri le redini del progetto Debian. Ne aveva già ceduto le responsabilità gestionali a Bruce Perens. Al pari di Murdock, Perens era un programmatore Unix innamoratosi di GNU/Linux non appena divennero chiare le capacità del programma. E come Murdock, Perens aderiva al piano politico di Stallman e della Free Software Foundation, pur se da lontano.

“Ricordo che dopo la diffusione del Manifesto GNU, GNU Emacs e GCC, lessi un articolo in cui si diceva che Stallman lavorava come consulente per Intel”, dice Perens, ricordando il suo primo attrito con Stallman sul finire degli anni ’80. “Gli scrissi per chiedergli come potesse sostenere il software libero da un parte ed essere stipendiato da Intel dall’altra. Mi rispose dicendo, ‘Lavoro come consulente per la produzione di software libero’. Si dimostrò molto educato e chiaro, ritenni la risposta perfettamente sensata.”

Nelle vesti di sviluppatore responsabile di Debian, tuttavia, Perens considerava con sgomento le battaglie tra Murdock e Stallman sulle questioni tecniche. Nell’assumersi la responsabilità del progetto, prese la decisione di prendere le distanze dalla Free Software Foundation. “Decisi che non avremmo aderito allo stile micro-gestionale di Richard”, ricorda.

Secondo Perens, Stallman fu colto alla sprovvista da tale decisione, ma ebbe la saggezza di adeguarvisi. “Lasciò calmare le acque e poi mi inviò un messaggio dicendo che avevamo davvero bisogno di instaurare qualche tipo di relazione. Ci chiese di usare il nome GNU/Linux, senza aggiungere altro. Mi andava bene così. Presi unilateralmente la deliberazione finale, e tutti tirarono un sospiro di sollievo.”

Col passar del tempo, Debian si guadagnò la reputazione di versione hacker di Linux, assieme a Slackware, altra distribuzione molto diffusa fondata nello stesso periodo, 1993-1994. Al di fuori del regno dei sistemi di ambito hacker, Linux andava comunque conquistando spazio nel mercato Unix. In North Carolina, un’azienda Unix che si faceva chiamare Red Hat si apprestava a un rilancio concentrato su Linux. Ne era responsabile Robert Young, l’ex-redattore di Linux Journal, che nel 1994 aveva chiesto a Linus Torvalds se non avesse qualche rimpianto per aver diffuso il kernel sotto GPL. La risposta di Torvalds ebbe un impatto “profondo” sulla considerazione di Linux da parte dello stesso Young. Anziché cercare un modo per aprirsi un varco nel mercato GNU/Linux tramite le tradizionali strategie del software, egli iniziò a considerare il possibile scenario di un’azienda che adottasse il medesimo approccio di Debian – costruire, cioè, un sistema operativo composto interamente da programmi di software libero. Cygnus Solutions, la società

fondata da Michael Tiemann e John Gilmore nel 1990, stava già concretizzando le vendite di software libero basato su funzioni di qualità e aperto alla personalizzazione. Perché non lanciare Red Hat verso una strategia analoga a quella di GNU/Linux?

"Nella tradizione scientifica occidentale si usa salire sulle spalle dei giganti", afferma Young, facendo eco alle parole di Torvalds e prima ancora di Sir Isaac Newton. "In campo imprenditoriale questo significa che non occorre reinventare la ruota man mano che si progredisce. La bellezza del modello [GPL] sta nell'aver reso il codice di pubblico dominio.[9] Per un rivenditore indipendente che voglia costruire applicazioni, per esempio un modem-dialer, be', che senso avrebbe cercare inventarlo dal nulla? Basta rubare PPP da Red Hat Linux e usarlo all'interno della propria configurazione. Se si ha bisogno di strumenti grafici, non è il caso di mettersi a scrivere una libreria grafica propria. Basta scaricare GTK. Improvvisamente abbiamo la possibilità di riutilizzare il meglio di quanto già esiste. E altrettanto improvvisamente il punto focale di un rivenditore di applicazioni si sposta dalla gestione del software alla riscrittura di programmi specificamente tagliati sulle esigenze dei propri utenti."

Young non era certo l'unico manager attratto dall'efficienza imprenditoriale del software libero. Verso la fine del 1996, gran parte delle aziende Unix si stavano svegliando, fiutando le potenzialità offerte dal codice in fase di fermentazione. Il settore Linux era ancora lontano un anno o due dall'esplosione commerciale, ma quelli più vicini alla comunità hacker ne avvertivano i primi sentori: stava per succedere qualcosa di grande. Il chip Intel 386, Internet e il World Wide Web avevano colpito il mercato con una serie di ondate mostruose, e Linux – assieme a una sfilza di programmi analoghi in termini di accessibilità del codice e permissività delle licenze – si annunciava come l'ondata più grande di tutte.

Per Ian Murdock, il programmatore corteggiato da Stallman e poi rimasto deluso dal suo stile micro-gestionale, tale ondata appariva sia come un adeguato tributo sia come una giusta punizione per colui che aveva speso così tanto tempo nel fornire un'identità al movimento del software libero. Come

molti entusiasti di Linux, Murdock aveva letto i messaggi originali. Aveva seguito l'ammonimento iniziale di Torvalds per il quale Linux era "soltanto un hobby". Aveva anche letto l'ammissione dello stesso Torvalds fatta ad Andrew Tanenbaum, creatore di Minix: "Se il kernel GNU fosse stato pronto la primavera scorsa, non mi sarei neppure preso la briga di dar vita al mio progetto".[10] Come molti altri, Murdock era perfettamente consapevole delle buone opportunità sprecate. Sapeva inoltre che l'eccitazione di poter cogliere nuove opportunità trasudava dalla stessa linfa vitale di Internet.

"Fu divertente trovarsi coinvolti in quel primo periodo di Linux", ricorda Murdock. "Offriva qualcosa da fare e, contemporaneamente, un buon modo di passare il tempo. Tornando indietro a leggere quei vecchi scambi [su comp.os.minix], se ne ricava un'impressione generale: ecco qualcosa con cui poter giocare finché non sarà pronto HURD. La gente era in trepida attesa. È buffo, ma sotto molti aspetti credo che Linux non sarebbe mai arrivato a esistere se si fosse giunti al veloce completamento di HURD."

Comunque sia, alla fine del 1996 posizioni tipo "che cosa sarebbe accaduto se…" apparivano abbondantemente superate. Che lo si chiami Linux oppure anche GNU/Linux, gli utenti avevano chiaramente espresso la propria opinione. Quella finestra di 36 mesi si era chiusa, nel senso che anche se il progetto GNU avesse partorito il kernel HURD, con tutta probabilità nessuno, al di fuori della comunità hacker più stretta, se ne sarebbe accorto. Il primo sistema operativo di software libero analogo a Unix era arrivato, e si stava imponendo ovunque. Tutto quello che rimaneva da fare agli hacker era sedersi e attendere che la prossima grossa ondata si infrangesse sulla loro testa. Compresa quella irsuta di un tale Richard M. Stallman.

Pronti o meno che fossero.

[1] Torvalds ha fatto quelle affermazioni in parecchi contesti diversi. Finora tuttavia la citazione più nota è quella che troviamo nel saggio di Eric Raymond, "La cattedrale e il bazaar" (maggio 1997). http://www.apogeonline.com/openpress/doc/cathedral.html

[2] Si veda Simson Garfinkel, "Is Stallman Stalled?", Wired (marzo 1993).

[3] La preoccupazione di Chassel relativa alla “finestra” di 36 mesi per un nuovo sistema operativo non è limitata unicamente al progetto GNU. All’inizio degli anni ’90, la versione libera della Berkeley Software Distribution venne bloccata dalla azione intentata dagli Unix System Laboratories, che mirava a limitare la diffusione di software derivato dalla BSD. Anche se parecchi utenti di prodotti derivati tipo FreeBSD e OpenBSD li considerano effettivamente superiori a GNU/Linux in termini sia di prestazioni che di sicurezza, il loro numero complessivo rimane minimo rispetto al totale della popolazione di GNU/Linux. Per saperne di più sull’analisi del successo di quest’ultimo in relazione ad altri sistemi operativi di software libero, si veda il saggio dell’hacker neozelandese Liam Greenwood, “Why is Linux Successful” (1999). http://www.freebsddiary.org/linux.php

[4] Si veda l’intervento presso il Maui High Performance Computing Center.
In una serie di e-mail, ho chiesto a Stallman che cosa volesse dire con il termine “bug di temporizzazione”. Egli rispose che “errori di temporizzazione” era il modo migliore per sintetizzare il problema e per offrire inoltre una chiara spiegazione tecnica su come errori di questo tipo possano ostacolare il funzionamento di un sistema operativo:
“Errori di temporizzazione” possono verificarsi in sistemi asincroni in cui processi eseguiti in parallelo possono in teoria avere un ordine qualsiasi e uno di questi possibili ordini crea problemi.
Immagina che il programma A esegua X e quello B esegua Y e che X e Y siano brevi routine che esaminano e aggiornano la stessa struttura di dati. Può accadere, anche se di rado, che lo scheduler faccia girare A fino ad eseguire la metà di X, e quindi faccia partire B, che esegue Y. Così Y sarà eseguito per intero, mentre X lo sarà solo a metà. Poiché esse aggiornano la medesima struttura di dati, interferiranno l’una con l’altra. Per esempio X, che forse ha già esaminato la struttura di dati, non noterà che c’è stato un cambiamento. Ci sarà quindi un errore non riproducibile, poiché dipende da fattori casuali (cioè da quando lo scheduler decide quale programma fare girare e per quanto tempo).
Per impedire errori di questo tipo occorre usare un dispositivo di blocco, per assicurarsi che X e Y girino contemporaneamente. I programmatori di sistemi asincroni conoscono l’importanza di tali dispositivi, ma a volte non si accorgono della necessità di porne uno in un determinato luogo o in una specifica struttura di dati. Così il programma contiene un errore di temporizzazione.

[5] Le cifre sulla popolazione di utenti GNU/Linux appaiono piuttosto discordanti, motivo per cui ho scelto un margine così ampio. Il totale di 100.000 è ripreso dal sito di Red Hat “Milestones”: http://www.redhat.com/about/corporate/milestones.html.

[6] Questa analogia con Winston Churchill mi è venuta in mente prima di ricevere dallo stesso Stallman un analogo commento non richiesto:
La Seconda Guerra Mondiale e la determinazione necessaria per vincerla rappresentano importanti ricordi della mia fanciullezza. Affermazioni come quella di Churchill, “Li combatteremo per terra e per mare… non ci arrenderemo mai”, rimangono scolpite nella mia memoria.

[7] Si veda Ian Murdock, “A Brief History of Debian”, (6 gennaio 1994): Appendix A, “The Debian Manifesto”. http://www.debian.org/doc/manuals/project-history/apA.html

[8] Jamie Zawinski, ex-programmatore presso la Lucid poi a capo del gruppo di sviluppo di Mozilla, gestisce un sito web che documenta il “fork” avvenuto tra la Lucid e GNU Emacs, sotto il titolo di “The Lemacs/FSFmacs Schism”. http://www.jwz.org/doc/lemacs.html

[9] L’espressione “pubblico dominio” viene qui usata in maniera poco corretta da Young. Di pubblico dominio significa non protetto da copyright. Per definizione i programmi sotto GPL sono tutelati da copyright.

[10] La citazione è ripresa dalla “flame war”, assai pubblicizzata, avvenuta tra Torvalds e Tanenbaum dopo il lancio iniziale di Linux. Nel difendere la propria scelta di un kernel monolitico non-adattabile, Torvalds sostiene di aver iniziato a lavorare su Linux per saperne di più sul suo nuovo PC 386. “Se il kernel GNU fosse stato pronto la primavera scorsa, non mi sarei neppure preso la briga di dar vita al mio progetto.” Si veda “Il dibattito tra Tanenbaum e Torvalds” in *Open Sources*, Apogeo, 1999, p. 237.

Open Source

Nel novembre 1995, Peter Salus, membro della Free Software Foundation e autore del volume *A Quarter Century of Unix*, uscito nel 1994, lanciò una richiesta per sollecitare i contributi degli iscritti alla mailing list di discussione sul progetto GNU. Salus, che ne era il responsabile di turno, voleva stimolare i colleghi hacker all'approssimarsi della "Conferenza sul software liberamente ridistribuibile" di Cambridge, in Massachusetts. Previsto per il febbraio successivo e sponsorizzato dalla Free Software Foundation, l'evento prometteva di imporsi come il primo di ambito tecnico dedicato esclusivamente al software libero e, a dimostrazione dell'unità con gli altri programmatori del settore, caldeggiava relazioni "su ogni aspetto di GNU, Linux, NetBSD, 386BSD, FreeBSD, Perl, Tcl/tk e altri programmi il cui codice fosse liberamente accessibile e ridistribuibile". Scriveva Salus:

> Nel corso degli ultimi 15 anni, il software libero e a basso costo è arrivato ovunque. Questa conferenza riunirà coloro che hanno implementato i diversi tipi di software liberamente ridistribuibile e gli editori di tale software (su vari supporti). Sono previste sessioni e relazioni su temi specifici, come pure gli interventi di Linus Torvalds e Richard Stallman.[1]

Uno dei primi a ricevere la e-mail di Salus fu Eric S. Raymond, membro del comitato organizzativo della conferenza stessa. Pur non essendo responsabile di alcun progetto o azienda come gli altri iscritti alla mailing list, Raymond si era costruito una solida reputazione nella comunità hacker in qualità di importante collaboratore di GNU Emacs nonché di curatore del The New Hacker Dictionary, versione cartacea del Jargon File, il lessico hacker in circolazione da dieci anni.

Per Raymond, la conferenza del 1996 costituiva un evento importante. Attivo collaboratore del progetto GNU già negli anni '80, se ne era però

allontanato nel 1992, incolpando, come molti altri prima di lui, lo stile “micro-gestionale” tipico di Stallman. “Richard fece un sacco di storie per le mie modifiche non autorizzate mentre stavo ripulendo le librerie della Emacs LISP”, ricorda Raymond. “Ci rimasi così male che decisi che non avrei mai pù lavorato con lui.”

Nonostante la rottura, Raymond continuò la sua attività nella comunità del software libero. A tal punto che, quando Salus suggerì una conferenza con la contemporanea presenza di Stallman e Torvalds come relatori, Raymond assecondò caldamente l’idea. Con Stallman a rappresentare il contingente anziano e saggio degli hacker legati all’ITS/Unix e Torvalds a guidare la generazione più giovane ed energica, la coppia avrebbe simboleggiato un’immagine di unità che non poteva non risultare positiva, specialmente per hacker ambiziosi e giovani (sotto i 40 anni) come Raymond. Il quale non manca di aggiungere: “Potevo sostenere a ragione di avere un piede in entrambe le staffe”.

Con l’approssimarsi dell’incontro, la tensione tra le due fazioni si era fatta palpabile. Ma entrambe avevano un elemento in comune: la possibilità di incontrare in carne e ossa per la prima volta il ragazzo prodigio finlandese. Non senza sorpresa, Torvalds si dimostrò relatore affabile e sicuro. Con un leggero accento svedese, conquistò i presenti grazie alle battute rapide e alla modestia.[2] Ancor più sorprendente, sostiene Raymond, apparve la convinzione con cui Torvalds non mancò di criticare altri noti hacker, compreso il più famoso di tutti, Richard Stallman. Alla fine della conferenza, furono le maniere metà da hacker e metà da scansafatiche di Torvalds ad averla vinta sul pubblico, sia giovane che anziano.

“Fu un momento storico”, rammenta Raymond. “Prima del 1996, Richard era l’unico credibile a potersi candidare come leader ideologico dell’intera cultura. Quelli che dissentivano non lo facevano certo in pubblico. Torvalds riuscì a infrangere quel tabù.”

La definitiva rottura del tabù si manifestò verso la conclusione dell’evento. Nel corso di una discussione sul crescente dominio sul mercato da parte di Windows e tematiche analoghe, Torvalds ammise di ammirare molto

PowerPoint, il software per le presentazioni di diapositive della Microsoft. Dal punto di vista dei puristi della vecchia guardia, era come se un Mormone in chiesa si vantasse di trangugiare whisky. Dal punto di vista di Torvalds e della rampante accolita di seguaci, si trattava semplicemente di buon senso. Perché mai prendersela con il software proprietario solo per principio? Essere un hacker non significava dover soffrire, ma piuttosto fare bene il proprio lavoro.

"Si trattò di un'affermazione decisamente pesante", ricorda Raymond. "Ma, di nuovo, egli se la poteva permettere perché, tra il 1995 e il 1996, stava conquistando punti in maniera vertiginosa."

Stallman, da parte sua, non rammenta alcuna tensione a quella conferenza, ma ricorda di essersi in seguito sentito piccato dalla celebrata sfacciataggine di Torvalds. "A un certo punto la documentazione Linux dice di stampare gli standard del codice GNU per poi strapparli in mille pezzi", dice Stallman, tanto per fare un esempio. "Va bene, non era d'accordo con alcune delle nostre convenzioni. Nulla di male, peccato che abbia deciso di manifestarlo in modo particolarmente odioso. Avrebbe potuto semplicemente spiegare, 'Ecco come credo che si debba indentare il codice'. In questo modo non si sarebbe creata nessuna ostilità."

Per Raymond, la calda accoglienza che gli altri hacker riservarono ai commenti di Torvalds non fece altro che confermare certi sospetti. La linea divisoria che separava gli sviluppatori Linux da quelli GNU/Linux era per lo più generazionale. Parecchi hacker Linux, compreso lo stesso Torvalds, erano cresciuti nel mondo del software proprietario. A meno che un programma risultasse chiaramente peggiore, la maggior parte di loro non vedeva alcun motivo per rifiutarlo solo per il problema della licenza. Da qualche parte nell'universo del software libero stava nascosto un programma che forse un giorno qualche hacker avrebbe trasformato in alternativa a PowerPoint. Ma fino a quel momento, perché invidiare alla Microsoft l'idea di aver realizzato quel programma e di essersene riservati i diritti?

In quanto ex-aderente al progetto GNU, Raymond percepì un'ulteriore dinamica nella tensione tra Stallman e Torvalds. Nei dieci anni successivi al lancio del progetto GNU, la reputazione raggiunta da Stallman faceva paura.

Lo stesso dicasi per la sua intransigenza rispetto alla progettazione del software e alla gestione delle risorse umane. Poco prima di quella conferenza del 1996, la Free Software Foundation avrebbe subìto un abbandono di vaste proporzioni, in gran parte dovuto proprio a Stallman. Brian Youmans, attuale membro dello staff assunto da Salus a seguito di quelle dimissioni in massa, ricorda la scena: “A un certo punto, era rimasto solo Peter [Salus] a lavorare in ufficio”.

Per Raymond, quell’abbandono di massa ribadì un sospetto diffuso: i recenti ritardi, come nel caso di HURD, e le recenti vicissitudini quali lo scisma Lucid-Emacs riflettevano problemi che di solito hanno a che fare con la gestione del progetto, non con lo sviluppo del codice. Poco dopo la “Conferenza sul software liberamente ridistribuibile”, Raymond iniziò a lavorare su un proprio progetto, una utility popmail chiamata “fetchmail”. Seguendo un suggerimento di Torvalds, Raymond distribuì il programma con la relativa promessa di aggiornarne il codice nel modo più rapido e frequente possibile. Quando gli utenti iniziarono a segnalare problemi ed eventuali migliorie, Raymond, che si aspettava di dover sbrogliare una complicata matassa, rimase meravigliato dalla robustezza del software risultante. Analizzando il successo dell’approccio proposto da Torvalds, giunse a una rapida conclusione: usando Internet come “scatola di Petri” e la verifica minuziosa della comunità hacker come forma di selezione naturale, Torvalds aveva creato un modello evolutivo che non prevedeva una pianificazione centralizzata.

Fatto ancora più rilevante, decise Raymond, Torvalds era riuscito ad aggirare la Legge di Brooks. Articolata per la prima volta da Fred P. Brooks, manager del progetto OS/360 dell’IBM e autore nel 1975 del libro *The Mythical Man-Month*, la Legge sostiene che aggiungere nuovi sviluppatori a un progetto finisce per causarne soltanto ulteriori ritardi. Al pari della maggioranza degli hacker, Raymond riteneva che il software, come una zuppa, riusciva meglio se in cucina lavoravano pochi cuochi e quindi percepiva che qualcosa di rivoluzionario si stava profilando all’orizzonte.

Invitando in cucina un numero sempre maggiore di cuochi, in realtà Torvalds era riuscito a produrre un software migliore.[3]

Raymond mise su carta quelle osservazioni, per trasformarle poi in un discorso che tenne subito alla presenza di alcuni amici e vicini a Chester County, Pennsylvania. L'intevento, dal titolo "The Cathedral and the Bazaar" (La cattedrale e il bazaar), metteva a confronto lo stile manageriale del progetto GNU con quello di Torvalds e degli hacker del kernel Linux. Raymond sostiene di aver ricevuto riscontri entusiasti, ma non tali da eguagliare quelli ottenuti durante il Linux Kongress del 1997, raduno di utenti Linux svoltosi in Germania la primavera successiva.

"Al Kongress l'intervento venne salutato con una standing ovation", ricorda Raymond. "La considerai un fatto significativo per due ragioni. Primo, stava a dimostrare l'evidente eccitazione dei presenti per quello che stavano ascoltando. Secondo, l'eccitazione persisteva anche dopo che era stata superata la barriera linguistica."

Alla fine Raymond trasferì quel discorso in un testo scritto, anch'esso intitolato "La cattedrale e il bazaar", a sottolineare l'analogia che costituiva il centro dell'analisi. I programmi GNU sembravano "cattedrali", monumenti all'etica hacker, impressionanti, pianificati in modo centralizzato, costruiti per durare nel tempo. Linux, d'altra parte, era più simile a "un grande bazaar vociante", un programma sviluppato grazie alle dinamiche sciolte e decentrate offerte da Internet.

Implicito in ogni analogia stava il raffronto tra Stallman e Torvalds. Laddove il primo rappresentava il classico modello di architetto di cattedrali – un "mago" della programmazione capace di sparire per 18 mesi per tornare con qualcosa come il GNU C Compiler – il secondo sembrava piuttosto il geniale organizzatore di una festa. Lasciando condurre agli altri la discussione sulla progettazione di Linux e intervenendo soltanto quando si rendeva necessaria la presenza di un arbitro, Torvalds era riuscito a creare un modello di sviluppo che rifletteva decisamente il proprio carattere di persona tranquilla, rilassata. Dal suo punto di vista, il compito manageriale più

importante consisteva non nell'imposizione del controllo bensì nella continua assistenza nel favorire il fluire delle idee.

Raymond conclude, in sintesi: "Credo che l'opera di hacking più sagace e coerente di Linus non sia stata la costruzione del kernel in sé, quanto piuttosto l'invenzione di quel nuovo modello di sviluppo"[4]. Nel riassumere i segreti del successo gestionale di Torvalds, lo stesso Raymond mise a segno un buon colpo. Tra i presenti in sala a quel Linux Kongress c'era Tim O'Reilly, editore specializzato in manuali e libri sul software. Dopo averne seguito l'intervento, O'Reilly lo invitò subito a ripeterlo durante la prima Perl Conference, organizzata dalla casa editrice qualche mese più tardi a Monterey, California.

Anche se centrato su Perl, linguaggio di scripting creato dall'hacker Unix Larry Wall, O'Reilly assicurò Raymond che l'evento si sarebbe occupato anche di altre tematiche connesse con il software libero. Considerando il crescente interesse commerciale nei confronti di Linux e di Apache, un noto server web di software libero, l'incontro voluto da O'Reilly avrebbe pubblicizzato il ruolo del software libero nella creazione dell'intera infrastruttura di Internet. Da linguaggi "web-friendly" come Perl e Python a programmi d'appoggio come BIND (Berkeley Internet Naming Daemon), grazie al quale è possibile sostituire gli arcani numeri degli indirizzi IP con nomi facili da ricordare (tipo Amazon.com), a sendmail, il programma di posta più diffuso su Internet, il software libero stava divenendo qualcosa di più di un fenomeno emergente.

Come in una colonia di formiche intente alla creazione di uno stupendo nido trasportando un granello di sabbia alla volta, l'unico elemento ancora assente in questo scenario era l'autocoscienza comunitaria.

O'Reilly considerò l'analisi di Raymond un ottimo elemento per stimolare tale consapevolezza, per rendere ben chiaro che lo sviluppo del software libero non iniziava e finiva con il progetto GNU. Linguaggi di programmazione tipo Perl e Python, e software per Internet come BIND, sendmail e Apache, stavano lì a dimostrare la diffusione e l'influenza raggiunta dal software libero. Egli garantì inoltre a Raymond un'accoglienza perfino più calorosa di quella ricevuta al Linux Kongress.

O’Reilly aveva ragione. “Stavolta l’ovazione in piedi arrivò ancor prima dell’intervento”, scherza Raymond.

Come previsto, il pubblico era composto non soltanto da hacker, ma anche da altre persone interessate alla rampante crescita del movimento del software libero. Un gruppo lavorava alla Netscape, la startup di Mountain View, in California, a quel tempo prossima a chiudere positivamente la battaglia durata tre anni contro la Microsoft per il controllo del mercato dei web-browser.

Impressionati dal discorso di Raymond e ansiosi di riconquistare la fetta di mercato perduta, i dirigenti di Netscape spiegarono quel messaggio al loro quartier generale. Qualche mese più tardi, nel gennaio 1998, l’azienda annunciò la prossima pubblicazione del codice sorgente del prodotto di punta, il browser Navigator, sperando così di motivare gli hacker a collaborare alle future migliorie.

Quando il responsabile di Netscape Jim Barksdale citò il saggio “La cattedrale e il bazaar” definendolo il principale responsabile di tale decisione, la società elevò immediatamente il suo autore al livello di celebrità hacker. Deciso a non lasciarsi sfuggire l’occasione, Raymond volò in California per concedere interviste, offrire consigli ai dirigenti di Netscape e presenziare alla grande festa in onore della diffusione dei sorgenti di Navigator. Questi vennero riuniti sotto il nome di “Mozilla”, con riferimento sia alla mole mastodontica del programma – 30 milioni di righe di codice – sia al suo asse ereditario. Sviluppato come derivato proprietario di Mosaic, il browser originariamente creato da Marc Andreessen alla University of Illinois, Mozilla costituiva la prova, ancora un volta, del fatto che quando si tratta di realizzare nuovi programmi la maggioranza degli sviluppatori preferisce operare su versioni già esistenti e modificabili.

Mentre si trovava in California, Raymond riuscì inoltre a fare un salto alla VA Research, azienda di Santa Clara che commercializzava workstation con il sistema operativo GNU/Linux pre-installato. Come voleva Raymond, la riunione era ristretta al fondatore della società Larry Augustin, ad alcuni dipendenti e a Christine Peterson, presidente del Foresight Institute, organizzazione di Silicon Valley specializzata in nanotecnologie.

“La scaletta dell’incontro si ridusse a un unico tema: come approfittare della decisione di Netscape in modo che altre aziende potessero seguirne l’esempio?” Raymond non rammenta la conversazione che ne seguì, ma ricorda le prime lamentele. Nonostante tutti gli sforzi da parte di Stallman e di altri hacker per spiegare alla gente che la parola “free” in “free software” andava intesa come sinonimo di “libero” non di “gratuito”, il messaggio stentava a passare. Gran parte dei dirigenti di società, nell’imbattersi per la prima volta in quel termine, lo interpretava come sinonimo di “a costo zero”, lasciando così cadere ogni possibile iniziativa. Fino a quando gli hacker non avessero trovato il modo di superare quella dissonanza cognitiva, il movimento del software libero avrebbe dovuto affrontare un’ardua scalata, anche dopo l’uscita di Netscape.

La Peterson, la cui organizzazione svolgeva un ruolo attivo nel sostegno della causa del software libero, propose un termine alternativo: open source.

Ripensandoci, la Peterson sostiene di aver pensato a quel termine mentre discuteva sulla decisione di Netscape con un’amica che si occupava di pubbliche relazioni. Non ricorda dove si fosse imbattuta in quella definizione oppure se si riferisse a qualche altro ambito, ma rammenta che all’amica non piacque affatto.[5]

A quella riunione, aggiunge, la reazione fu diametralmente opposta: “Esitavo a suggerirlo. Non avevo alcuna voce in quel gruppo, così iniziai a usarlo in maniera casuale, senza evidenziare che si trattava di un termine nuovo”. Con sua stessa sorpresa, il termine sembrò imporsi. A fine riunione quasi tutti i presenti, compreso Raymond, sembravano apprezzarlo.

Raymond afferma di non aver usato in pubblico il termine “open source” come sostituto di “free software” fino a uno o due giorni dopo la festa per il lancio di Mozilla, in occasione di un incontro organizzato da O’Reilly per discutere ulteriormente sul software libero. Chiamando la riunione “Freeware Summit”, O’Reilly dice di aver voluto dirigere l’attenzione dei media e della comunità su altri importanti progetti che avevano incoraggiato Netscape alla diffusione di Mozilla. “Tutte queste persone avevano tante cose in comune, che ero sorpreso che non si conoscessero personalmente” spiega O’Reilly.

“Volevo inoltre informare il mondo sul grande impatto già provocato dalla cultura del software libero. La gente ignorava gran parte della sua tradizione.”

Nel mettere insieme l’elenco degli invitati, tuttavia, O’Reilly prese una decisione che avrebbe provocato conseguenze politiche a lungo termine. Scelse di invitare soltanto gli sviluppatori della costa occidentale, tra cui Wall, Eric Allman, creatore di sendmail, e Paul Vixie, autore di BIND. Non mancavano naturalmente le eccezioni: Raymond, residente in Pennsylvania ma già in zona per il lancio di Mozilla, guadagnò rapidamente l’invito; lo stesso dicasi per Guido van Rossum, abitante in Virginia e creatore di Python. “Frank Willison, redattore capo e punto di riferimento per Python in azienda, lo invitò senza neppure consultarmi”, ricorda O’Reilly. “Fui contento della sua presenza, ma quando la cosa fu concepita si trattava soltanto di un raduno a livello locale.”

Per alcuni osservatori la scelta di non inserire anche il nome di Stallman rappresentò un vero e proprio affronto. “Proprio per questo decisi di non andarci”, afferma Perens ricordando quell’incontro. Raymond, che invece ci andò, dice di aver insistito invano perché fosse invitato. Le voci di una ripicca trovano ulteriore conferma nel fatto che O’Reilly, organizzatore dell’evento, si era reso protagonista di un pubblico alterco con Stallman sulla questione del copyright per i manuali dei programmi. Poco tempo prima di quell’incontro, Stallman aveva sostenuto che i manuali di software avrebbero dovuto poter essere copiati e modificati liberamente al pari dei relativi programmi. O’Reilly ribatteva invece che il mercato a valore aggiunto dei volumi non-liberi avrebbe accresciuto l’utilità del software libero estendendone l’accesso a una comunità più vasta. I due si erano trovati in disaccordo anche sul titolo dell’evento, con Stallman che insisteva per “Free Software” contro “Freeware”, termine politicamente meno significativo.

Pur col senno di poi, O’Reilly non considera un affronto la decisione di lasciar fuori Stallman. “Allora non l’avevo mai incontrato di persona, ma nelle interazioni via e-mail si era dimostrato inflessibile e refrattario al dialogo. Per assicurarmi che la tradizione GNU venisse rappresentata in quella riunione, invitai John Gilmore e Michael Tiemann, che conoscevo personalmente e

sapevo essere appassionati sostenitori dei valori della GPL, ma che sembravano più disposti a dialogare sui punti di forza e di debolezza dei vari progetti di software libero e delle relative tradizioni. Considerati i recenti dissapori, vorrei aver invitato anche Richard, ma non credo certo che il non averlo fatto vada interpretato come una mancanza di rispetto per il progetto GNU o per Richard a livello personale."

Che si sia trattato o meno di un affronto, O'Reilly e Raymond confermano entrambi che il termine "open source" ottenne l'approvazione di un numero di convenuti tale da potersi definire un successo. I presenti all'incontro condivisero idee ed esperienze su come migliorare l'immagine del software libero. Punto chiave risultò il modo di mettere in evidenza pubblicamente i suoi successi, particolarmente nell'ambito dell'infrastruttura di Internet, anziché insistere sulla sfida di GNU/Linux contro Windows. Ma come accadde nella precedente riunione presso VA Research, la discussione finì presto per ruotare sui problemi associati al termine "free software". O'Reilly rammenta un commento particolarmente ficcante di Torvalds, anch'egli presente al summit.

"A quel tempo Linus si era appena trasferito nella Silicon Valley, e spiegò come soltanto recentemente fosse venuto a conoscenza del doppio significato del termine 'free' in inglese – per indicare sia 'libero che 'gratuito'."

Michael Tiemann, fondatore di Cygnus, propose "sourceware" come alternativa a quel termine problematico. "Nessuno ne parve entusiasta", ricorda ancora O'Reilly. "Fu allora che Eric buttò lì 'open source'."

Anche se alcuni lo trovarono interessante, la decisione di cambiare ufficialmente terminologia apparve tutt'altro che unanime. Alla fine di un'intera giornata di discussione, si decise di mettere ai voti i termini in ballo – free software, open source e sourceware. Secondo O'Reilly, in nove su quindici scelsero "open source". Anche se non tutti sembravano convinti, si raggiunse l'accordo generale di adottare quel termine nei futuri incontri con la stampa. "Volevamo diffondere un messaggio solidale", conclude O'Reilly.

Non ci volle molto perché il termine entrasse nel lessico nazionale. Poco dopo quella riunione, O'Reilly ne chiamò a raccolta i partecipanti per una

conferenza stampa a cui erano presenti giornalisti del New York Times, del Wall Street Journal e di altre importanti testate. Nel giro di qualche mese, il volto di Torvalds compariva sulla copertina della rivista Forbes, mentre nelle pagine interne c'erano quelli di Stallman, Larry Wall (creatore di Perl) e Brian Behlendorf, leader del gruppo Apache. Fu così che l'open source venne lanciato in orbita.

Per alcuni tra i presenti al summit, come Tiemann, la cosa più importante era il messaggio di solidarietà. Nonostante la propria azienda avesse raggiunto un discreto successo commercializzando strumenti e servizi di software libero, egli rimaneva cosciente delle difficoltà in cui si dibattevano altri programmatori e imprenditori.

"È fuor di dubbio che l'utilizzo della parola 'free' abbia creato parecchia confusione", spiega Tiemann. "Il termine 'open source' si posizionava in modo amichevole e sensibile nei confronti del mondo imprenditoriale, mentre 'free software' voleva essere moralmente corretto. Nel bene o nel male, ritenemmo più vantaggioso allinearci con quanti optarono per 'open source'."

La risposta di Stallman alla nuova terminologia si fece attendere. Raymond sostiene che Richard considerò per qualche tempo la possibilità di adottarla, per poi decidere di rifiutarla. "Lo so perché ne discutemmo in maniera specifica", afferma Raymond.

Alla fine del 1998 Stallman aveva chiarito la propria posizione: open source, pur risultando utile nel comunicare i vantaggi tecnici del software libero, al contempo finiva per allontanarsi dalla questione della libertà nel software. Considerando questo un aspetto negativo, Stallman avrebbe continuato a usare il termine "free software".

Riassumendo quella posizione nel corso del LinuxWorld del 1999, evento definito dallo stesso Torvalds come la festa per il lancio pubblico della comunità Linux, Stallman implorò i colleghi hacker a opporre resistenza alle lusinghe dei facili compromessi.

"Dopo aver dimostrato di che cosa siamo capaci, non dobbiamo sentirci disperati al punto da farci assumere in qualche azienda o da scendere a patti sui nostri obiettivi", spiegò Stallman durante una tavola rotonda. "Aspettiamo

che siano loro a farci un’offerta e allora accetteremo. Non dobbiamo cambiare la nostra prassi per ottenere qualche aiuto. Si può procedere un passo dopo l’altro, e pian piano ci troveremo a raggiungere l’obiettivo. Oppure si può decidere di fare un mezzo salto in avanti, con la conseguenza che non faremo più un altro passo e non toccheremo mai la meta.”

Ancor prima del LinuxWorld, tuttavia, Stallman pareva orientato a inimicarsi anche i colleghi più concilianti. Alcuni mesi dopo il Freeware Summit, fu la volta della seconda Perl Conference organizzata da O’Reilly. Stavolta Stallman era presente tra il pubblico. Durante una tavola rotonda in cui si lodava l’IBM per aver inserito il web server di software libero Apache nelle proprie offerte commerciali, Stallman, approfittando di un microfono riservato agli interventi dalla platea, interruppe la discussione lanciandosi in una tirata contro uno degli ospiti sul palco, John Ousterhout, creatore del linguaggio di scripting Tcl. Stallman lo accusò di essere un “parassita” della comunità del software libero per aver commercializzato una versione proprietaria di Tcl tramite la Scriptics, la start-up da lui stesso fondata. “Non credo che la Scriptics sia necessaria per tenere in vita Tcl”, affermò Stallman tra i fischi di protesta dei colleghi in sala.[6]

“Fu una scena piuttosto sgradevole”, ricorda Rich Morin della Prime Time Freeware. “John aveva realizzato ottime cose: Tcl, Tk, Sprite. Si era dimostrato un prezioso collaboratore.” Nonostante le simpatie per Stallman e per le sue posizioni, Morin si sentì vicino alle vittime di quel comportamento scostante.

La sfuriata di Stallman alla Perl Conference provocò il temporaneo allontanamento di un altro potenziale alleato, Bruce Perens. Nel 1998 Eric Raymond propose il lancio della Open Source Initiative, ossia OSI, organizzazione che aveva lo scopo di monitorare il corretto uso del termine “open source” e di fornire una definizione adeguata del concetto per le aziende interessate a realizzare programmi propri. Raymond ingaggiò Perens per stilare il testo di tale definizione.[7]

Successivamente Perens si sarebbe dimesso dalla OSI, esprimendo rincrescimento per il fatto che l’entità finisse per operare in opposizione a

Stallman e alla FSF. Eppure, riflettendo sulla necessità di trovare in quel momento una definizione al di fuori degli auspici della stessa Free Software Foundation, Perens comprese il motivo dell'ulteriore presa di distanza da parte degli altri hacker. "Richard mi piace davvero, lo ammiro", affermò Perens. "Ma credo che svolgerebbe meglio il proprio ruolo se mostrasse maggiore equilibrio. Per esempio se si prendesse qualche periodica vacanza di un paio di mesi dal software libero."

Gli sforzi monomaniaci di Stallman avrebbero contribuito ben poco a rintuzzare il successo di relazioni pubbliche conquistato da chi proponeva l'open source. Nell'agosto 1998, quando il produttore di chip Intel acquistò un pacchetto azionario in Red Hat, distributore di GNU/Linux, un articolo di spalla del New York Times descrisse l'azienda come il prodotto del movimento "noto alternativamente come software libero e open source".[8] Sei mesi dopo, il titolo in testa a un articolo di John Markoff sulla Apple Computer proclamava l'adozione da parte di quest'ultima del server "open source" Apache.[9]

Un simile successo coincideva con la rampante crescita di società che avevano attivamente abbracciato il termine "open source". Nell'agosto '99 Red Hat, azienda che oggi vanta con orgoglio il titolo di "open source", iniziò a vendere propri titoli al Nasdaq. In dicembre, VA Linux – già nota come VA Research – stava pompando il proprio ingresso a Wall Street a livelli storici. Aprendo con 30 dollari ad azione, le quotazioni esplosero rapidamente oltre quota 300, per poi assestarsi su 239 dollari per azione. Gli azionisti così fortunati da esser partiti al livello più basso e aver resistito fino in fondo avrebbero registrato un aumento del 698% del valore cartaceo, record assoluto per il Nasdaq.

Tra questi fortunati azionisti si annoverava anche Eric Raymond, il quale, membro del direttivo aziendale fin dal lancio di Mozilla, aveva ricevuto 150.000 azioni di VA Linux. Colpito dal fatto che quel saggio sui contrastanti stili manageriali di Stallman e Torvalds gli avesse fruttato qualcosa come 36 milioni di dollari, almeno sulla carta, decise di dargli un seguito. Nel suo

nuovo saggio, Raymond rifletteva sul rapporto tra etica hacker e ricchezza monetaria:

> In questo periodo capita spesso che i giornalisti mi chiedano se la comunità open source finirà per essere corrotta da questo grande flusso di denaro. Rispondo per come la vedo io: la domanda di programmatori da parte delle aziende è stata così intensa e così a lungo che quanti potevano essere seriamente distratti dal denaro hanno già fatto i bagagli. La nostra comunità ha prodotto un'autoselezione interna sulla base dell'attenzione riservata ad altri elementi – risultati concreti, orgoglio, passione artistica e comunanza.[10]

Che riescano o meno a dissipare il sospetto che Raymond e altri proponenti dell'open source abbiano agito per denaro, tali commenti paiono comunque chiarire il messaggio centrale dell'intera comunità: tutto quel che serve per vendere l'idea del software libero è un volto simpatico e un messaggio sensato. Anziché combattere il mercato frontalmente, come aveva fatto Stallman, Raymond, Torvalds e gli altri nuovi leader della comunità hacker avevano adottato un atteggiamento più rilassato, ignorando il mercato in certi settori, facendo leva su di esso in altri. Invece di giocare il ruolo degli studenti emarginati, avevano scelto quello delle celebrità, ampliando nel frattempo il proprio potere personale.

"Nelle sue giornate peggiori, Richard ritiene che io e Linus Torvalds abbiamo cospirato per dirottare la sua rivoluzione", sostiene Raymond. "Secondo me, il rifiuto del termine 'open source' e la deliberata creazione di un confronto ideologico da parte sua, derivano da uno strano miscuglio di idealismo e di territorialità. Parecchia gente è convinta che sia tutta colpa dell'ego personale di Richard. Non credo sia così. C'entra di più il fatto che, identificandosi a tal punto con l'idea del software libero, considera qualunque posizione diversa su questo punto alla stregua di una minaccia personale."

Ironicamente però il successo dell'open source e dei suoi sostenitori, come Raymond, non ha comportato il ridimensionamento di Stallman nel suo ruolo di leader. Sembra anzi fornirgli nuovi seguaci da convertire. Eppure le accuse di territorialità avanzate da Raymond rivelano una certa fondatezza. Non si contano le situazioni in cui Stallman è rimasto tutto d'un pezzo più per abitudine che per i principi in gioco: il suo iniziale disinteresse per il kernel

Linux, ad esempio, e la sua attuale opposizione, in quanto figura politica, ad avventurarsi al di fuori del regno del software.

Eppure, di nuovo, come conferma anche il recente dibattito sull'open source, i casi in cui Stallman si è dimostrato intransigente generalmente gli hanno consentito di guadagnare terreno. "Uno dei suoi maggiori tratti caratteriali è che non molla mai", dice Ian Murdock. "Se occorre, è capace di aspettare anche fino a dieci anni per vedere qualcuno aderire alle proprie posizioni."

Da parte sua Murdock considera questa natura così testarda un elemento rinfrescante e prezioso. Può anche darsi che Stallman non possa più considerarsi leader incontrastato del movimento del software libero, ma rimane comunque la calamita dell'intera comunità. "Puoi esser certo che le sue posizioni rimarranno sempre coerenti", conclude Murdock. "La maggior parte di noi invece non è così. Che si sia d'accordo o meno con le sue posizioni, questo merita indubbio rispetto."

[1] Si veda Peter Salus, "FYI-Conference on Freely Redistributable Software, 2/2, Cambridge", (1995) (archiviato da Terry Winograd). http://hci.stanford.edu/pcd-archives/pcd-fyi-/1995/0078.html

[2] Nonostante Linus Torvalds sia finlandese, la sua lingua madre è lo svedese. Il documento "The Rampantly Unofficial Linus FAQ" ne offre una breve spiegazione:
In Finlandia vive una minoranza significativa (circa il 6%) di popolazione che parla svedese. Definendosi "finlandssvensk" o "finlandssvenskar", essa si considera finlandese; molte di queste famiglie hanno vissuto in Finlandia per secoli. Lo svedese è una delle due lingue ufficiali della Finlandia.
http://tuxedo.org/~esr/faqs/linus/

[3] La legge di Brooks può considerarsi la sintesi della seguente citazione tratta dal suo libro:
Poiché la costruzione del software è intrinsecamente un lavoro sistemico – un esercizio di complesse interrelazioni – richiede un notevole sforzo comunicativo, e ciò finisce per imporre la rapida diminuzione del tempo di lavoro individuale causato dall'ulteriore suddivisione dello stesso. Aggiungere nuove persone quindi allunga i tempi, invece di accorciarli.
Si veda Fred P. Brooks, *The Mythical Man-Month*, Addison Wesley Publishing, 1995.

[4] Si veda Eric Raymond, "La cattedrale e il bazaar", (1997). http://www.apogeoline.com-/openpress/doc/cathedral.html

[5] Si veda Malcolm Maclachlan, "Profit Motive Splits Open Source Movement", *TechWeb News*, 26 agosto 1998. http://content.techweb.com/wire/story/TWB19980824S0012

[6] Si veda Malcolm Maclachlan, “Profit Motive Splits Open Source Movement”, *TechWeb News*, 26 agosto 1998. http://content.techweb.com/wire/story/TWB19980824S0012

[7] Si veda Bruce Perens et al., “The Open Source Definition”, The Open Source Initiative (1998). http://www.opensource.org/docs/definition.html

[8] Si veda Amy Harmon, “For Sale: Free Operating System”, *New York Times*, 28 settembre 1998. http://www.nytimes.com/library/tech/98/09/biztech/articles/28linux.html

[9] Si veda John Markoff, “Apple Adopts ‘Open Source’ for its Server Computers”, *New York Times*, 17 marzo 1999. http://www.nytimes.com/library/tech/99/03/biztech/articles-/17apple.html

[10] Si veda Eric Raymond, “Surprised by Wealth”, Linux Today (10 dicembre 1999). http://linuxtoday.com/news_story.php3?ltsn=1999-12-10-001-05-NW-LF

Una breve discesa nell’inferno hacker

Richard Stallman guarda fisso davanti a sé alla guida di una macchina a noleggio, aspettando che il semaforo diventi verde, mentre ci troviamo nel centro di Kihei.

Siamo diretti alla vicina cittadina di Pa’ia, dove ci incontreremo fra circa un’ora con un gruppo di programmatori e relative consorti per andare a cena tutti insieme.

Sono trascorse ormai due ore dal discorso di Stallman al Maui High Performance Center, e Kihei, luogo apparentemente così invitante prima dell’evento, ora sembra profondamente ostile. Come molte città costruite sulla spiaggia, Kihei non è altro che una sfilza unidimensionale di agglomerati suburbani. Guidando sull’arteria principale, affiancata da un’interminabile sequenza di chioschi dove si vendono panini, agenzie immobiliari e negozi di bikini, è difficile non sentirsi come un boccone, protetto da un abitacolo di metallo, che percorre l’apparato digerente di un gigantesco verme solitario. Una sensazione esacerbata dalla mancanza di strade laterali. Si può soltanto andare dritti, e così il traffico si muove a piccoli sbalzi. 500 metri più avanti il semaforo diventa verde. Quando riusciamo a muoverci è tornato nuovamente giallo.

Per Stallman, da sempre residente sulla costa orientale degli Stati Uniti, la prospettiva di dover trascorrere la parte migliore di un assolato pomeriggio hawaiiano intrappolato in un traffico da lumaca, è sufficiente a provocare un attacco apoplettico. Ancor peggio, sapendo che sarebbe bastato girare a destra all’incrocio giusto, quattrocento metri prima, per evitare questa situazione. Purtroppo siamo alla mercé dell’autista davanti a noi, un programmatore

locale che conosce la strada e che ha deciso di farci passare per l'itinerario panoramico di Pa'ia invece di imboccare la parallela Pilani Highway.

"È spaventoso", si scalda Stallman, al colmo della frustrazione. "Perché mai non abbiamo preso l'altra strada?"

Ecco diventare di nuovo verde il semaforo qualche centinaio di metri più avanti. E ancora un volta riusciamo ad avanzare appena di qualche macchina. La situazione si protrae per altri dieci minuti, fino a quando raggiungiamo finalmente l'incrocio che promette accesso all'adiacente autostrada.

La macchina davanti decide di ignorarlo e procede diritta.

"Ma perché non gira?" mugugna Stallman, alzando le braccia al cielo in segno di frustrazione. "È incredibile!"

Decido anch'io di ignorare la situazione. Mi pare già altrettanto incredibile il fatto di trovarmi a Maui, in un'autovettura guidata da Stallman. Fino a due ore prima non sapevo neppure che sapesse guidare. Adesso, ascoltando un pezzo di musica folk, con le note basse di un violoncello emesse dallo stereo della macchina e osservando il tramonto alla nostra sinistra, faccio del mio meglio per sprofondare nel sedile.

Quando finalmente si presenta un'altra opportunità per girare, Stallman mette la freccia a destra tentando di attirare l'attenzione della macchina davanti. Niente da fare. Ancora una volta superiamo lentamente l'incrocio, per poi bloccarci a quattrocento metri buoni dal semaforo successivo. A questo punto Stallman diventa livido di rabbia.

"Sembra che abbia deliberatamente deciso di ignorarci", sbotta, gesticolando furiosamente come l'addetto all'atterraggio di un aereo, nel vano tentativo di richiamare l'attenzione della nostra guida. La quale non si scompone minimamente, e per i cinque minuti successivi tutto quello che riusciamo a vedere è una piccola porzione del suo volto nello specchietto retrovisore.

Guardo fuori dal finestrino di Stallman. Le vicine isole Kahoolawe e Lanai fanno da cornice ideale per il calar del sole. Una vista mozzafiato, di quelle che consentono ai locali di sopportare meglio momenti simili, credo. Tento di

spostare verso quel panorama l'attenzione di Stallman, ma non mi dà retta, ossessionato com'è dalla disattenzione dell'autista davanti.

Quando questo supera l'ennesimo semaforo verde, disinteressandosi del cartello "Pilani Highway" sulla destra, mi trovo a digrignare i denti. Mi viene in mente un avvertimento che mi aveva dato tempo addietro Keith Bostic, programmatore del BSD. "Stallman non può sopportare le stupidate. Se una persona fa o dice qualcosa di stupido, non ci pensa due volte a guardarla dritto negli occhi e dichiarare, 'È una cretinata'."

Osservando l'ignaro autista che ci precede, mi rendo conto come sia proprio quella stupidità, non tanto la circostanza poco simpatica, a far dannare in quel momento Stallman.

"Sembra che abbia scelto la strada senza assolutamente pensare al modo più efficiente per arrivare a destinazione", insiste Stallman.

La parola "efficiente" rimane nell'aria come un cattivo odore. Sono poche le cose che irritano la mente hacker più dell'inefficienza. Fu l'inefficienza di dover andare a verificare di persona due o tre volte al giorno la stampante laser della Xerox che spinse Stallman a occuparsi una prima volta del suo codice. Fu l'inefficienza di dover riscrivere programmi contrabbandati dai rivenditori di software commerciale che lo convinsero a opporsi alla Symbolics e a lanciare il progetto GNU. Se l'inferno sono gli altri, come opinò una volta Jean Paul Sartre, l'inferno hacker sta nella duplicazione degli stupidi errori altrui, e non è esagerato affermare che l'intera esistenza di Stallman è stata un tentativo di salvare l'umanità da simili abissi infuocati.

La metafora dell'inferno appare ancora più appropriata mentre attraversiamo lentamente la scena locale. Con quella moltitudine di negozi, parcheggi e semafori non coordinati, più che a una città Kihei assomiglia a un enorme software mal progettato. Invece di dirigere il traffico e distribuire il flusso di veicoli lungo strade laterali e tangenziali, i pianificatori cittadini hanno deciso di far transitare tutti i veicoli lungo un'unica arteria principale. Dal punto di vista dell'hacker, starsene seduti in macchina in mezzo a quella gran confusione è come ascoltare un CD a tutto volume che riproduca lo stridio delle unghie su una lavagna.

“Sono i sistemi imperfetti a far infuriare gli hacker”, osserva Steven Levy, un’ulteriore affermazione a cui avrei dovuto prestare orecchio prima di salire in macchina con Stallman. “È questa una delle ragioni per cui in genere gli hacker detestano guidare – quell’insieme mal programmato di semafori rossi e di assurdi sensi unici provoca ritardi così dannatamente inutili da suscitare l’impulso di riorganizzare la segnaletica, aprire le scatole di controllo dei semafori… ridisegnare l’intero sistema.”[1]

Ancor più frustrante appare tuttavia l’atteggiamento incerto del nostro fidato accompagnatore. Anziché tentare qualche rapida scorciatoia – come avrebbe istintivamente fatto un vero hacker – l’autista che ci precede ha invece deciso di rispettare le regole del gioco volute da chi ha pianificato la città. Come Virgilio nell’Inferno dantesco, la nostra guida è ben decisa a offrirci una visita completa di questo inferno hacker, che lo vogliamo o no.

Prima di poter riferire l’osservazione a Stallman, ecco finalmente che la macchina davanti mette la freccia a destra. Le spalle ingobbite di Stallman si rilassano un po’, e per un attimo la tensione sparisce. Per tornare subito dopo, quando la macchina rallenta. Ai lati della carreggiata compaiono dei cartelli con la scritta “lavori in corso”, e anche se la Pilani Highway si trova a poche centinaia di metri, la strada a due corsie che ci separa dall’imbocco autostradale è bloccata da un bulldozer a riposo e da due grandi cumuli di terra.

Ci vuole qualche secondo prima che Stallman si renda conto di quanto accade, mentre la nostra guida inizia a compiere una serie di ardue manovre per fare la conversione a U. Quando anche Stallman si accorge del bulldozer e subito dietro della segnaletica che avvisa “Strada chiusa”, non riesce più a trattenersi.

“Perché, perché, perché?” inizia a lamentarsi gettando all’indietro la testa. “Avresti dovuto saperlo che la strada era bloccata. Avresti dovuto sapere che di qua non si passava. L’hai fatto apposta!”

L’autista davanti conclude la manovra e ci passa di fianco, tornando indietro sulla strada principale, mentre scuote la testa chiedendoci scusa con una scrollata di spalle. Assieme a un sorriso appena abbozzato, il gesto rivela un

po’ della frustrazione tipica di chi vive sulla terraferma, mitigato però dalla protettiva dose di fatalismo di chi è abituato vivere su quell’isola. Passando vicino al finestrino chiuso della nostra macchina a noleggio, lancia una rapida battuta: “Siamo a Maui, che ci vuoi fare?”

A quel punto Stallman perde le staffe.

“Non c’è niente da ridere” urla, sbuffando sul finestrino. “È solo colpa tua. Sarebbe stato tutto molto più semplice se avessimo fatto a modo mio.”

Stallman sottolinea le parole “a modo mio” aggrappandosi al volante e chinandosi su di esso due volte. Sembra un bambino che ha un attacco d’ira in macchina, come sottolinea ulteriormente il tono della voce. A metà tra la rabbia e l’angoscia, sembra sul punto di scoppiare in lacrime.

Fortunatamente si trattiene. Come un acquazzone estivo, l’attacco d’ira finisce rapidamente come era iniziato. Dopo qualche altro sospiro, Stallman mette la retromarcia e inizia la manovra di conversione. Non appena riprendiamo la strada, il suo volto torna a farsi impassibile come quando abbiamo lasciato l’albergo mezz’ora prima.

In meno di cinque minuti raggiungiamo l’incrocio successivo. Da qui imbocchiamo facilmente l’autostrada e in pochi secondi viaggiamo verso Pa’ia a una velocità ragionevole. Il sole che poco prima appariva giallo e brillante sopra la spalla sinistra di Stallman ora brucia di un bel rosso-arancione nello specchietto retrovisore, inondando del suo colore le due file di alberi locali disposti lungo i lati dell’autostrada.

Per i successivi venti minuti, oltre al mormorio del motore e delle gomme sull’asfalto, in macchina si sente soltanto la musica di un trio di violini e violoncelli che si cimenta nelle dolenti melodie di un motivo folk dei Monti Appalachi.

[1] Si veda Steven Levy, *Hackers*, Penguin USA, 1984, p. 40.

La lotta continua

Nel caso di Richard Stallman, se è vero che il tempo non riesce a lenire tutte le ferite, si dimostra comunque un potente alleato.

Quattro anni dopo l'uscita de "La cattedrale e il bazaar", egli prova ancora irritazione per le critiche di Raymond. Né manca di lamentarsi per l'elevazione di Linus Torvalds al ruolo di hacker più famoso del mondo. Al riguardo ricorda una maglietta che iniziò a circolare diffusamente nelle mostre commerciali di Linux verso il 1999. Ricalcando il manifesto promozionale originale di Star Wars, nel disegno si vedeva Torvalds brandire una spada luminosa come Luke Skywalker, mentre sul corpo del robot R2D2 c'era la testa di Stallman. Una cosa che gli dà ancora sui nervi, non soltanto perché lo presenta come spalla di Torvalds, ma anche perché assegna a quest'ultimo la leadership della comunità free software/open source, un ruolo che lo stesso Torvalds appare assai restio ad accettare. "È buffo", fa notare Stallman con una punta di tristezza. "Raccogliere quella spada è esattamente l'azione che Linus rifiuta di compiere. Prima fa in modo che tutti lo considerino il simbolo del movimento, e poi non vuole combattere. Che cosa c'è di buono in questo comportamento?"

Ma è esattamente il rifiuto di "raccogliere la spada" da parte di Torvalds che ha lasciato la porta aperta a Stallman per confermare la propria reputazione di pilastro etico della comunità hacker. Nonostante le lamentele, Stallman deve ammettere che gli ultimi anni hanno portato frutti positivi, sia per lui che per la sua organizzazione.

Relegato ai margini dall'imprevisto successo di GNU/Linux, Stallman è riuscito comunque a riconquistare efficacemente l'iniziativa. Il suo programma di interventi pubblici dal gennaio 2000 al dicembre 2001

comprendeva tappe in sei continenti e una serie di incontri in quei paesi dove la nozione di software libero riveste un significato tutto particolare – Cina e India, ad esempio.

Oltre alle prediche dal pulpito, Stallman è inoltre riuscito a far leva sul prestigio personale di custode della GNU General Public License (GPL). Durante l'estate 2000, quando anche per Linux il boom borsistico dell'anno precedente si era sgonfiato, Stallman e la Free Software Foundation misero a segno due importanti vittorie. A luglio la Troll Tech, azienda norvegese produttrice di Qt, valido pacchetto di strumenti grafici per il sistema operativo GNU/Linux, annunciò la distribuzione del software sotto la GPL. Qualche settimana più avanti, la Sun Microsystems, società che fino a quel momento era saltata con molta prudenza sul treno dell'open source, senza rinunciare al controllo totale del proprio software, aveva finalmente ceduto annunciando che anch'essa avrebbe diffuso il nuovo pacchetto OpenOffice sotto due licenze diverse, la Lesser GNU Public License (LGPL) e la Sun Industry Standards Source License (SISSL).

Vittorie tanto più importanti dal momento che Stallman non si è adoperato granché per ottenerle. Nel caso della Troll Tech, gli è bastato giocare il ruolo di gran sacerdote del software libero. Nel 1999 l'azienda norvegese aveva messo a punto una licenza che recepiva le condizioni della Free Software Foundation ma, esaminandola più a fondo, Stallman scoprì delle incompatibilità legali che avrebbero reso impossibile inserire il programma Qt all'interno di altro software tutelato dalla GPL. Sfiniti dal confronto con Stallman, alla fine i dirigenti della Troll Tech decisero di realizzare due distinte versioni, una sotto GPL e l'altra sotto QPL, consentendo così agli sviluppatori di aggirare le questioni di compatibilità citate da Stallman.

La Sun, da parte sua, aveva piena intenzione di aderire alle condizioni della Free Software Foundation. Nel corso della O'Reilly Open Source Conference del 1999, il co-fondatore nonché responsabile della ricerca, Bill Joy, difese la licenza dei "sorgenti comunitari" stilata dalla società, una sorta di compromesso annacquato che consentiva agli utenti di copiare e modificare il software proprietario, senza però poter imporre alcuna tariffa allo stesso prima

di aver trovato uno specifico accordo con l'azienda relativamente alle royalty. Un anno dopo quell'intervento di Joy, il vicepresidente Marco Boerries apparve sullo stesso palco per illustrare il nuovo compromesso raggiunto per Open-Office, pacchetto applicativo per ufficio appositamente progettato per il sistema operativo GNU/Linux.

"Posso sintetizzare il tutto a tre lettere", disse Boerries. "GPL."

All'epoca, aggiunse, la decisone della Sun non dipese tanto da Stallman quanto dal successo dei programmi sotto la GPL. "In pratica si riconobbe che prodotti diversi attiravano comunità diverse, e la licenza d'uso dipende dal tipo di comunità a cui si punta", spiegò Boerries. "Con [OpenOffice], appariva evidente che l'affinità maggiore fosse con la comunità GPL."[1]

Tali osservazioni mettono in luce il forte impatto, spesso sottovalutato, della GPL e indirettamente la genialità politica della persona che ha rivestito un ruolo di primo piano nella sua creazione. "Non esiste sulla terra un avvocato che ne avrebbe stilato il testo così com'è", sostiene Eben Moglen, professore di legge presso la Columbia University e consulente legale della Free Software Foundation. "Ma funziona. E funziona grazie alla filosofia progettuale di Richard."

Ex-programmatore professionista, Moglen iniziò a collaborare gratuitamente con Stallman fin dal 1990, quando quest'ultimo ne richiese l'assistenza per una questione privata. Moglen, allora impegnato nelle battaglie legali a difesa dell'esperto di crittazione Philip Zimmerman[2] contro il governo federale, dice di essersi sentito onorato da quella richiesta. "Gli spiegai che usavo l'Emacs ogni giorno, e ci sarebbero volute molte consulenze legali prima di poterlo ripagare di quel debito."

Da allora, Moglen ha avuto forse più di qualunque altro la possibilità di osservare da vicino l'applicazione in ambito legale delle filosofie hacker di Stallman. Secondo Moglen, il suo approccio al codice legale e a quello del software è sostanzialmente identico. "Da avvocato, devo dire che non ha molto senso l'idea secondo cui anche in un documento legale tutto quello che bisogna fare è scoprirne i difetti per poi passare a correggerli", spiega Moglen. "Ogni procedimento legale riveste per sua natura caratteristiche di ambiguità e

agli avvocati spetta catturare i benefici di tale ambiguità a favore dei propri clienti. Invece Richard mira a una posizione completamente opposta. Vorrebbe eliminare quest'ambiguità, operazione intrinsecamente impossibile. Non è possibile stilare una licenza in grado di controllare tutte le circostanze di tutti i sistemi legali in ogni parte del mondo. Ma se è lì che si vuole arrivare, allora bisogna farlo a modo suo. E l'eleganza, la semplicità progettuale che ne risultano riescono quasi a raggiungere l'obiettivo prefissato. Da questo punto in poi, basta l'intervento di un avvocato per arrivare molto lontano."

Nei panni di chi ha l'incarico di portare avanti il progetto di Stallman, Moglen comprende la frustrazione dei potenziali alleati. "Richard non ammette compromessi su questioni che ritiene fondamentali", afferma, "né accetta facilmente le distorsioni terminologiche o anche soltanto quella artificiosa ambiguità che spesso la società impone a molti di noi".

Considerando l'indisponibilità della Free Software Foundation a farsi coinvolgere in questioni al di fuori dello sviluppo di GNU e dell'applicazione della GPL, Moglen ha dedicato le energie eccedenti alla Electronic Frontier Foundation, organizzazione che recentemente ha fornito assistenza legale a persone accusate di violazioni del copyright, come ad esempio Dmitri Skylarov. Nel 2000, Moglen ha inoltre operato come consulente di un gruppo di hacker incriminati per aver fatto circolare il programma deCSS, grazie al quale è possibile infrangere i sistemi di crittazione dei DVD. Nonostante il silenzio del suo maggior cliente in entrambi i casi, Moglen ha imparato ad apprezzare la testardaggine di Stallman. "Nel corso degli anni mi è capitato più volte di essermi recato da Richard per dirgli: 'Dobbiamo fare questo e quello. Ecco lo scenario strategico. Ecco la prossima mossa in cui dobbiamo impegnarci.' E la risposta di Richard è stata sempre la stessa: 'Non dobbiamo fare proprio nulla. Basta aspettare. Quel che deve essere fatto, si farà da sé'."

"E sai una cosa?" aggiunge Moglen. "Generalmente ha sempre avuto ragione."

Si tratta di osservazioni di segno opposto alla definizione che Stallman dà di se stesso: "Non mi ci vedo a giocare simili partite", risponde ai molti critici invisibili che lo considerano invece un consumato stratega. "Non sono in

grado di proiettarmi in avanti e di anticipare le mosse di qualcun altro. Il mio approccio è sempre stato quello di concentrarmi sui pilastri portanti, dicendo 'Vediamo di costruire delle fondamenta il più possibile solide'."

La crescente popolarità e la continua energia gravitazionale ottenute dalla GPL rappresentano i migliori tributi alle fondamenta create da Stallman e dagli altri colleghi del progetto GNU. Anche se non gli è più possibile autodefinirsi "ultimo vero hacker", Stallman può comunque vantarsi di aver costruito da solo il contesto etico alla base del movimento del software libero. Poco importa se gli attuali programmatori si sentano o meno a proprio agio nell'operare all'interno di tale contesto. Il fatto stesso che possano avere l'opportunità di affrontare una simile scelta costituisce il lascito più importante di Stallman.

Potrà apparire forse prematuro discutere oggi di quale sarà il suo ruolo storico. A 48 anni, Stallman ha sicuramente davanti a sé ancora qualche anno da aggiungere o sottrarre a un tale lascito. Eppure la natura da pilota automatico del movimento del software libero porta a tentare di esaminarne la vita al di fuori delle battaglia quotidiane dell'industria del software per ricondurla in un ambito più ampio e di portata storica.

A suo vantaggio va detto che Stallman rifiuta ogni opportunità di speculazione. "Non sono mai stato capace di prevedere in dettaglio che cosa avrebbe portato il futuro", insiste, tracciando il proprio prematuro epitaffio. "Ho detto soltanto, 'Sono deciso a lottare. Chissà dove andrò a finire?"'

Non c'è dubbio che nella scelta delle battaglie da portare avanti Stallman si sia alienato proprio coloro che altrimenti avrebbero potuto rivelarsi i maggiori alleati. Ma a testimonianza della sua inflessibile natura morale, va aggiunto che molti dei suoi più fieri avversari politici finiscono, alla fine, per dire bene di lui. Tuttavia, la tensione esistente tra Stallman l'ideologo e Stallman il genio hacker, porta il biografo a porsi una domanda: in che modo verrà considerato una volta messa fuori gioco la sua personalità?

Si tratta di quella che, nelle prime stesure del volume, ho definito "la domanda dei 100 anni". Nella speranza di stimolare punti di vista obiettivi su Stallman e il suo operato, ho chiesto a diversi luminari dell'industria del

software di estraniarsi dall'attuale periodo per porsi invece nella posizione di uno storico futuro che voglia analizzare il movimento del software libero tra un secolo. Dal punto di vista della storia contemporanea, balzano all'occhio le analogie tra Stallman e alcune importanti figure del passato americano, le quali, pur se in qualche misura marginali quando erano in vita, hanno successivamente raggiunto un'importanza notevole rispetto al proprio periodo storico. I primi raffronti comprendono Henry David Thoreau, filosofo trascendentalista nonché autore di On Civil Disobedience, e John Muir, fondatore del Sierra Club e progenitore del movimento ambientalista moderno. È inoltre facile notare somiglianze con individui tipo William Jennings Bryan, noto anche come "The Great Commoner" (Il grande comunitario), leader del movimento populista, nemico dei monopoli e persona che, pur se dotata di grande prestigio, oggi sembra scomparsa dagli annali della storia.

Pur non essendo certo il primo a ritenere il software una proprietà pubblica, è grazie alla GPL che Stallman si è garantito almeno una nota a piè di pagina nei testi di storia. Muovendo da questo dato di fatto, potrebbe aver senso fare un passo indietro per esaminare l'eredità di Richard Stallman al di là dell'epoca attuale. Nell'anno 2102 la GPL verrà ancora usata dai programmatori, oppure sarà caduta completamente in disuso? Il termine "software libero" apparirà politicamente bizzarro come oggi un'espressione tipo "argento libero", oppure verrà considerato stranamente premonitore alla luce degli eventi politici successivi?

Predire il futuro rimane pur sempre un'attività rischiosa, ma di fronte a una simile domanda varie persone si mostrano decise a dire la loro. "Fra cento anni Richard e un paio di altri individui meriteranno qualcosa in più di una semplice nota a piè di pagina", ritiene Moglen. "Verranno considerati figure importanti a livello storico."

In quel "paio di altri individui" che Moglen vuole come protagonisti di altrettanti capitoli nei futuri testi scolastici, troviamo John Gilmore, consigliere di Stallman sulla GPL e successivo co-fondatore della Electronic Frontier Foundation, e Theodor Holm Nelson, meglio noto come Ted Nelson,

autore nel 1982 del libro *Literary Machines*. Secondo Moglen, la rilevanza storica di figure quali Stallman, Nelson e Gilmore si pone in maniera significativa senza reciproche sovrapposizioni. A Nelson, a cui viene generalmente attribuita la paternità del termine "ipertesto", va riconosciuto il merito di aver evidenziato la difficile situazione della proprietà dell'informazione nell'era digitale. Gilmore e Stallman hanno invece il merito di aver individuato gli effetti politici negativi derivanti dal controllo dell'informazione e per aver contribuito alla nascita di organizzazioni – la Electronic Frontier Foundation nel caso di Gilmore e la Free Software Foundation per Stallman – il cui obiettivo rimane quello di controbilanciare tali effetti negativi. Tra le due, Moglen considera tuttavia l'attività di Stallman di natura più individuale che politica.

"L'unicità di Richard risiede nel fatto che le implicazioni etiche del software non-libero gli apparvero chiare fin dall'inizio", spiega Moglen. "Questo è strettamente legato alla sua personalità, la quale viene spesso dipinta come un epifenomeno o perfino come un serio ostacolo per la sua vita professionale."

Gilmore, che definisce il proprio inserimento tra l'erratico Nelson e l'irascibile Stallman come una sorta di "onore ambiguo", concorda comunque con la tesi di Moglen. Scrive infatti:

> Secondo me gli scritti di Stallman acquisteranno una rilevanza analoga a quelli di Thomas Jefferson; scrive in maniera assai chiara ed è altrettanto chiaro sui suoi principi… Difficile stabilire se la sua influenza risulterà pari a quella di Jefferson, poiché questo dipenderà dall'importanza che astrazioni chiamate "diritti civili" rivestiranno o meno da qui a cento anni, rispetto ad altre astrazioni che definiamo "software" oppure "restrizioni imposte a livello tecnico".

Un altro elemento dell'eredità di Stallman da non sottovalutare, prosegue il testo di Gilmore, rimane il modello di sviluppo collaborativo del software, di cui si è fatto pioniere il progetto GNU. Pur se talvolta difettoso, tale modello si è comunque evoluto in standard oggi comuni nell'industria del software. Ciò detto, conclude Gilmore, si tratta di un modello cooperativo che potrebbe avere maggiore influenza perfino del progetto GNU, della licenza GPL o di qualsiasi programma sviluppato da Stallman:

> Prima dell'arrivo di Internet, era alquanto difficile collaborare a distanza nella creazione di un software, anche all'interno di gruppi che si conoscevano e che si fidavano l'uno dell'altro. Richard è stato il primo ad avviare questi sforzi cooperativi, coinvolgendo soprattutto una serie di volontari disorganizzati, che finiscono per incontrarsi assai di rado. Richard non ha realizzato nessuno degli strumenti di base necessari a concretizzare tutto questo (il protocollo TCP, le mailing list, diff e patch, i file tar, RCS o CVS o remote-CVS), ma è riuscito tuttavia a impiegare al meglio le persone disponibili per dar vita a gruppi sociali di programmatori in grado di collaborare efficacemente tra loro.

Lawrence Lessig, professore di legge a Stanford e autore nel 2001 dell'opera The Future of Ideas, si mostra ugualmente ottimista. Come molti studiosi di diritto, egli considera la GPL un pilastro fondamentale della cosiddetta "collettività digitale", quel vasto agglomerato composto da programmi dalla proprietà condivisa e da standard di rete e di telecomunicazione cui si deve la crescita esponenziale di Internet negli ultimi trent'anni. Anziché inserire Stallman tra i pionieri di Internet, uomini come Vannevar Bush, Vinton Cerf e J.C.R. Licklider che convinsero la gente a considerare la tecnologia informatica in una prospettiva più ampia, Lessig considera l'impatto di Stallman importante soprattutto a livello personale, introspettivo e, in ultima analisi, come un elemento decisamente unico:

> [Stallman] ha spostato il dibattito da "è" a "dovrebbe". Ha messo in chiaro l'importanza della posta in gioco, realizzando un sistema in grado di portare avanti questi ideali… Ciò detto, non lo vedrei di fianco a Cerf o Licklider. L'innovazione è qualcosa di diverso. Nel suo caso, non si tratta soltanto di un determinato tipo di codice, o del lancio di Internet in generale. Riguarda maggiormente la capacità di mostrare alla gente il valore di un certo sviluppo di Internet. Non credo esista nessun altro in quest'ambito, prima o dopo di lui.

Naturalmente non tutti ritengono che l'eredità di Stallman sia scolpita nella pietra. Eric Raymond, il sostenitore dell'open source che pensa che la leadership di Stallman si sia ridotta in modo significativo dal 1996 in poi, intravede segnali ambigui nella sfera di cristallo del 2102:

> Credo che gli artefatti da lui realizzati (GPL, Emacs, GCC) verranno considerati opere rivoluzionarie, pietre miliari del mondo dell'informazione. Ritengo però che la storia si mostrerà meno clemente con alcune delle teorie da cui muove RMS, e per nulla favorevole alle tendenze personali a comportamenti territoriali, da leader di culto.

Per quanto concerne lo stesso Stallman, anch'egli suggerisce segnali contrastanti:

> Quel che rimarrà nella storia del progetto GNU, da qui a vent'anni, dipende da chi vincerà la battaglia sulla libertà di utilizzo della conoscenza pubblica. Se saremo noi a perdere, allora verremo relegati in una nota a piè di pagina. Se invece vinceremo, non è detto che la gente verrà a conoscenza del ruolo svolto dal sistema operativo GNU – se lo identificherà con "Linux" si creerà una falsa immagine di quanto è accaduto e del perché.
> Ma anche se dovessimo vincere, quale sarà la storia che la gente imparerà tra cento anni dipenderà probabilmente da chi si troverà a dominare la scena politica.

Tentando di definire il personaggio storico del XIX secolo a lui più somigliante, Stallman sceglie la figura di John Brown, il militante abolizionista considerato un eroe da una parte del Mason Dixon[3], e un folle dall'altra.

La rivolta degli schiavi capitanata da John Brown non riuscì mai a concretizzarsi, ma nel successivo processo egli riuscì a mettere efficacemente in luce le ragioni a sostegno dell'abolizione della schiavitù a livello nazionale. Durante la Guerra Civile, John Brown venne considerato un eroe; nel secolo successivo e per gran parte del 1900, i libri di storia lo dipingevano invece soltanto come un esagitato. Nell'epoca della segregazione legale, quando dominava l'intolleranza, gli Stati Uniti accettarono parzialmente la storia che si accordava con la versione degli stati meridionali, e di conseguenza i testi di storia contenevano parecchie bugie sulla Guerra Civile e i relativi eventi.

Simili raffronti rivelano sia che Stalmann stesso percepisce come marginale il suo attuale ambito operativo, sia la natura ambivalente della sua corrente reputazione. Nonostante appaia difficile immaginare che questa possa raggiungere gli infimi livelli toccati da quella di Brown negli anni della post-ricostruzione – Stallman, al di là di occasionali analogie di stampo bellico, ha fatto ben poco per ispirare violenza – è facile prevedere un futuro in cui le sue idee possano disperdersi come un mucchio di cenere al vento. Nel sostenere la causa del software libero, non in quanto movimento di massa bensì come un insieme di battaglie private contro le forze della tentazione proprietaria, Stallman sembra aver creato uno scenario che si preclude ogni vittoria, soprattutto per i molti accoliti accomunati da un'analoga testarda volontà.

Eppure, ancora una volta, forse un giorno sarà proprio tale volontà a rivelarsi il lascito più importante e duraturo di Stallman. Moglen, attento

osservatore della situazione negli ultimi dieci anni, mette sull'avviso quanti ritengono erroneamente la personalità di Stallman controproducente o secondaria rispetto agli stessi "artefatti" da lui realizzati finora. Secondo Moglen, senza tale personalità, non esisterebbero che pochi preziosi prodotti di cui parlare. Spiega Moglen, ex-funzionario presso la Corte Suprema:

> La persona più geniale per cui ho lavorato è stato Thurgood Marshall. Sapevo che cosa lo rendeva importante. Sapevo perché era stato capace di cambiare il mondo a modo suo. Forse il paragone è un po' forzato, perché i due non potrebbero essere più diversi tra loro. Thurgood Marshall era un uomo di mondo, il rappresentante degli emarginati all'interno della società in senso lato, ma pur sempre qualcuno ben inserito. Le sue capacità operavano a livello sociale. Ma anche lui era tutto d'un pezzo. Per quanto diversa sotto ogni altro punto di vista, la persona che gli si avvicina di più in tal senso, ossia tutto d'un pezzo, compatto, plasmato di quella materia che rende grandi, fino in fondo, è Stallman.

A chiarimento della propria tesi, Moglen riflette su un episodio avvenuto nella primavera del 2000. Il successo dell'entrata in borsa di VA Linux risuonava ancora nei grandi media, e un certo numero di tematiche relative al software libero circolava tra le notizie quotidiane. Accerchiati da un violento uragano di questioni e articoli che esigevano un rapido commento, Moglen ricorda di essere andato a pranzo con Stallman sentendosi come un naufrago improvvisamente catapultato nell'occhio del ciclone. Per un'ora, aggiunge, la conversazione ruotò tranquillamente su un unico argomento: come rafforzare la GPL.

"Ce ne stavamo seduti a discutere su cosa avremmo fatto per risolvere certi problemi sorti in Europa orientale e su come avremmo dovuto reagire quando la questione della proprietà dei contenuti avesse iniziato a minacciare il software libero", ricorda Moglen. "Mentre parlavamo, mi venne da pensare per un attimo a come potevamo apparire agli occhi di un passante. Eccoli qui, due piccoli anarchici barbuti, a complottare e pianificare le prossime mosse. E naturalmente Richard è intento a sciogliersi i nodi dai capelli, facendoli cadere nella minestra e comportandosi come al solito. Chiunque avesse avuto modo di orecchiare, non avrebbe potuto fare a meno di considerarci pazzi, ma io sapevo: sapevo che la rivoluzione stava proprio lì, a quel tavolo. Questo è lo spirito che la anima. E quest'uomo è la persona che la sta rendendo possibile."

Moglen dice che fu quel momento, più di ogni altro, a porre in evidenza l'elementare semplicità dello stile di Stallman.

"Fu divertente", rammenta. "Gli dissi, 'Sai, Richard, io e te siamo gli unici due a non aver intascato un soldo da questa rivoluzione'. E poi pagai per il pranzo, perché sapevo che lui non poteva permetterselo."

[1] Si veda Marco Boerries, intervista con l'autore (luglio 2000).

[2] Per ulteriori informazioni sui problemi legali di Zimmerman, si veda *Crypto* di Steven Levy, pp. 278-288. Nella versione originale di Free as in Freedom, ho raccontato che Moglen ha difeso Zimmerman contro la National Security Agency. Secondo il resoconto di Levy, Zimmerman era invece indagato dalla Procura e dalla dogana degli Stati Uniti, non dalla NSA.

[3] In origine indicava il confine tra Maryland e Pennsylvania, stabilito tra il 1763 e il 1767 dai due astronomi britannici Charles Mason e Jeremiah Dixon, a conclusione della causa legale tra le famiglie Calvert e Penn allora proprietarie dei due stati; per estensione, iniziò a indicare il confine tra gli "stati liberi" e quelli in cui vigeva la schiavitù nella prima metà del XVIII secolo, la linea di demarcazione tra Unionisti e Confederati durante la guerra civile. [*N.d.T.*]

Come sconfiggere la solitudine

Nel libro *The Autobiography of Malcolm X*, Alex Haley offre ai lettori un colpo d'occhio al dramma che si svolge dietro le quinte. Abbandonando il ruolo di ghostwriter, Haley decide di narrare l'epilogo finale in prima persona. Un epilogo in cui illustra il percorso di un giornalista freelance, inizialmente definito "strumento" e "spia" dal portavoce della Nation of Islam, che riesce poi a superare una serie di ostacoli personali e politici pur di mettere su carta il racconto della vita di Malcolm X.

Pur esitando a paragonare il mio libro a quello di Haley, ho un debito di gratitudine con lui per quello schietto epilogo. Nel corso degli ultimi 12 mesi mi è servito come una sorta di manuale d'uso per il trattamento di un soggetto biografico che ha costruito l'intera carriera sul continuo dissenso con chiunque altro. Fin dall'inizio avevo previsto di chiudere questo lavoro con un epilogo analogo, sia in omaggio a Haley sia per informare i lettori sull'evoluzione stessa del libro.

La storia dietro questa storia inizia in appartamento di Oakland, California, per dipanarsi successivamente nelle varie località menzionate nel libro – Silicon Valley, Maui, Boston e Cambridge. Ma in definitiva il racconto ruota intorno a due città: New York, capitale mondiale dell'editoria, e Sebastopol, California, capitale editoriale di Sonoma County.

Una storia che prende avvio nell'aprile 2000, epoca in cui collaboravo con il defunto sito Web BeOpen (*http://www.beopen.com/*). Uno dei miei primi lavori fu un'intervista telefonica con Richard M. Stallman. Il pezzo venne fuori abbastanza bene, al punto che Slashdot (*http://www.slashdot.org/*), il noto sito di "news for nerds" di proprietà della VA Software, Inc. (già VA Linux Systems e ancor prima VA Research), vi inserì un link nell'elenco dei

riferimenti quotidiani. Dopo poche ore, i server web di BeOpen presero a surriscaldarsi per i numerosi utenti che volevano leggere l'intervista.

Considerando gli scopi e gli obiettivi iniziali, la storia avrebbe dovuto chiudersi qui. Invece tre mesi dopo quell'intervista, mentre seguivo la O'Reilly Open Source Conference di Monterey, California, ricevetti la seguente e-mail da Tracy Pattison, responsabile dei diritti esteri presso un grande editore di New York:

> To: sam@BeOpen.com
> Subject: RMS Interview
> Date: Mon, 10 Jul 2000 15:56:37 -0400
> Ho letto con molto interesse la tua intervista con Richard Stallman su BeOpen. Seguo da tempo RMS e le sue attività, e direi che il tuo pezzo è veramente riuscito a catturare buona parte dello spirito di quanto Stallman sta facendo con GNU-Linux e la Free Software Foundation.
> >Mi piacerebbe tuttavia saperne di più – e non credo di essere la sola a volerlo. Credi sia possibile reperire ulteriori fonti e/o informazioni per espandere e aggiornare l'intervista e adattarla in una sorta di profilo di Stallman? Grazie forse a notizie più aneddotiche sulla sua personalità e sul suo passato che possano interessare e illuminare quei lettori al di fuori della ristretta cerchia dei programmatori?

Il messaggio suggeriva di telefonarle in modo da elaborare ulteriormente quell'idea. Fu esattamente quello che feci. Tracy mi riferì che il suo editore stava per lanciare una collana di libri elettronici, ed era alla ricerca di titoli in grado di attirare la prima ondata di utenti. Il formato dell'e-book non superava le 30.000 parole, circa 100 pagine, e lei stessa aveva suggerito di aprire con una delle figure più importanti della comunità hacker. L'idea era piaciuta e nella fase di ricerca si era imbattuta nell'intervista a Stallman su BeOpen. Aveva così deciso di inviarmi quella e-mail.

Ecco cosa mi chiedeva Tracy: sarei stato disposto ad ampliare l'intervista per arrivare ad un vero e proprio profilo di quella lunghezza?

La mia replica fu immediata: ci sto. Prima dell'accettazione definitiva, Tracy suggerì di mettere insieme una proposta formale da presentare ai suoi dirigenti. Due giorni dopo le inviai un progetto dettagliato. Trascorsa una settimana, Tracy replicò via e-mail di aver ottenuto semaforo verde.

Devo ammettere che ebbi un ripensamento sul fatto di poter coinvolgere Stallman in un progetto di libro elettronico. Da giornalista attento al mondo

open source, ero ben consapevole della sua enorme pignoleria. A quel punto avevo già raccolto una mezza dozzina di e-mail in cui venivo sgridato per aver usato “Linux” anziché “GNU/Linux”.

Eppure sapevo anche che Stallman era alla ricerca di nuove opportunità per far giungere il proprio messaggio a un pubblico più vasto. Forse se gli avessi presentato il progetto sotto questo punto di vista, si sarebbe mostrato maggiormente ricettivo. In caso negativo, avrei sempre potuto fare affidamento sulla notevole mole di documenti, interviste e resoconti online di conversazioni diffuse in giro da Stallman così da mettere insieme una biografia non autorizzata.

Durante la ricerca, scovai un saggio dal titolo “Freedom Or Copyright?” (Libertà o copyright?), scritto da Stallman e apparso sul numero di giugno 2000 della *MIT Technology Review*, dove si bacchettavano gli e-book per una serie di peccati commessi nel software. Non soltanto i lettori erano costretti a ricorrere a programmi proprietari per leggerli, lamentava Stallman, ma i metodi utilizzati a prevenzione di copie non autorizzate apparivano eccessivamente pesanti. Anzichè fare il download di un comune file HTML o PDF, gli utenti prelevavano un file cifrato. Essenzialmente l’acquisto di un e-book significava comprare una chiave non trasferibile tramite la quale poter decifrare il contenuto protetto da crittografia. Ogni tentativo di aprire tale contenuto in assenza dell’apposita chiave costituiva un reato penale sulla base del Digital Millennium Copyright Act, la legge del 1998 mirata a imporre la stretta tutela del copyright su Internet. Analogo lo scenario per quei lettori che avessero convertito il contenuto di un libro elettronico in un file di formato comune, pur se con l’unica intenzione di leggerlo altrove su un computer diverso. Al contrario di un normale volume cartaceo, ai lettori veniva negato il diritto a prestare, copiare o rivendere l’e-book. Veniva loro riconosciuto soltanto il diritto a leggerlo su una specifica macchina autorizzata, nella spiegazione di Stallman:

> Con i libri cartacei ci vengono tuttora riconosciute le libertà di una volta. Ma se questi dovessero essere sostituiti dagli e-book, ciò servirebbe a ben poco. Con il cosiddetto “inchiostro elettronico”, grazie al quale è possibile effettuare il download di nuovo testo in un foglio

> apparentemente stampato, perfino i giornali rischiano di divenire effimeri. Facile immaginarne le conseguenze: non più librerie dell'usato; impossibile prestare un libro a un amico o prenderne uno in prestito dalla biblioteca pubblica; nessuna possibilità che qualcuno possa leggere senza pagare. (E a giudicare dalle pubblicità di Microsoft Reader, basta perfino con l'acquisto anonimo di libri). Questo lo scenario che gli editori starebbero preparandoci[1].

Inutile aggiungere come il saggio suscitasse più di qualche preoccupazione. Né Tracy né io avevamo discusso il software a cui la casa editrice sarebbe ricorsa per l'e-book, e neppure si era parlato del tipo di copyright che ne avrebbe tutelato l'utilizzo. Le menzionai l'articolo su *Technology Review*, chiedendole di fornirmi ragguagli sulle procedure aziendali al riguardo. Tracy promise che mi avrebbe fatto sapere.

Impaziente di darmi da fare, decisi di chiamare comunque Stallman per informarlo del progetto. Quando lo feci, egli espresse un immediato interesse e un'altrettanto immediata preoccupazione. "Hai letto il mio saggio sugli e-book?" mi chiese. Gli risposi affermativamente, aggiungendo di essere in attesa della replica dell'editore. A quel punto egli pose due condizioni: non avrebbe appoggiato un meccanismo di licenza dell'e-book cui fondamentalmente si opponeva, e non voleva apparire come qualcuno che potesse offrire un simile sostegno. "Non voglio partecipare in alcunché possa farmi apparire un ipocrita", spiegò.

Per Stallman, la questione del software era secondaria rispetto a quella del copyright. Mi disse che non si sarebbe curato della scelta del software imposta dall'editore o dai rivenditori coinvolti, purché nella dicitura del copyright venisse specificatamente consentita agli utenti la libertà di fare e distribuire copie letterali del contenuto originale. Stallman segnalò come possibile modello *The Plant* di Stephen King. Nel giugno 2000 quest'ultimo annunciò sul proprio sito Web l'intenzione di auto-pubblicare il libro a puntate. Secondo quanto riportato nell'annuncio, il costo totale del libro finale sarebbe stato 13 dollari, diviso in rate da un dollaro ciascuna. Fintanto che il 75% dei lettori avesse pagato per ogni capitolo, King promise che avrebbe continuato a distribuire i successivi. Ad agosto il progetto parve funzionare, con due capitoli pubblicati e un terzo in arrivo.

“Potrei accettare qualcosa di analogo”, concluse Stallman. “Purché venga permessa la copia letterale”.

Inoltrai quanto sopra a Tracy. Fiducioso che saremmo riusciti a raggiungere un accordo equo per tutti, richiamai poi Stallman così da stabilire la prima intervista per il libro. Si disse d’accordo senza pormi ulteriori domande sullo stato delle cose. Poco tempo dopo la prima intervista, mi sbrigai a fissare la successiva (quella di Kihei), cercando di incastrarla prima che Stallman andasse in vacanza a Tahiti per due settimane.

Fu durante questa sua vacanza che ricevetti la brutta notizia da Tracy. Il dipartimento legale dell’azienda non voleva modificare alcun copyright sull’e-book. I lettori che avessero desiderato trasferirne i contenuti, sarebbero così stati costretti a superare il codice di cifratura oppure a convertirne il contenuto in un formato aperto tipo HTML. In un caso o nell’altro, ciò avrebbe significato infrangere la legge e andare incontro a conseguenze penali.

Con due interviste fresche già nel cassetto, non vedevo altro modo per scrivere il libro se non quello di raccogliere nuovo materiale. Organizzai rapidamente un viaggio a New York per incontrare il mio agente e Tracy così da studiare un’eventuale soluzione di compromesso.

Una volta a New York, vidi prima il mio agente, Henning Guttman. Era il nostro primo incontro faccia a faccia, e si dimostrò pessimista sulla possibilità di forzare qualche compromesso, almeno rispetto alla posizione dell’editore. Le case editrici grandi e famose guardavano già con sospetto al formato del libro elettronico e non parevano propense a sperimentare una formulazione del copyright che facilitasse agli utenti la possibilità di non pagare. In quanto agente specializzato in editoria tecnica, tuttavia, Henning si mostrò interessato all’aspetto romanzesco della situazione. Lo informai sulle due interviste già raccolte e sulla promessa di non pubblicare il libro in alcuna maniera che avesse potuto far apparire Stallman “come un’ipocrita”. Concordando sul fatto che fossi vincolato da un accordo morale, suggerì di far leva su un simile vincolo.

A parte ciò, aggiunse Henning, avremmo sempre potuto ricorrere alla strategia del bastone e della carota. Quest’ultima riguardava la pubblicità

derivante dall’uscita di un e-book rispettoso dei principi etici interni alla comunità hacker. Il bastone era costituito dai rischi insiti nella pubblicazione di un libro elettronico che invece andasse contro tali principi. Nove mesi prima che Dmitri Skylarov divenisse un caso celebre su Internet, eravamo consapevoli che sarebbe stata soltanto questione di tempo prima che qualche abile programmatore rivelasse come infrangere i sistemi di protezione degli e-book. Sapevamo altresì come la pubblicazione presso un grande editore di un libro elettronico protetto da crittografia su Richard M. Stallman equivalesse a mettere in copertina una frase tipo: “ruba questo e-book”.

Dopo l’incontro con Henning, decisi di richiamare Stallman. Nella speranza di rendere la carota più allettante, discutemmo alcuni possibili compromessi. Perché non proporre all’editore di pubblicare il libro sotto una doppia licenza, qualcosa di simile a quanto aveva fatto la Sun Microsystems con Open Office, il pacchetto desktop di software libero? La versione commerciale dell’e-book avrebbe potuto essere pubblicata in un formato normale, approfittando delle opzioni aggiuntive offerte dall’apposito software, e al contempo sarebbe stata diffusa la versione copiabile nel formato HTML, esteticamente meno appetibile.

Stallman replicò di non aver problemi con la doppia licenza, ma era contrario all’idea di rendere la versione liberamente copiabile inferiore rispetto a quella a pagamento. Inoltre, aggiunse, l’idea era fin troppo complicata. La doppia licenza funzionò nel caso di Open Office soltanto perché non era possibile esercitare alcun controllo sulla decisione presa. In questa circostanza, aggiunse, poteva invece influenzare il risultato finale. Poteva rifiutarsi di collaborare.

Avanzai qualche altra proposta senza ottenere grande effetto. L’unica concessione che riuscii a strappare a Stallman fu che il copyright dell’e-book potesse limitare ogni forma di condivisione alla “redistribuzione non-commerciale”.

Prima di chiudere la conversazione, mi suggerì di informare l’editore sulla mia promessa di garantire la libera circolazione dell’opera. Gli risposi che, pur non potendomi dichiarare totalmente d’accordo su questa posizione, non

vedevo come avrei mai potuto finire il libro senza la sua personale cooperazione. Apparentemente soddisfatto, Stallman chiuse la telefonata con il solito saluto: “Happy hacking”.

Il giorno seguente io e Henning c’incontrammo con Tracy, che c’informò della disponibilità da parte dell’editore a pubblicare porzioni dell’e-book liberamente copiabili in formato non cifrato, senza però superare le 500 parole. Henning le spiegò che ciò non mi avrebbe consentito di tener fede all’impegno morale preso con Stallman. Tracy ribatté facendo presente gli obblighi contrattuali assunti dall’azienda con distributori quali Amazon.com. Anche nel caso della libera disponibilità del testo soltanto per questa volta, esisteva il rischio che gli altri partner l’avrebbero considerato una rottura degli impegni contrattuali. Escludendo qualsiasi cambiamento di opinione da parte dell’editore o di Stallman, la decisione finale pesava esclusivamente sulle mie spalle. Avrei potuto usare le interviste già realizzate e ribaltare il mio precedente accordo con Stallman, oppure avrei potuto rifarmi all’etica giornalistica per ribadire l’accordo verbale sulla pubblicazione del libro.

Dopo la riunione, io e il mio agente andammo a sederci in un bar sulla Third Avenue. In quei giorni usavo il suo telefono cellulare per chiamare Stallman, lasciando un messaggio quando non rispondeva. Henning si assentò un momento, così da lasciarmi il tempo di raccogliere le idee. Quando tornò, teneva in mano il cellulare.

“È Stallman”, annunciò.

La conversazione partì subito col piede sbagliato. Gli riferii i commenti di Tracy sugli obblighi contrattuali dell’editore.

“E allora?”, fece Stallman bruscamente. “Perché mai dovrei fregarmene dei loro impegni contrattuali?”

“Perché è troppo chiedere a una grande casa editrice di rischiare una battaglia legale con i distributori per un libro elettronico di 30.000 parole”, azzardai.

“Ma non te ne rendi conto?”, replicò Stallman. “È proprio questo il punto. Io voglio un segnale di vittoria. Voglio costringerli a decidere tra la libertà e i soliti affari commerciali.”

Con la frase “un segnale di vittoria” che mi echeggiava nella testa, spostai temporaneamente l’attenzione verso i passanti fuori dal bar. Entrando nel locale, avevo notato con piacere che si trovava a due passi dall’angolo immortalato nel pezzo del 1976 dei Ramones “53rd and 3rd”, canzone che mi era sempre piaciuto rifare nel mio passato da musicista. Come il vagabondo in perenne frustrazione protagonista di quel pezzo, mi sentii crollare addosso in un attimo tutto quello che avevo costruito fino a quel momento. L’ironia era palpabile. Dopo aver trascorso settimane a registrare con piacere le lamentele altrui, eccomi nella posizione di dover tentare la più difficile delle manovre: strappare un compromesso a Richard Stallman.

Fu mentre continuavo a blaterare sulla posizione dell’editore, rivelando così la crescente simpatia nei suoi confronti, che Stallman, come un animale alla vista del sangue, partì all’attacco.

“Così stanno le cose, dunque? Vuoi abbandonarmi? Sei proprio deciso a piegarti ai loro voleri?”

Sollevai nuovamente la questione del doppio copyright.

“Intendi dire, doppie licenze”, tagliò corto Stallman.

“Si, licenze, copyright, quella roba lì”, risposi, sentendomi improvvisamente come un tonno preso all’amo che lascia una scia di sangue nel mare dietro di sè.

“Ah, ma perché diavolo non hai fatto come ti avevo suggerito?”, urlò.

Devo aver sostenuto la posizione dell’editore fino alla fine, perché nei miei appunti sono riuscito a prendere nota della bordata finale di Stallman: “Non m’importa nulla. Quel che stanno cercando di fare è sbagliato. Addio.”

Appena messo giù il telefono, ecco il mio agente farmi scivolare davanti un boccale di Guinness alla spina. “Ho pensato che ne avresti avuto bisogno”, fece ridendo. “Verso la fine della telefonata, mi sono accorto che stavi tremando.”

Proprio così. E avrei smesso di tremare soltanto dopo aver scolato più di mezzo boccale di birra. Era strano sentirsi definire un emissario del “male”. Ancor più perché appena tre mesi prima me ne stavo nel mio appartamento di Oakland cercando ispirazione per il prossimo articolo. Adesso invece ero

seduto in una parte del mondo conosciuta soltanto attraverso dei pezzi rock, tenendo riunioni con direttori editoriali e bevendo birra davanti a un agente letterario che non avevo mai visto in faccia fino al giorno prima. Pareva tutto così irreale, come guardare la mia vita scorrere all'indietro nel montaggio di un film.

Fu in quel momento che si fece avanti il mio senso dell'assurdo. Il tremore iniziale diede luogo a risate convulse. Agli occhi del mio agente devo essere parso uno dei tanti fragili scrittori colpito da un improvviso collasso emotivo. Mi sembrava invece di iniziare ad apprezzare la cinica bellezza di quella situazione. Che l'affare fosse andato in porto o meno, avevo già per le mani un buon articolo. Si trattava soltanto di trovare il posto giusto dove farlo uscire. Quando finalmente riuscii a fermare quelle convulsioni, alzai il bicchiere per proporre un brindisi.

"Benvenuto alla linea del fronte, amico mio", dissi brindando con il mio agente. "Meglio non prendersela troppo".

Se stessimo davvero seguendo la trama di uno spettacolo teatrale, a questo punto sarebbe stato appropriato un interludio romantico. Affranta dall'intensità del precedente incontro, Tracy ci aveva invitato ad andare a bere qualcosa insieme ai colleghi. Lasciammo perciò il bar sulla Third Avenue, scendendo giù verso l'East Village per unirci a lei e ai suoi amici.

Una volta lì, io e Tracy cominciammo a conversare avendo però cura di evitare ogni questione professionale. Una conversazione piacevole, rilassata. Prima di lasciarci, rimanemmo d'accordo che ci saremmo visti nuovamente l'indomani sera. Ancora una volta la conversazione fu piacevole, a tal punto che l'e-book di Stallman divenne quasi un lontano ricordo.

Tornato a Oakland, chiamai vari amici e conoscenti giornalisti. Raccontai loro le mie vicissitudini. Per lo più venni rimproverato per aver lasciato troppo spazio a Stallman nelle trattative precedenti le interviste. Un ex professore di scuola media mi suggerì di ignorare gli "ipocriti" commenti di Stallman e procedere tranquillamente con la stesura del libro. Alcuni reporter a conoscenza della sua sagacia nei confronti dei media, espressero solidarietà

ma offrirono una risposta unanime: la decisione finale pesa unicamente sulle tue spalle.

Decisi allora di mettere a riposo il progetto. Anche con quelle interviste, non è che stessi facendo grandi progressi. In tal modo potevo inoltre continuare a vedere Tracy senza dover prima discutere con Henning. Entro Natale ci eravamo scambiati reciprocamente la visita: lei era volata in California, io ero andato una seconda volta a New York. Il giorno prima di fine anno, le chiesi di sposarmi. Dovendo scegliere su quale costa vivere, optai per New York. A febbraio impacchettai il portatile e tutto il materiale relativo alla biografia di Stallman, e traslocai definitivamente sulla East Coast. Io e Tracy ci sposammo l'11 maggio. Decisamente un bel risultato per un contratto editoriale fallito.

Durante l'estate, iniziai a contemplare la possibilità di trasformare le interviste in articoli per qualche rivista. Eticamente mi sembrava una posizione corretta, poiché nell'accordo originale non si era stabilito nulla riguardo le comuni pubblicazioni cartacee. A essere onesti, mi sentivo maggiormente a mio agio provando a scrivere qualcosa su Stallman trascorsi otto mesi di silenzio radio. Dopo quella telefonata a settembre, mi aveva inviato soltanto due e-mail. In entrambe venivo redarguito per aver usato "Linux" anziché "GNU/Linux" in un paio di articoli per la rivista web *Upside Today*. A parte ciò, avevo apprezzato quel silenzio. A giugno, una settimana dopo il suo intervento alla New York University, decisi di rompere il ghiaccio buttando giù un articolo di 5.000 parole su Stallman. Stavolta le parole fluivano bene. La distanza aveva contribuito a farmi riguadagnare una prospettiva emotiva.

A luglio, un anno esatto dopo la prima e-mail di Tracy, ricevetti una telefonata da Henning. Mi diceva dell'interesse di O'Reilly & Associates, casa editrice di Sebastopol, California, alla pubblicazione della biografia di Stallman. La notizia non mancò di farmi piacere. Fra tutti gli editori al mondo, O'Reilly, lo stesso per cui era uscito *The Cathedral and the Bazaar* di Eric Raymond, sembrava il più sensibile alle questioni che avevano di fatto stroncato il precedente e-book. Come giornalista, avevo utilizzato

pesantemente i riferimenti storici contenuti nel volume *Open Sources*, anch'esso apparso per i tipi di O'Reilly[2]. Sapevo inoltre che parecchi capitoli di quest'ultimo, compreso quello curato da Stallman, erano accompagnati da un copyright che ne consentiva la redistribuzione. Informazione utile nel caso in cui il formato elettronico fosse saltato nuovamente fuori.

Come infatti accadde puntualmente. Tramite Henning venni a sapere dell'intenzione di O'Reilly di pubblicare la biografia sia come libro cartaceo sia all'interno del servizio a pagamento per abbonati denominato *Safari Tech Books Online*. La licenza stabilita in tal caso avrebbe previsto alcune limitazioni particolari[3], mi avvisò Henning, ma O'Reilly si dichiarò d'accordo alla stesura di un copyright che consentisse agli utenti di copiare e distribuire il testo del libro a prescindere dal formato. Praticamente, in quanto autore, mi veniva offerta la possibilità di scegliere tra due tipi di licenza: la Open Publication License oppure la GNU Free Documentation License.

Verificai contenuti e antefatti relativi a ciascuna possibilità. La Open Publication License (OPL)[4] riconosce ai lettori il diritto alla riproduzione e alla distribuzione dell'opera, per intero o in parte, tramite ogni supporto "fisico o elettronico", purché l'opera copiata mantenga l'identica Open Publication License. Vengono altresì consentite alcune modifiche sulla base di particolari condizioni. Sono infine incluse una serie di opzioni che, se così specificate dall'autore, possono limitare la creazione di versioni "modificate in maniera sostanziale" o di derivati sotto forma di libro privi di approvazione preventiva da parte dello stesso autore.

Invece la GNU Free Documentation License (GFDL)[5] consente la copia e distribuzione di un documento tramite ogni supporto, purché l'opera risultante segua la medesima licenza. Permette inoltre di apportare modifiche rispettando determinate condizioni. Al contrario della OPL, tuttavia, non offre agli autori l'opzione di limitare ulteriormente specifici cambiamenti. Neppure consente agli stessi autori di respingere quelle modifiche che possano risultare in un prodotto editoriale in diretta concorrenza. Richiede però l'inclusione di informazioni specifiche in copertina e sul retro-copertina nel caso in cui

qualcuno, al di fuori del detentore del copyright, volesse pubblicare oltre 100 copie di un'opera così tutelata.

Nel lavoro di ricerca sulle licenze, mi assicurai di studiare l'apposita pagina presente nel sito del progetto GNU, sotto il titolo "Various Licenses and Comments About Them" (Licenze varie e relativi commenti)[6]. Fu qui che trovai un'analisi della Open Publication License firmata da Stallman. La critica verteva sulla creazione di opere modificate e sulla possibilità riservata all'autore di scegliere una qualsiasi delle opzioni della OPL onde limitare le modifiche consentite. Nel caso in cui però l'autore non fosse intenzionato a selezionare alcuna opzione, faceva notare Stallman, meglio sarebbe stato ricorrere alla GFDL, la quale minimizzava il rischio della successiva apparizione di quelle opzioni non selezionate nelle versioni modificate di un documento.

In entrambe le licenze l'importanza delle modifiche ne rifletteva l'obiettivo originario – ovvero, consentire ai possessori di manuali software la possibilità di migliorarne il testo tenendone comunque informato il resto della comunità. Dato che in questo caso non si trattava di un manuale, le clausole sulle modifiche delle due licenze mi riguardavano ben poco. La mia unica preoccupazione stava nel garantire agli utenti la libertà di scambiare copie del libro o copiarne il contenuto, assicurando una libertà identica a quella acquisita grazie all'acquisto del volume cartaceo. Considerando valide entrambe le licenze in tal senso, decisi di firmare il contratto con O'Reilly.

Eppure ero attratto dall'idea di consentire modifiche senz'alcuna restrizione. Nella fase iniziale delle trattative con Tracy, avevo menzionato i meriti di una licenza tipo GPL per il contenuto dell'e-book. Nel caso peggiore, mi dicevo, quella licenza avrebbe garantito una buona pubblicità positiva. Nel caso migliore, avrebbe incoraggiato i lettori a partecipare al processo di scrittura del libro. Come autore avrei consentito agli altri l'emendamento del testo purché il mio nome fosse comparso sempre davanti a quelli altrui. Inoltre, sarebbe stato interessante osservare l'evoluzione del progetto. Immaginavo le edizioni successive simili a versioni online del Talmud, con il

mio testo originale in una colonna centrale circondato dai perspicaci commenti altrui ai margini.

Sostanzialmente l'idea traeva ispirazione dal progetto Xanadu (*http://www.xanadu.com/*), la leggendaria idea di software inizialmente concepita da Ted Nelson nel 1960. Durante la O'Reilly Open Source Conference del 1999, avevo seguito la prima dimostrazione del derivato open source di quel progetto, Udanax, rimanendo assai colpito dai risultati ottenuti. Nella visualizzazione di una sequenza, Udanax presentava l'opera originale e quella derivata tramite il semplice formato testuale affiancato su due colonne. Grazie a un semplice clic, il programma visualizzava delle linee che collegavano ogni frase del testo originario con l'analogo concettuale dello scritto derivato. Non sembrava certo necessario ricorrere a Udanax per una biografia in formato e-book di Richard M. Stallman, ma vista l'esistenza di simili capacità tecnologiche, perché non offrire agli utenti la possibilità di darsi da fare?[7]

Quando Laurie Petrycki, redattore incaricato del progetto presso O'Reilly, mi propose la scelta tra la OPL e la GFDL, venni nuovamente cullato da quella fantasia. A settembre 2001, in occasione della firma del contratto, i libri elettronici erano divenuti pressoché lettera morta. Numerose case editrici, inclusa quella in cui lavorava Tracy, stavano chiudendo le relative collane per mancanza d'interesse. Sorse spontanea la domanda: se gli editori avessero trattato gli e-book non come un formato editoriale bensì in quanto possibilità mirata alla costruzione di una comunità, forse il libro elettronico sarebbe riuscito a sopravvivere?

Dopo aver firmato il contratto, informai Stallman della ripresa del progetto. Nominai la scelta offertami da O'Reilly tra la Open Publication License e la GNU Free Documentation License, aggiungendo di propendere per la prima, anche soltanto perché non vedevo motivo per offrire ai concorrenti di O'Reilly la possibilità di stampare l'identico libro con una copertina diversa. Stallman replicò a favore della GFDL, notando come lo stesso O'Reilly l'avesse già usata parecchie volte in passato. Nonostante quanto accaduto l'anno precedente, proposi un patto. Avrei scelto la GFDL se in cambio Stallman si

fosse dichiarato disponibile a ulteriori interviste, oltre a garantire una certa pubblicità al libro di O'Reilly. Egli si dichiarò d'accordo sul primo punto, ma la sua partecipazione a eventi promozionali sarebbe dipesa dal contenuto finale del volume. Considerandola una posizione corretta, organizzai un'intervista per il 17 dicembre 2001 a Cambridge.

L'intervista coincideva con un viaggio d'affari di mia moglie Tracy a Boston. Due giorni prima di partire, Tracy mi suggerì d'invitare Stallman fuori a cena. "Dopo tutto", disse, "è stato lui a farci incontrare".

Gli mandai una e-mail, alla quale replicò prontamente accettando l'invito. Il giorno seguente guidai fino a Boston, incontrai Tracy in albergo e poi saltammo sul tram T per raggiungere il MIT. Arrivati nell'edificio di Tech Square, bussando alla porta di Stallman lo trovammo nel bel mezzo di una conversazione.

"Spero non ci siano problemi", disse mentre apriva la porta quel tanto che bastava perché io e Tracy notassimo la presenza della controparte. Si trattava di una ragazza, direi sui 25 anni, di nome Sarah.

"Mi sono preso la libertà di invitare qualcun altro a cena con noi", spiegò Stallman offrendomi lo stesso sorriso felino di quella volta al ristorante di Palo Alto.

A dire il vero, non ne rimasi troppo sorpreso. Qualche settimana prima avevo saputo della sua nuova amica, una notizia passatami da sua madre. "Anzi, il mese scorso sono andati insieme in Giappone, quando Richard vi si recò per ritirare il premio della Takeda", mi informò allora la signora Lippman[8].

Mentre andavamo al ristorante, appresi le circostanze del primo incontro tra Sarah e Richard. Fatto curioso, si trattava di uno scenario assai familiare. Sarah stava scrivendo un romanzo, e aveva sentito parlare di Stallman e di quanto fosse un tipo interessante. Decise subito di creare un personaggio che gli assomigliasse e, nello sviluppo del lavoro di ricerca, organizzò un'intervista dal vivo. Da qui le cose presero rapidamente un'altra piega. Era dall'inizio del 2001 che si frequentavano assiduamente, aggiunse Sarah.

“Ammiro molto il modo in cui Richard è riuscito a costruire un vero e proprio movimento politico muovendo da una questione profondamente individuale”, disse per spiegare la sua attrazione per Stallman.

Mia moglie rilanciò immediatamente la domanda: “Di quale questione si trattava?”

“Come sconfiggere la solitudine.”

Durante la cena lasciai gestire la conversazione alle donne, passando invece gran parte del tempo a seguire eventuali indizi per aiutarmi a stabilire se gli ultimi 12 mesi avessero ammorbidito Stallman in maniera significativa. Non notai nulla a sostegno di questa tesi. Anche se più civettuolo di quanto ricordassi – lo confermavano tra l’altro le molte volte che gli occhi Stallman sembrarono incantarsi sul seno di mia moglie – non mi apparve meno pungente di prima. Ad un certo punto, Tracy si lasciò andare ad un enfatico “Dio ce ne scampi!”, solo per ricevere un tipico rimprovero di Stallman.

“Mi spiace fartelo notare, ma Dio non esiste”, ribatté.

Più tardi, a cena conclusa e senza più Sarah, Stallman sembrò abbassare un po’ la guardia. Camminando verso una libreria nei pressi, ammise che gli ultimi 12 mesi avevano drasticamente cambiato le sue prospettive sul futuro. “Credevo di dover essere costretto a vivere da solo”, disse. “Sono contento di essermi sbagliato.”

Prima di lasciarci, Stallman mi allungò la sua “pleasure card”, un biglietto da visita completo di indirizzo, numero di telefono e passatempi preferiti (“condividere un bel libro, la buona cucina, la musica e le danze esotiche”), in modo da poter organizzare un’intervista conclusiva.

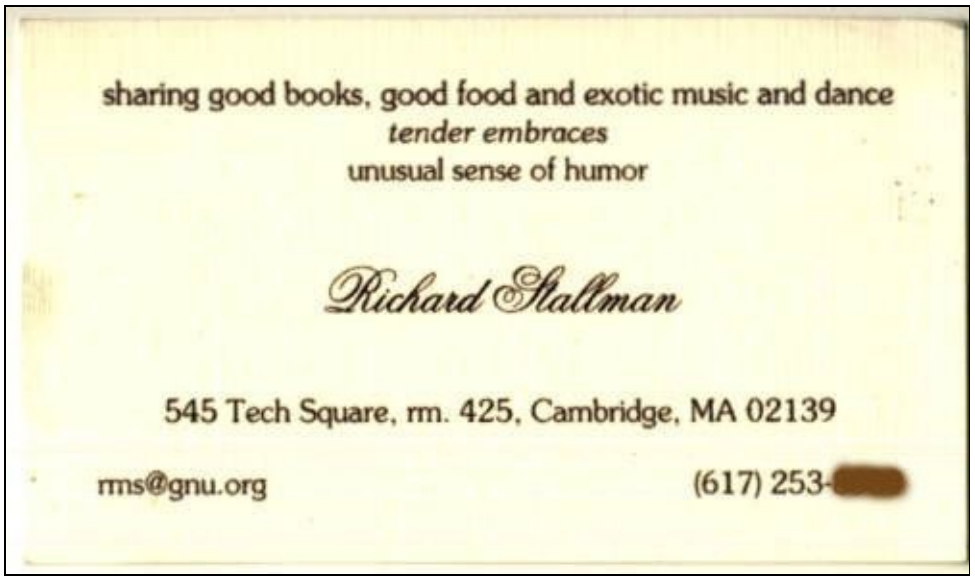


Figura 14.1 La “pleasure card” lasciatami da Stallman la sera della cena.

Il giorno dopo, davanti a un altro piatto di *dim sum*, Stallman parve ancora più ammiccante della sera precedente. Ricordando quelle discussioni nel dormitorio della Currier House sui pro e i contro di un ipotetico filtro dell’immortalità, espresse la speranza che un giorno qualche scienziato potesse scoprirne la formula. “Ora che finalmente sto iniziando ad assaporare la felicità in vita mia, vorrei averne ancora”, spiegò.

Quando gli menzionai la battuta di Sarah su “come sconfiggere la solitudine”, Stallman non riuscì a notare la connessione tra la solitudine a livello fisico o spirituale e quella di un hacker. “L’impulso a condividere il codice è basato sull’amicizia, ma in un ambito di gran lunga inferiore”, sostenne. Ma in seguito, quando la questione non tardò a ripresentarsi, dovette ammettere che la solitudine, o meglio, la paura della solitudine eterna, aveva giocato un ruolo di primo piano nella sua determinazione ad andare avanti nella prima fase del progetto GNU.

“Il fascino del computer non rappresentava nulla di esterno”, spiegò. “Non ne sarei rimasto meno ammaliato anche se fossi stato una persona famosa oppure se avessi avuto frotte di donne intorno a me. È però certamente vero che l’esperienza di non avere avuto una casa, trovarne una e perderla, poi trovarne un’altra e vedersela distruggere, tutto ciò mi colpì profondamente.

Quella perduta fu il dormitorio, quella distrutta il laboratorio di intelligenza artificiale. La precarietà di non poter fare affidamento su alcun tipo di casa o di comunità, provocò pesanti conseguenze. Mi spinse a combattere per riconquistare tutto ciò."

Dopo l'intervista, non potei fare a meno di provare una sensazione di simmetria emotiva. Ascoltando prima Sarah descrivere quel che l'aveva attratta a Stallman e poi quest'ultimo esprimere le emozioni che lo avevano spinto ad abbracciare la causa del software libero, mi vennero in mente le motivazioni alla base di questo libro. Fin dal luglio 2000 avevo imparato ad apprezzare sia il lato seducente sia quello sgradevole della sua persona. Come già Eben Moglen prima di me, ritengo che liquidare tali aspetti come marginali o fuorvianti rispetto al movimento del software libero nel suo complesso rappresenterebbe un grave errore. Per molti versi i due lati dipendono l'uno dall'altro, tanto da risultare pressoché indistinguibili.

Pur nella consapevolezza che non tutti i lettori proveranno la stessa affinità con Stallman – può darsi anzi che per qualcuno, dopo aver letto il libro, tale affinità si riduca a zero – sono sicuro che la maggioranza sia d'accordo con me. Esistono poche persone al mondo capaci di offrire un ritratto umano come quello di Richard M. Stallman. È mia sincera speranza che, portata a termine la mia descrizione e grazie all'aiuto della GFDL, qualcun altro sentirà un desiderio analogo di aggiungere il proprio punto di vista a questo ritratto.

[1] Si veda "Safari Tech Books Online; Subscriber Agreement: Terms of Service." http://safari.oreilly.com/mainhlp.asp?help=service

[2] Tr. it. *Open Sources. Voci dalla rivoluzione open source*, Apogeo, 1999

[3] http://safari.oreilly.com/mainhlp.asp?help=service

[4] Si veda "The Open Publication License: Draft v1.0" (8 giugno 1999). http://opencontent.org/openpub/

[5] Si veda "The GNU Free Documentation License: Version 1.2" (novembre 2002). http://www.gnu.org/copyleft/fdl.html

[6] Si veda http://www.gnu.org/philosophy/license-list.html

[7] Chiunque fosse intenzionato a modificare questo testo in modo da funzionare con Udanax, la versione di software libero di Xanadu, riceverà il mio entusiastico sostegno. Per saperne di più su questa intrigante tecnologia, si veda: http://www.udanax.com/.

[8] Fino al ritorno di Stallman dal viaggio in Giappone per la cerimonia di consegna, non seppi nulla della decisione della Takeda Foundation di premiare Stallman, insieme a Linus Torvalds e Ken Sakamura, con la prima edizione del “Techno-Entrepreneurial Achievement for Social/Economic Well-Being” (premio tecnico-imprenditoriale per l’avanzamento socioeconomico). Per ulteriori informazioni sull’iniziativa e il relativo premio da un milione di dollari, si veda il sito della Takeda: http://www.takeda-foundation.jp/.

Terminologia

Nella maggior parte dei casi, ho deciso di usare il termine GNU/Linux in riferimento al sistema operativo di software libero e Linux per indicare in modo particolare il kernel su cui si basa tale sistema operativo. L'eccezione più evidente compare nel Capitolo 9, quando descrivo l'evoluzione iniziale di Linux come un derivato di Minix. È corretto sostenere che durante i primi due anni di sviluppo del progetto, il sistema operativo su cui Torvalds e colleghi stavano lavorando presentava poca somiglianza con il sistema GNU impostato da Stallman, anche se gradualmente iniziò a condividerne alcuni componenti chiave come il compilatore C GNU e il debugger GNU.

La mia scelta trova ulteriore riscontro nel fatto che, prima del 1993, Stallman non insiste per il riconoscimento dei propri meriti.

Qualcuno potrebbe considerare arbitraria la decisione di usare GNU/Linux per le successive versioni del medesimo sistema operativo. Vorrei sottolineare come ciò non vada in alcun modo considerato un prerequisito per ottenere la cooperazione di Stallman nella stesura del libro. L'ho deciso di mia spontanea volontà, in parte per la natura modulare del sistema operativo e della relativa comunità, e in parte per la natura apolitica del termine Linux. Trattandosi qui della biografia di Richard Stallman, non mi è parso opportuno definire il sistema operativo in termini apolitici.

Nella fase finale del progetto, quando divenne chiaro che O'Reilly & Associates avrebbe pubblicato l'edizione originale del libro, Stallman pose come condizione l'uso di "GNU/Linux" anziché Linux nel caso O'Reilly avesse richiesto il suo sostegno promozionale dopo l'uscita del libro. Una volta venuto a conoscenza di questa richiesta, decisi di ribadire le mie scelte precedenti lasciando giudicare a Stallman se il testo finale rispettasse o meno

la sua condizione. Mentre scrivo queste note, non ho alcuna idea sull'esito del suo giudizio finale.

Analoga la situazione per quanto concerne i termini "software libero" e "open source". Di nuovo, ho optato per il primo, maggiormente motivato a livello politico, per indicare quei programmi il cui codice sia liberamente copiabile e modificabile. Anche se più diffuso, ho scelto invece il secondo soltanto in riferimento a gruppi e aziende che ne hanno esplicitamente sostenuto l'utilizzo. In alcuni passaggi, tuttavia, i termini sono completamente intercambiabili, e in tal caso ho preso questa decisione seguendo il consiglio di Christine Peterson, la persona generalmente indicata per avere coniato quel termine. "Nelle circostanze in cui suona meglio, si dovrebbe continuare a usare 'software libero"', scrive la Peterson. "['Open source'] si è imposto soprattutto perché esisteva un gran bisogno di un nuova terminologia, non perché fosse l'ideale."

Hack, hacker e hacking

Per comprendere appieno il significato del termine “hacker”, può essere d’aiuto esaminarne lo sviluppo etimologico seguito nel corso degli anni.

Il *New Hacker Dictionary*, compendio online dove sono raccolti i termini gergali dei programmatori, elenca ufficialmente nove diverse connotazioni per la parola “hack” e un numero analogo per “hacker”. Eppure la stessa pubblicazione include un saggio d’accompagnamento in cui si cita Phil Agre, un hacker del MIT che mette in guardia i lettori a non farsi fuorviare dall’apparente flessibilità del termine. “Hack ha solo un significato”, sostiene Agre. “Quello estremamente sottile e profondo di qualcosa che rifiuta ulteriori spiegazioni.”

A prescindere dall’ampiezza della definizione, la maggioranza degli odierni hacker ne fa risalire l’etimologia al MIT, dove il termine fece la sua comparsa nel gergo studentesco all’inizio degli anni ’50. Secondo una pubblicazione diffusa nel 1990 dal *MIT Museum* a documentare il fenomeno dell’hacking, per quanti frequentavano l’istituto in quegli anni il termine “hack” veniva usato con un significato analogo a quello dell’odierno “goof” (scemenza, goliardata). Stendere una vecchia carcassa fuori dalla finestra del dormitorio veniva considerato un “hack”, ma altre azioni più pesanti o dolose – ad esempio, tirare delle uova contro le finestre del dormitorio rivale, oppure deturpare una statua nel campus – superavano quei limiti. Era implicito nella definizione di “hack” lo spirito di un divertimento creativo e innocuo.

È a tale spirito che s’ispirava il gerundio del termine: “hacking”. Uno studente degli anni ’50 che trascorreva gran parte del pomeriggio chiacchierando al telefono o smontando una radio, poteva descrivere quelle attività come “hacking”. Di nuovo, l’equivalente moderno per indicare le

stesse attività potrebbe essere la forma verbale derivata da “goof” – “goofing” o “goofing off” (prendere in giro qualcuno, divertirsi).

Più avanti negli anni ’50, il termine “hack” acquistò una connotazione più netta e ribelle. Al MIT degli anni ’50 vigeva un elevato livello di competizione e l’attività di hacking emerse sia come reazione sia come estensione di una tale cultura competitiva. Goliardate e burle varie divennero tutt’a un tratto un modo per scaricare la tensione accumulata, per prendere in giro l’amministrazione del campus, per dare spazio a quei pensieri e comportamenti creativi repressi dal rigoroso percorso di studio dell’istituto. Va poi aggiunto che quest’ultimo, con la miriade di corridoi e tunnel sotterranei, offriva ampie opportunità esplorative per quegli studenti che non si facevano intimorire da porte chiuse e da cartelli tipo “Vietato l’ingresso”. Fu così che “tunnel hacking” divenne l’accezione usata dagli stessi studenti per indicare queste incursioni sotterranee non autorizzate. In superficie il sistema telefonico del campus offriva analoghe opportunità. Grazie ad esperimenti casuali ma accurati, gli studenti impararono a fare scherzi divertenti. Traendo ispirazione dal più tradizionale “tunnel hacking”, questa nuova attività venne presto battezzata “phone hacking”.

La combinazione tra divertimento creativo ed esplorazioni senza limiti costituirà la base per le future mutazioni del termine hacking. I primi ad auto-qualificarsi “computer hacker” nel campus del MIT negli anni ’60 traevano origine da un gruppo di studenti appassionati di modellismo ferroviario, che negli ultimi anni ’50 si erano riuniti nel Tech Model Railroad Club. Una ristretta enclave all’interno di quest’ultimo era il comitato Signals and Power (segnali ed elettricità) – gli addetti alla gestione del sistema del circuito elettrico dei trenini del club. Un sistema costituito da un sofisticato assortimento di relé e interruttori analogo a quello che regolava il sistema telefonico del campus. Per gestirlo era sufficiente che un membro del gruppo inviasse semplicemente i vari comandi tramite un telefono collegato al sistema, osservando poi il comportamento dei trenini.

I nuovi ingegneri elettrici responsabili per la costruzione e il mantenimento di tale sistema considerarono lo spirito di simili attività analogo a quello del

phone hacking. Adottando il termine hacking, iniziarono così a raffinarne ulteriormente la portata. Dal punto di vista del comitato Signals and Power, usare un relé in meno in un determinato tratto di binari significava poterlo utilizzare per qualche progetto futuro. In maniera sottile, il termine hacking si trasformò da sinonimo di gioco ozioso, a un gioco in grado di migliorare le prestazioni o l'efficienza complessiva del sistema ferroviario del club. Quanto prima i membri di quel comitato cominciarono a indicare con orgoglio l'attività di ricostruzione e miglioramento del circuito per il funzionamento delle rotaie con il termine "hacking", mentre "hacker" erano quanti si dedicavano a tali attività.

Considerata la loro affinità per i sistemi elettronici sofisticati – per non parlare della tradizionale avversione degli studenti del MIT verso porte chiuse e divieti d'ingresso – non ci volle molto prima che gli hacker mettessero le mani su una macchina appena arrivata al campus. Noto come TX-0, si trattava di uno dei primi modelli di computer lanciati sul mercato. Sul finire degli anni '50, l'intero comitato Signals and Power era emigrato in massa nella sala di controllo del TX-0, portandosi dietro lo stesso spirito di gioco creativo. Il vasto reame della programmazione informatica avrebbe portato a un'ulteriore mutamento etimologico. "To hack" non indicava più l'attività di saldare circuiti dalle strane sembianze, bensì quella di comporre insieme vari programmi, con poco rispetto per quei metodi o procedure usati nella scrittura del software "ufficiale". Significava inoltre migliorare l'efficienza e la velocità del software già esistente che tendeva a ingolfare le risorse della macchina. Rimanendo fedele alla sua radice, il termine indicava anche la realizzazione di programmi aventi l'unico scopo di divertire o di intrattenere l'utente.

Un classico esempio di quest'ampliamento della definizione di hacker è Spacewar, il primo video game interattivo. Sviluppato nei primi anni '60 dagli hacker del MIT, Spacewar includeva tutte le caratteristiche dell'hacking tradizionale: era divertente e casuale, non serviva ad altro che a fornire una distrazione serale alle decine di hacker che si divertivano a giocarvi. Dal punto di vista del software, però, rappresentava una testimonianza incredibile

delle innovazioni rese possibili dalle capacità di programmazione. Inoltre era completamente libero (e gratuito). Avendolo realizzato per puro divertimento, gli hacker non vedevano alcun motivo di mettere sotto scorta la loro creazione, che finì per essere ampiamente condivisa con altri programmatori. Verso la fine degli anni ’60, Spacewar divenne così il passatempo preferito di quanti lavoravano ai mainframe in ogni parte del mondo.

Furono i concetti di innovazione collettiva e proprietà condivisa del software a distanziare l’attività di computer hacking degli anni ’60 da quelle di tunnel hacking e phone hacking del decennio precedente. Queste ultime tendevano a rivelarsi attività condotte da soli o in piccoli gruppi, per lo più limitate all’ambito del campus, e la natura segreta di tali attività non favoriva l’aperta circolazione di nuove scoperte. Invece i computer hacker operavano all’interno di una disciplina scientifica basata sulla collaborazione e sull’aperto riconoscimento dell’innovazione. Non sempre hacker e ricercatori “ufficiali” andavano a braccetto, ma nella rapida evoluzione di quell’ambito le due specie di programmatori finirono per impostare un rapporto basato sulla collaborazione – si potrebbe perfino definire una relazione simbiotica.

Il fatto che la successiva generazione di programmatori, incluso Richard M. Stallman, aspirasse a seguire le orme dei primi hacker, non fa altro che testimoniare le prodigiose capacità di questi ultimi. Nella seconda metà degli anni ’70 il termine “hacker” aveva assunto la connotazione di élite. In senso generale, computer hacker era chiunque scrivesse il codice software per il solo gusto di riuscirci. In senso specifico, indicava abilità nella programmazione. Al pari del termine “artista”, il significato conteneva delle connotazioni tribali. Definire hacker un collega programmatore costituiva un segno di rispetto. Auto-descriversi come hacker rivelava un’enorme fiducia personale. In entrambi i casi, la genericità iniziale dell’appellativo computer hacker andava diminuendo di pari passo alla maggiore diffusione del computer.

Con il restringimento della definizione, l’attività di “computer” hacking acquistò nuove connotazioni semantiche. Per potersi definire hacker, una persona doveva compiere qualcosa di più che scrivere programmi interessanti; doveva far parte dell’omonima “cultura” e onorarne le tradizioni allo stesso

modo in cui un contadino del Medio Evo giurava fedeltà alla corporazione dei vinai. Pur se con una struttura sociale non così rigida come in quest'ultimo esempio, gli hacker di istituzioni elitarie come il MIT, Stanford e Carnegie Mellon iniziarono a parlare apertamente di "etica hacker": le norme non ancora scritte che governavano il comportamento quotidiano dell'hacker. Nel libro del 1984 *Hackers*, l'autore Steven Levy, dopo un lungo lavoro di ricerca e consultazione, codificò tale etica in cinque principi fondamentali.

Sotto molti punti di vista, i principi elencati da Levy continuano a definire l'odierna cultura del computer hacking. Eppure l'immagine di una comunità hacker analoga a una corporazione medievale, è stata scalzata dalle tendenze eccessivamente populiste dell'industria del software. A partire dai primi anni '80 i computer presero a spuntare un po' ovunque, e i programmatori, che una volta dovevano recarsi presso grandi istituzioni o aziende soltanto per aver accesso alla macchina, improvvisamente si trovarono a stretto contatto con hacker di grande livello via ARPAnet. Grazie a questa vicinanza, i comuni programmatori presero ad appropriarsi delle filosofie anarchiche tipiche della cultura hacker di ambiti come quello del MIT. Tuttavia, nel corso di un simile trasferimento di valori andò perduto il tabù culturale originato al MIT contro ogni comportamento malevolo, doloso. Mentre i programmatori più giovani iniziavano a sperimentare le proprie capacità con finalità dannose – creando e disseminando virus, facendo irruzione nei sistemi informatici militari, provocando deliberatamente il blocco di macchine quali lo stesso Oz del MIT, popolare nodo di collegamento con ARPAnet – il termine "hacker" assunse connotati punk, nichilisti. Quando polizia e imprenditori iniziarono a far risalire quei crimini a un pugno di programmatori rinnegati che citavano a propria difesa frasi di comodo tratte dall'etica hacker, quest'ultimo termine prese ad apparire su quotidiani e riviste in articoli di taglio negativo. Nonostante libri come *Hackers* avessero fatto parecchio per documentare lo spirito originale di esplorazione da cui nacque la cultura dell'hacking, per la maggioranza dei giornalisti "computer hacker" divenne sinonimo di "rapinatore elettronico".

Anche di fronte alla presenza, durante gli ultimi due decenni, delle forti lamentele degli stessi hacker contro questi presunti abusi, le valenze ribelli del termine risalenti agli anni ’50 rendono difficile distinguere tra un quindicenne che scrive programmi capaci di infrangere le attuali protezioni cifrate, dallo studente degli anni ’60 che rompe i lucchetti e sfonda le porte per avere accesso a un terminale chiuso in qualche ufficio. D’altra parte, la sovversione creativa dell’autorità per qualcuno non è altro che un problema di sicurezza per qualcun altro. In ogni caso, l’essenziale tabù contro comportamenti dolosi o deliberatamente dannosi trova conferma a tal punto da spingere la maggioranza degli hacker ad utilizzare il termine “cracker” – qualcuno che volontariamente decide di infrangere un sistema di sicurezza informatico per rubare o manomettere dei dati – per indicare quegli hacker che abusano delle proprie capacità.

Questo fondamentale tabù contro gli atti dolosi rimane il primario collegamento culturale esistente tra l’idea di hacking del primo scorcio del XXI secolo e quello degli anni ’50. È importante notare come, mentre la definizione di computer hacking abbia subìto un’evoluzione durante gli ultimi quattro decenni, il concetto originario di hacking in generale – ad esempio, burlarsi di qualcuno oppure esplorare tunnel sotterranei – sia invece rimasto inalterato. Nell’autunno 2000 il *MIT Museum* onorò quest’antica tradizione dedicando al tema un’apposita mostra, la *Hall of Hacks*. Questa comprendeva alcune fotografie risalenti agli anni ’20, inclusa una in cui appare una finta auto della polizia. Nel 1993, gli studenti resero un tributo all’idea originale di hacking del MIT posizionando la stessa macchina della polizia, con le luci lampeggianti, sulla sommità del principale edificio dell’istituto. La targa della macchina era IHTFP, acronimo dai diversi significati e molto diffuso al MIT. La versione maggiormente degna di nota, anch’essa risalente al periodo di alta competitività nella vita studentesca degli anni ’50, è “I hate this fucking place” (Odio questo posto fottuto). Tuttavia nel 1990, il *Museum* riprese il medesimo acronimo come punto di partenza per una pubblicazione sulla storia dell’hacking. Sotto il titolo Institute for Hacks Tomfoolery and Pranks (Istituto

per scherzi folli e goliardate), la rivista offre un adeguato riassunto di quelle attività.

“Nella cultura dell’hacking, ogni creazione semplice ed elegante riceve un’alta valutazione come si trattasse di scienza pura”, scrive Randolph Ryan, giornalista del Boston Globe, in un articolo del 1993 incluso nella mostra in cui compariva la macchina della polizia. “L’azione di hack differisce da una comune goliardata perché richiede attenta pianificazione, organizzazione e finezza, oltre a fondarsi su una buona dose di arguzia e inventiva. La norma non scritta vuole che ogni hack sia divertente, non distruttivo e non rechi danno. Anzi, talvolta gli stessi hacker aiutano nell’opera di smantellamento dei propri manufatti”.

Il desiderio di confinare la cultura del computer hacking all’interno degli stessi confini etici appare opera meritevole ma impossibile. Nonostante la gran parte dell’hacking informatico aspiri al medesimo spirito di eleganza e semplicità, il medium stesso del software offre un livello inferiore di reversibilità. Smontare una macchina della polizia è opera semplice in confronto allo smantellamento di un’idea, soprattutto quanto è ormai giunta l’ora per l’affermazione di tale idea. Da qui la crescente distinzione tra “black hat” e “white hat” (“cappello nero” e “cappello bianco”) – hacker che rivolgono nuove idee verso finalità distruttive, dolose contro hacker che invece mirano a scopi positivi o, quantomeno, informativi.

Una volta oscuro elemento del gergo studentesco, la parola “hacker” è divenuta una palla da biliardo linguistica, soggetta a spinte politiche e sfumature etiche. Forse è questo il motivo per cui a così tanti hacker e giornalisti piace farne uso. Nessuno può tuttavia indovinare quale sarà la prossima sponda che la palla si troverà a colpire.

Licenza per Documentazione Libera GNU

NOTA *Questa è semplicemente una traduzione della licenza che accompagna il libro e non ha valore legale. Per ogni questione di rilievo fare riferimento alla versione originale in lingua inglese, riportata di seguito.*

VERSIONE 1.2, NOVEMBRE 2002
Copyright © 2000, 2001, 2002 Free Software Foundation, Inc. 9 Temple Place, Suite 330, Boston, MA 02111-1307 USA Chiunque può copiare e distribuire copie letterali di questo documento di licenza, ma non ne è permessa la modifica.

0. PREAMBOLO

Lo scopo di questa licenza è di rendere un manuale, un testo o altri documenti utili e funzionali, “liberi” nel senso di assicurare a tutti la libertà effettiva di copiarli e redistribuirli, con o senza modifiche, a fini di lucro o meno. In secondo luogo questa licenza prevede per autori ed editori il modo per ottenere il giusto riconoscimento del proprio lavoro, preservandoli dall’essere considerati responsabili per modifiche apportate da altri.

Questa licenza è un “copyleft”: ciò vuol dire che i lavori che derivano dal documento originale devono essere ugualmente liberi. È il complemento alla Licenza Pubblica Generale GNU, che è una licenza di tipo “copyleft” pensata per il software libero.

Abbiamo progettato questa licenza al fine di applicarla alla documentazione del software libero, perché il software libero ha bisogno di documentazione libera: un programma libero dovrebbe accompagnarsi a manuali che forniscano la stessa libertà del software. Ma questa licenza non è limitata alla documentazione del software; può essere utilizzata per ogni testo che tratti un

qualsiasi argomento e al di là dell’avvenuta pubblicazione cartacea. Raccomandiamo principalmente questa licenza per opere che abbiano fini didattici o per manuali di consultazione.

1. APPLICABILITÀ E DEFINIZIONI

Questa licenza si applica a qualsiasi manuale o altra opera, su qualsiasi supporto, che contenga una nota del detentore del diritto d’autore indicante che si può distribuire nei termini di questa licenza. Tale nota garantisce una licenza di utilizzo dell’opera, secondo le condizioni indicate, libera da diritti, in tutto il mondo, per una durata illimitata. Con “documento”, in seguito ci si riferisce a qualsiasi manuale o opera. Ogni fruitore è un destinatario della licenza e viene indicato con “voi”. Se si copia, modifica o distribuisce l’opera secondo modalità tutelate dalla legge sul diritto d’autore, si accettano le condizioni di questa licenza.

Con “versione modificata” del documento si intende ogni opera contenente il documento stesso o parte di esso, sia riprodotto alla lettera sia con modifiche e/o traduzioni in un’altra lingua.

Una “sezione secondaria” è un’appendice cui si fa riferimento o una premessa del documento e riguarda esclusivamente il rapporto dell’editore o dell’autore del documento con l’argomento generale del documento stesso (o argomenti affini) e non contiene nulla che possa essere compreso nell’argomento principale. (Per esempio, se il documento è in parte un manuale di matematica, una sezione secondaria non può contenere spiegazioni di matematica). Il rapporto con l’argomento può essere un tema collegato storicamente con il soggetto principale o con soggetti affini, o essere costituito da argomentazioni legali, commerciali, filosofiche, etiche o politiche pertinenti.

Le “sezioni non modificabili” sono particolari sezioni secondarie i cui titoli sono esplicitamente dichiarati essere sezioni non modificabili, nella nota che indica che il documento è rilasciato sotto questa licenza. Se una sezione non rientra nella precedente definizione di sezione secondaria, allora non può

essere identificata come non modificabile. Il documento può non contenere sezioni non modificabili. Se non vi sono sezioni identificate come non modificabili, allora il documento non ne contiene.

I “testi di copertina” sono brevi brani di testo che sono elencati nella nota che indica che il documento è rilasciato sotto questa licenza. Un testo di copertina può contenere al massimo 5 parole, mentre un testo di quarta di copertina ne può contenere al massimo 25.

Una copia “trasparente” del documento indica una copia leggibile da un calcolatore, codificata in un formato le cui specifiche sono disponibili pubblicamente, i cui contenuti possono essere visti e modificati direttamente, ora e in futuro, con generici editor di testi o (per immagini composte da pixel) con generici editor di immagini o (per i disegni) con qualche editor di disegni ampiamente diffuso, e la copia deve essere adatta al trattamento per la formattazione o per la conversione in una varietà di formati atti alla successiva formattazione. Una copia fatta in un altro formato di file trasparente il cui markup, o l’assenza di markup, è stato organizzato per intralciare o scoraggiare modifiche future da parte dei lettori non è trasparente. Un formato immagine è non trasparente quando utilizzato per una quantità significativa di testo. Una copia che non è trasparente è “opaca”.

Esempi di formati adatti per copie trasparenti sono l’ASCII puro senza markup, il formato di input per Texinfo, il formato di input per LaTex, SGML o XML accoppiati a una DTD pubblica e disponibile, e semplice HTML, PostScript o PDF conforme agli standard e progettato per essere modificato manualmente. Esempi di formati immagine trasparenti sono PNG, XCF e JPG. Formati opachi sono formati proprietari che possono essere letti e modificati solo con word processor proprietari, SGML o XML per cui non è in genere disponibile la DTD o gli strumenti per il trattamento, e HTML, PostScript o PDF generati automaticamente da qualche word processor per il solo output.

La “pagina del titolo” di un libro stampato indica la pagina del titolo stessa, più qualche pagina seguente per quanto necessario a contenere in modo leggibile il materiale che la licenza prevede che compaia nella pagina del

titolo. Per opere in formati in cui non sia contemplata esplicitamente la pagina del titolo, con “pagina del titolo” si intende il testo prossimo al titolo dell’opera, precedente l’inizio del corpo del testo.

Una sezione “intitolata XYZ” indica una porzione dotata di un nome, del documento, il cui titolo è esattamente XYZ o contiene tra parentesi XYZ seguito dal testo che lo traduce in un’altra lingua. (In questo caso XYZ si riferisce a sezioni specifiche menzionate più avanti, quali “Ringraziamenti”, “Dediche”, “Riconoscimenti” e “Storia”). “Conservare il titolo” di tali sezioni quando si modifica il documento significa che deve rimanere una sezione “intitolata XYZ” secondo questa definizione.

Il documento più includere un’avvertenza di non garanzia accanto alla nota che indica l’applicazione di questa licenza al documento. Tale avvertenza deve essere considerata come inclusa per riferimento in questa licenza, ma solo per quanto riguarda la segnalazione che non c’è garanzia: qualsiasi altra implicazione di tale avvertenza è da ritenersi nulla e non ha alcun effetto sul significato di questa licenza.

2. COPIE ALLA LETTERA

Si può copiare e distribuire il documento con l’ausilio di qualsiasi mezzo, per fini di lucro e non, fornendo per tutte le copie questa licenza, le note sul copyright e l’avviso che questa licenza si applica al documento, e che non si aggiungono altre condizioni al di fuori di quelle della licenza stessa. Non si possono usare misure tecniche per impedire o controllare la lettura o la produzione di copie successive alle copie che si producono o distribuiscono. Però si possono ricavare compensi per le copie fornite. Se si distribuiscono un numero sufficiente di copie si devono seguire anche le condizioni della sezione 3.

Si possono anche prestare copie e con le stesse condizioni sopra menzionate possono essere utilizzate in pubblico.

3. COPIARE IN NOTEVOLI QUANTITÀ

Se si pubblicano a mezzo stampa (o con altro mezzo che possiede in genere una copertina stampata) più di 100 copie del documento, e la nota della licenza indica che esistono uno o più testi di copertina, si devono includere nelle copie, in modo chiaro e leggibile, tutti i testi di copertina indicati: il testo della prima di copertina in prima di copertina e il testo di quarta di copertina in quarta di copertina. Entrambi devono identificare l'editore che pubblica il documento. La prima di copertina deve presentare il titolo completo con tutte le parole che lo compongono egualmente visibili ed evidenti. Si può aggiungere altro materiale alle copertine. Il copiare con modifiche limitate alle sole copertine, purché si preservino il titolo e le altre condizioni viste in precedenza, è considerato alla stregua di copiare alla lettera.

Se il testo richiesto per le copertine è troppo voluminoso per essere riprodotto in modo leggibile, se ne può mettere una prima parte per quanto ragionevolmente può stare in copertina, e continuare nelle pagine immediatamente seguenti.

Se si pubblicano o distribuiscono copie opache del documento in numero superiore a 100, si deve anche includere una copia trasparente leggibile da un calcolatore per ogni copia o menzionare per ogni copia opaca un indirizzo di una rete di calcolatori pubblicamente accessibile in cui vi sia una copia trasparente completa del documento, spogliato di materiale aggiuntivo, e a cui si possa accedere anonimamente e gratuitamente per scaricare una copia trasparente completa del documento senza materiale aggiunto, usando i protocolli standard e pubblici generalmente usati. Se si adotta l'ultima opzione, si deve prestare la giusta attenzione, nel momento in cui si inizia la distribuzione in quantità elevata di copie opache, ad assicurarsi che la copia trasparente rimanga accessibile all'indirizzo stabilito fino ad almeno un anno di distanza dall'ultima distribuzione (direttamente o attraverso rivenditori) di quell'edizione al pubblico.

È caldamente consigliato, benché non obbligatorio, contattare l'autore del documento prima di distribuirne un numero considerevole di copie, per metterlo in grado di fornire una versione aggiornata del documento.

4. MODIFICHE

Si possono copiare e distribuire versioni modificate del documento rispettando le condizioni delle precedenti sezioni 2 e 3, purché la versione modificata sia realizzata seguendo scrupolosamente questa stessa licenza, con la versione modificata che svolga il ruolo del “documento”, così da estendere la licenza sulla distribuzione e la modifica a chiunque ne possieda una copia. Inoltre nelle versioni modificate si deve:

1. Usare nella pagina del titolo (e nelle copertine se ce ne sono) un titolo diverso da quello del documento, e da quelli di versioni precedenti (che devono essere elencati nella sezione storia del documento ove presenti). Si può usare lo stesso titolo di una versione precedente se l’editore di quella versione originale ne ha dato il permesso.
2. Elencare nella pagina del titolo, come autori, una o più persone o gruppi responsabili in qualità di autori delle modifiche nella versione modificata, insieme ad almeno cinque fra i principali autori del documento (tutti gli autori principali se sono meno di cinque) salvo il caso in cui gli autori non lo richiedano.
3. Dichiarare nella pagina del titolo il nome dell’editore della versione modificata in qualità di editore.
4. Conservare tutte le note sul copyright del documento originale.
5. Aggiungere un’appropriata licenza per le modifiche di seguito alle altre licenze sui copyright.
6. Includere immediatamente dopo la nota di copyright, un avviso di licenza che dia pubblicamente il permesso di usare la versione modificata nei termini di questa licenza, nella forma mostrata nell’addendum alla fine di questo testo.
7. Preservare in questo avviso di licenza l’intera lista di sezioni non modificabili e testi di copertina richiesti come previsto dalla licenza del documento.
8. Includere una copia non modificata di questa licenza.

9. Conservare la sezione intitolata “Storia”, e conservare il suo titolo, e aggiungere a questa un elemento che riporti almeno il titolo, l’anno, i nuovi autori, e gli editori della versione modificata come figurano nella pagina del titolo. Se non ci sono sezioni intitolate “Storia” nel documento, crearne una che riporti il titolo, gli autori, gli editori del documento come figurano nella pagina del titolo, quindi aggiungere un elemento che descriva la versione modificata come detto in precedenza.
10. Conservare l’indirizzo in rete riportato nel documento, se c’è, al fine del pubblico accesso ad una copia trasparente, e possibilmente l’indirizzo in rete per le precedenti versioni su cui ci si è basati. Questi possono essere collocati nella sezione “Storia”. Si può omettere un indirizzo di rete per un’opera pubblicata almeno quattro anni prima del documento stesso, o se l’editore originale della versione cui ci si riferisce ne dà il permesso.
11. Per ogni sezione intitolata “Ringraziamenti” o “Dediche”, si conservino il titolo della sezione e si conservi il senso e il tono della sezione stessa.
12. Si conservino inalterate le sezioni non modificabili del documento, nei propri testi e nei propri titoli. I numeri della sezione o equivalenti non sono considerati parte del titolo della sezione.
13. Si cancelli ogni sezione intitolata “Riconoscimenti”. Questa sezione può non essere inclusa nella versione modificata.
14. Non si modifichi il titolo di sezioni esistenti in “Riconoscimenti” o in modo da creare confusione con i titoli di sezioni non modificabili.
15. Si conservino tutte le avvertenze di non garanzia.

Se la versione modificata comprende nuove sezioni di primaria importanza o appendici che ricadono in “sezioni secondarie”, e non contengono materiale copiato dal documento, si ha facoltà di rendere non modificabili quante sezioni si voglia. Per fare ciò si aggiunga il loro titolo alla lista delle sezioni non modificabili nella nota di copyright della versione modificata. Questi titoli devono essere diversi dai titoli di ogni altra sezione.

Si può aggiungere una sezione intitolata “Riconoscimenti”, a patto che non contenga altro che le approvazioni alla versione modificata prodotte da vari

soggetti – per esempio, affermazioni di revisione o che il testo è stato approvato da una organizzazione come la definizione normativa di uno standard.

Si può aggiungere un brano fino a cinque parole come testo di copertina, e un brano fino a 25 parole come testo di quarta di copertina, alla fine dell'elenco dei testi di copertina nella versione modificata. Solamente un brano del testo di copertina e uno del testo di quarta di copertina possono essere aggiunti (anche con adattamenti) da ciascuna persona o organizzazione. Se il documento include già un testo di copertina per la stessa copertina, precedentemente aggiunto o adattato da voi o dalla stessa organizzazione nel nome della quale si agisce, non se ne può aggiungere un altro, ma si può sostituire il vecchio ottenendo l'esplicita autorizzazione dall'editore precedente che aveva aggiunto il testo copertina.

L'autore/i e l'editore/i del "documento" non ottengono da questa licenza il permesso di usare i propri nomi per pubblicizzare la versione modificata o rivendicare l'approvazione di ogni versione modificata.

5. UNIONE DI DOCUMENTI

Si può unire il documento con altri realizzati sotto questa licenza, seguendo i termini definiti nella precedente sezione 4 per le versioni modificate, a patto che si includa l'insieme di tutte le sezioni non modificabili di tutti i documenti originali, senza modifiche, e si elenchino tutte come sezioni non modificabili dell'opera composita nella licenza della stessa, e che si conservino tutte le avvertenze di non garanzia.

Nell'opera composita è necessaria una sola copia di questa licenza, e più sezioni non modificabili, se identiche, possono essere sostituite da una singola copia. Se ci sono più sezioni non modificabili con lo stesso nome ma contenuti differenti, si renda unico il titolo di ciascuna sezione aggiungendovi alla fine e fra parentesi, il nome dell'autore o editore della sezione, se noti, o altrimenti un numero distintivo. Si facciano le stesse correzioni ai titoli delle

sezioni nell’elenco delle sezioni non modificabili nella nota di copyright dell’opera composita.

Nell’opera composita si devono unire le varie sezioni intitolate “Storia” nei vari documenti originali di partenza per formare una unica sezione intitolata “Storia”; allo stesso modo si unisca ogni sezione intitolata “Ringraziamenti”, e ogni sezione intitolata “Dediche”. Si devono eliminare tutte le sezioni intitolate “Riconoscimenti”.

6. RACCOLTE DI DOCUMENTI

Si può produrre una raccolta che consista del documento e di altri realizzati sotto questa licenza; e sostituire le singole copie di questa licenza nei vari documenti con una sola inclusa nella raccolta, solamente se si seguono le regole fissate da questa licenza per le copie alla lettera come se si applicassero a ciascun documento.

Si può estrarre un singolo documento da una raccolta e distribuirlo individualmente sotto questa licenza, solo se si inserisce una copia di questa licenza nel documento estratto e se si seguono tutte le altre regole fissate da questa licenza per le copie alla lettera del documento.

7. RACCOGLIERE INSIEME A OPERE INDIPENDENTI

Una raccolta del documento o sue derivazioni con altri documenti o opere separate o indipendenti, all’interno di o a formare un volume di archivio o un supporto per la distribuzione, è detto “aggregato” se il copyright risultante dalla raccolta non viene usato per limitare i diritti legali dell’utente di tale raccolta più di quanto non consentano le singole opere. Se il documento è contenuto in un “aggregato”, questa licenza non si applica alle altre opere nell’aggregato che non siano opere derivate del documento stesso.

Se le esigenze del testo di copertina della sezione 3 sono applicabili a queste copie del documento allora, se il documento è inferiore a metà dell’intero aggregato i testi di copertina del documento possono essere

posizionati in copertine che delimitano solo il documento all'interno dell'aggregato, o l'equivalente elettronico di pagine di copertina qualora il documento fosse in formato elettronico. Altrimenti devono apparire nella copertina stampata che raggruppa l'intero aggregato.

8. TRADUZIONE

La traduzione è considerata un tipo di modifica, e di conseguenza si possono distribuire traduzioni del documento seguendo i termini della sezione 4. Sostituire le sezioni non modificabili con traduzioni richiede un permesso speciale da parte dei detentori del diritto d'autore, ma si possono includere traduzioni di una o più sezioni non modificabili in aggiunta alle versioni originali di queste sezioni non modificabili. Si può fornire una traduzione della presente licenza a patto che si includa anche l'originale versione inglese di questa licenza, comprese le note di licenza e le avvertenze di non garanzia. In caso di discordanza fra la traduzione e l'originale inglese di questa licenza, o con le note di licenza o le avvertenze di non garanzia, la versione originale inglese prevale sempre.

Se il documento contiene una sezione intitolata "Ringraziamenti", "Dedica" o "Storia", l'obbligo (sezione 4) a conservarne il titolo (sezione 1) comporterà generalmente una modifica del titolo stesso.

9. LIMITI DI APPLICABILITÀ

Non si può applicare un'altra licenza al documento, copiarlo, modificarlo, o distribuirlo al di fuori dei termini espressamente previsti da questa licenza. Ogni altro tentativo di applicare un'altra licenza al documento, copiarlo, modificarlo, o distribuirlo è deprecato e pone fine automaticamente ai diritti previsti da questa licenza. Comunque, per quanti abbiano ricevuto copie o abbiano diritti coperti da questa licenza, essi non cessano se si rimane perfettamente coerenti con quanto previsto dalla stessa.

10. REVISIONI FUTURE DI QUESTA LICENZA

La Free Software Foundation può pubblicare nuove, rivedute versioni della Licenza per Documentazione Libera GNU volta per volta. Qualche nuova versione potrebbe essere simile nello spirito alla versione attuale ma differire in dettagli per affrontare nuovi problemi e concetti. Si veda http://www.gnu.org/copyleft/.

Ad ogni versione della licenza viene dato un numero che distingue la versione stessa. Se il documento specifica che si riferisce ad una versione particolare della licenza contraddistinta dal numero o “ogni versione successiva”, si ha la possibilità di seguire termini e condizioni sia della versione specificata che di ogni versione successiva pubblicata (non come bozza) dalla Free Software Foundation. Se il documento non specifica un numero di versione particolare di questa licenza, si può scegliere ogni versione pubblicata (non come bozza) dalla Free Software Foundation.

COME USARE QUESTA LICENZA PER I VOSTRI DOCUMENTI

Per applicare questa licenza ad un documento che si è scritto, si includa una copia della licenza nel documento e si inserisca il seguente avviso di copyright appena dopo la pagina del titolo:

> Copyright © ANNO VOSTRO NOME.
> È garantito il permesso di copiare, distribuire e/o modificare questo documento seguendo i termini della Licenza per Documentazione Libera GNU, Versione 1.2 o ogni versione successiva pubblicata dalla Free Software Foundation; senza alcuna sezione non modificabile, senza testo di copertina e senza testo di quarta di copertina. Una copia della licenza è acclusa nella sezione intitolata Licenza per Documentazione Libera GNU.

Se ci sono sezioni non modificabili, sostituire il testo da “senza alcuna…” fino al punto con il seguente:

> con le sezioni non modificabili ELENCARNE I TITOLI, con i testi di copertina ELENCO e con i testi di quarta di copertina ELENCO.

Se ci sono sezioni non modificabili ma non testi di copertina, o qualunque altra combinazione dei tre elementi, si usi una versione composta delle due alternative per rispecchiare la situazione.

Se il vostro documento contiene esempi non banali di programma in codice sorgente si raccomanda di realizzare gli esempi contemporaneamente applicandovi anche una licenza di software libero di vostra scelta, come ad esempio la Licenza Pubblica Generale GNU, al fine di permetterne l’uso come software libero.

GNU Free Documentation License

VERSION 1.2, NOVEMBER 2002
Copyright © 2000, 2001, 2002 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a worldwide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standardconforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

1. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

2. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
3. State on the Title page the name of the publisher of the Modified Version, as the publisher.
4. Preserve all the copyright notices of the Document.
5. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
6. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
7. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
8. Include an unaltered copy of this License.
9. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
10. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
11. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the

substance and tone of each of the contributor acknowledgements and/or dedications given therein.

12. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
13. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
14. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
15. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply

endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License

into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warrany Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

HOW TO USE THIS LICENSE FOR YOUR DOCUMENTS

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with… Texts.” line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Sommario

Prefazione

Premessa

Capitolo primo Per volontà di una stampante

Capitolo secondo 2001: Odissea di un hacker

Capitolo terzo Un ritratto dell’hacker da giovane

Capitolo quarto Processiamo Dio

Capitolo quinto Una piccola pozzanghera di libertà

Capitolo sesto La comune dell’Emacs

Capitolo settimo Una difficile scelta morale

Capitolo ottavo Sant’Ignucius

Capitolo nono La GNU General Public License

Capitolo decimo GNU/Linux

Capitolo undicesimo Open Source

Capitolo dodicesimo Una breve discesa nell’inferno hacker

Capitolo tredicesimo La lotta continua

Epilogo Come sconfiggere la solitudine

Appendice A Terminologia

Appendice B Hack, hacker e hacking

Appendice C Licenza per Documentazione Libera GNU

Appendice D GNU Free Documentation License