SOFTWARE LIBERO E "OPEN SOURCE"

Angelo Raffaele Meo

Dopo una breve storia del movimento del software libero e delle sue realizzazioni più importanti - GNU, Linux, Internet, Apache - si discutono le ragioni per le quali alcuni studiosi ritengono il software libero preferibile al software proprietario dai punti di vista di qualità, manutenibilità e sicurezza. Infine, si disegna schematicamente lo scenario del nuovo comparto industriale costruito sui primi prodotti, comprensivo dell'informatica "embedded", delle applicazioni in tempo reale e degli apparati e sistemi per le telecomunicazioni.

Negli ultimi anni, il dibattito entro la comunità degli informatici sul software "libero" e sul software "open source" si è fatto particolarmente vivace. Moltissimi studiosi e operatori del settore hanno portato contributi interessanti nonostante la frequente contrapposizione delle opinioni espresse. Infatti, non soltanto i sostenitori del software libero si sono spesso scontrati con i difensori del software proprietario, ma anche fra le varie fazioni del software libero il dibattito è stato acceso e a volte violento.

Nel seguito di questo articolo si evidenzia il fatto che software "libero" e software "open source" non sono sinonimi, anche se i due concetti sono strettamente legati fra loro. Proprio lo scontro fra i rivoluzionari del software libero e i riformisti dell'"open source", il prodotto di un amore contrastato fra solidarietà e mercato, costituisce uno dei capitoli più interessanti di questo dibattito.

L'autore di questo articolo crede nel pluralismo ma è un fanatico sostenitore del software libero. Per questa ragione ha disatteso in parte l'invito della redazione della rivista a parlare di "open source", preferendo parlare prevalentemente di software libero.

Il primo paragrafo contiene una breve storia del movimento del software libero, della sua ideologia e delle realizzazioni più importanti: GNU, Linux, Internet. Il secondo paragrafo è dedicato a un'analisi delle ragioni per le quali l'autore considera il software libero superiore a quello proprietario dal triplice punto di vista della qualità, manutenibilità e sicurezza. Il terzo paragrafo descrive sinteticamente il comparto industriale che si sta costituendo sulle fondamenta delle prime realizzazioni del software libero e che si estende anche fuori dei tradizionali prodotti per l'ufficio, lo studio e la fabbrica abbracciando l'informatica "embedded" in migliaia di prodotti industriali, i sistemi di controllo in tempo reale, gli apparati e i sistemi per le telecomunicazioni.

L'autore, infine, auspica che i modelli di business del software libero, più vicini a quelli dell'artigiano che a quelli della grande industria, si diffondano rendendo possibile uno

FIGURA 1
Richard Stallman, il fondatore della Free Software Foundation nel 1983, nelle vesti di Saint I-GNU-tius

sviluppo più equo e armonico, con le stesse opportunità per le grandi e piccole imprese, per i Paesi ricchi ed evoluti e quelli in via di sviluppo.

1. BREVE STORIA DEL SOFTWARE LIBERO

1.1. Richard Stallman e il progetto GNU

Il più noto fra i primi protagonisti della storia del software libero è certamente Richard Marshall Stallman, "l'ultimo degli hacker" come amava definirsi (Figura 1).
La sua biografia intellettuale e personale é strettamente connessa a quella degli hacker puri dei primi laboratori informatici, per la stessa passione del calcolatore e per la stessa insofferenza verso gli intoppi burocratici, la segretezza, l'assenza di cooperazione e di scambio intellettuale. Stallman sosteneva che la sua inflessibilità morale rendeva difficile accettare le regole di scambio opportunistiche delle relazioni umane. Questa sua ricerca di informalità e di libertà lo aveva indotto a lasciare l'aristocratica e conservatrice Harvard e a trasferirsi nella comunità-baluardo degli hacker, il laboratorio di Intelligenza Artificiale del M.I.T. (*Massachussets Institute of Technology*), dove gli era stato offerto un posto di programmatore sistemista. E qui, dal 1971 al 1983, aveva ingaggiato un'infaticabile battaglia per un sistema "aperto" a tutti gli utenti, contro l'utilizzo obbligatorio di codici di accesso e contro i segreti dei sistemi di sicurezza.
La sua convinzione sulla non utilità e, anzi, sulla dannosità di non diffondere il codice di controllo della macchina, basata su premesse insieme etiche e funzionali, trovava una continua conferma nei molti problemi quotidiani connessi all'utilizzo dei computer e di altra strumentazione elettronica. L'esempio più noto, che dimostrò come i principi della proprietà intellettuale costituissero un vincolo all'efficienza impedendo di risolvere un fastidiosissimo inconveniente, riguardò il caso della stampante laser Xerox, che messa generosamente a disposizione del Laboratorio di Intelligenza Artificiale dalla stessa azienda, si fermava in continuazione. Per ovviare ai frequenti guasti, Stallman aveva pensato di modificare il programma, per attuare, in un modo più veloce, un pronto intervento cooperativo senza aspettare l'addetto della Xerox. Condizione essenziale per l'attuazione della nuova procedura era la conoscenza del codice sorgente della macchina, ma la Xerox, diversamente dal passato, negò l'accesso a quel codice, in quanto protetto dal *copyright*. Il programma proprietario non poteva essere più conosciuto e trasformato. Il mondo esterno con le sue regole stava ponendo serie barriere al lavoro della comunità degli hacker e progressivamente ne incrinava anche i valori ideali, rendendo i suoi membri più sensibili a comportamenti strumentali.
Così, quando anche il laboratorio di Intelligenza Artificiale, con il rischio dei tagli dei finanziamenti, dovette adeguarsi a quelle richieste di sicurezza che imponevano di limitare e controllare il libero collegamento ai calcolatori del laboratorio, e quando con l'acquisto di un nuovo Digital PDP-10 si interruppe la consuetudine della condivisione libera delle risorse della macchina, resa possibile dal sistema operativo ITS (*Incompatible Timesharing System*), costruito dagli stessi programmatori e basato su un'architettura aperta, Stallman lasciò il M.I.T..
Abbandonando il suo lavoro di programmatore sistemista al M.I.T., Stallman si pose co-

me primo obiettivo lo sviluppo di un sistema operativo compatibile con lo Unix di AT&T (American Telephone and Telegraph, Inc.), il sistema operativo allora più diffuso nel mondo dei minicalcolatori. Nel giorno del Thanksgiving del 1983 attraverso Arpanet comunicò alla comunità hacker e agli interessati allo sviluppo del software libero la decisione di lasciare il M.I.T. per impegnarsi nella realizzazione del nuovo sistema operativo Unix-compatibile. Il diritto di proprietà, il *copyright*, stava, infatti, contaminando e indebolendo la comunità degli hacker e, insieme ad essa, l'opportunità di un libero scambio sia materiale che intellettuale del software e dei suoi contenuti.

Stallman battezzò il nuovo sistema operativo con l'acronimo GNU, come "*Gnu is Not Unix*", una definizione formulata secondo un'antica consuetudine della comunità hacker. In altri termini: "GNU non è lo Unix di AT&T, non è quindi proprietario, ma ha le stesse funzionalità ed è compatibile con quello".

Stallman impose per GNU un requisito fondamentale, destinato a giocare un ruolo centrale nel mondo del software libero: essere ***open source***.

Un programma ***open source*** é un insieme di moduli disponibili nel cosiddetto formato "sorgente", ossia nella forma in cui le singole unità che lo compongono sono state scritte dai vari programmatori.

Il software sorgente, prima di essere utilizzato, deve essere "compilato", ossia tradotto nel codice "eseguibile" o codice di macchina (Figura 2), un'innumerevole sequenza di "uno" e di "zero", che la macchina é in grado di interpretare, ma l'uomo generalmente no, a meno di affrontare anni di duro lavoro anche su programmi relativamente brevi.

Il software proprietario viene generalmente venduto in formato eseguibile, per rendere praticamente impossibile la sua interpretazione e la sua modifica in funzione delle esigenze del suo utilizzatore. Viceversa, i programmi "open source", proprio perché disponibili in forma simbolica, che un programmatore riesce facilmente ad interpretare, sono veramente "*open*", nel senso che possono essere letti, corretti e trasformati in funzione di specifiche esigenze.

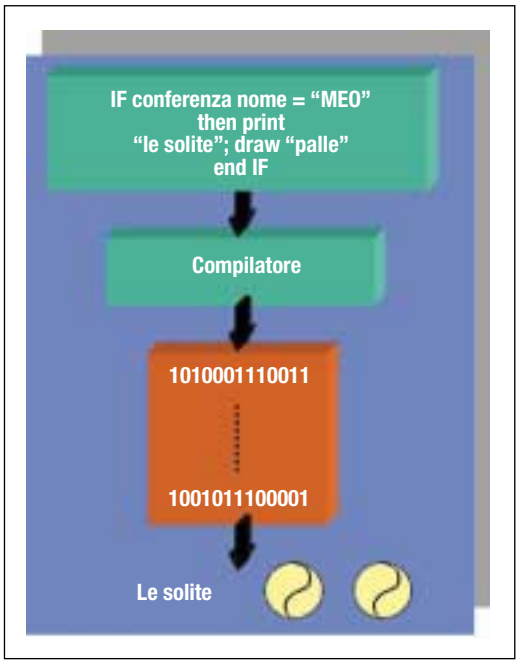

FIGURA 2
Il codice "sorgente" scritto dal programmatore viene tradotto dal compilatore nel linguaggio di macchina, fatto solo da 0 e 1, che il calcolatore a interpretare per produrre il risultato

1.2. La "Free Software Foundation" e la sua ideologia

Per portare a compimento il progetto GNU, nel 1985 Stallman costituì la FSF (*Free Software Foundation*), un'organizzazione no profit basata su contributi volontari in lavoro e in denaro. Infatti gli aderenti alla FSF possono offrire come proprio contributo sia lavoro per la scrittura di codice o documentazione, sia denaro offerto come donazione o sottoscrizione, e godere degli sgravi fiscali come tutti i contributi in beneficenza.

La costituzione di questa istituzione rispondeva all'esigenza pratica di finanziare lo sviluppo di GNU, e raccogliere attorno al progetto un insieme permanente di professionisti programmatori in grado di svolgere con continuità e a tempo pieno attività di programmazione e assistenza tecnica. Oggi la FSF occupa a tempo pieno alcuni programmatori e tecnici, oltre ad alcuni impiegati che gestiscono l'attività organizzativa. Accanto ad essi si raccoglie quel mare di volontari appassionati che l'estensione di Internet ha reso possibile.

La FSF è diventata progressivamente un punto di riferimento per gli ideatori di software e si è anche qualificata come un'istituzione di garanzia della qualità di un prodotto e di protezione dei diritti del software libero. Infatti collabora con la "League for Programming Freedom", che raggruppa accademici, pro-

grammatori e tecnici informatici, utilizzatori e piccole compagnie di software, con l'obiettivo di difendere il diritto di scrivere programmi di software libero. La Lega non si oppone al sistema legislativo del *copyright*, ma intende porre un limite alle sentenze giudiziarie che accordano una protezione diffusa a qualsiasi idea che nasca nell'ambito dell'industria informatica. In questo modo, vengono nei fatti favoriti gli interessi specifici delle grandi imprese, le sole che possono pagare le spese legali di deposito del brevetto anche nel caso delle soluzioni più banali. La FSF sostiene la League in nome di una comune difesa alla produzione di software anche per le piccole imprese che producono software proprietario, la cui attività é messa seriamente in pericolo dal sistema dei brevetti del software e dal *copyright* per le interfacce. Obiettivo della FSF è, comunque, quello più ampio di promuovere un progetto etico e un nuovo modo di lavorare.

Entrambe le dimensioni derivano direttamente dal significato attribuito al ***free software***. Le ragioni e i vantaggi connessi alla libertà del software richiamano i principi dell'etica hacker, un'etica che non é mai stata formalizzata in alcun documento ufficiale, ma che é stata spontaneamente e variamente applicata e condivisa [3, 4 , 6].

In breve essi si possono così sintetizzare.

- *Il diritto alla libera circolazione del software e alla sua duplicazione* rimandano al principio generale che tutta l'informazione deve essere libera con grande beneficio per il sistema sociale nel suo complesso. La condivisione dell'informazione é un bene potente e positivo per la crescita sociale e della democrazia, una difesa contro controlli dall'alto e pericoli tecnocratici. Esso dà all'utilizzatore l'opportunità di interagire con il prodotto e di controllarlo. Aiuta a stimolare lo sviluppo della conoscenza e a diffonderla. In senso ampio, quindi, può favorire il processo di partecipazione contribuendo a formare un membro della comunità o un cittadino più preparato.
- *La rivoluzione digitale e la diffusione dei calcolatori*, rendendo più semplice lo scambio di informazione, possono apportare un beneficio generale.

Un libero scambio di informazioni, soprattutto quando si traduce in uno strumento importante come un programma per computer, promuove una maggiore creatività complessiva. Si eviterebbe, così, di sprecare tempo, risorse, energie per replicare quello che altri già fanno, cioè di perdere tempo per reinventare la ruota.

- *Il modo migliore per favorire il libero scambio delle informazioni* è quello di promuovere un apprendimento diffuso e qualificato attraverso sistemi "aperti", che non pongano barriere fra il lavoro e l'informazione. Nella tradizione e nell'etica degli hacker le barriere alla circolazione dell'informazione costituiscono pesanti limiti alla conoscenza e, nel divieto di accesso all'artefatto, cioè al programma, viene individuato un ostacolo alla creatività e alla libera espressione del pensiero.
- *La libertà di modificare il software* richiama l'imperativo del buon inventore artigiano "di metterci le mani" per capire il funzionamento delle cose e per migliorarle.
- *Il piacere e il divertimento* sono un importante incentivo alla programmazione. Il computer non é solo uno strumento funzionale per facilitare compiti ripetitivi, ma anche un mezzo per estendere l'immaginazione personale. La flessibilità dei programmi, le loro possibilità di evoluzione costituiscono una sfida continua per chi li usa in modo intelligente.
- *Le possibilità di innovazione* continua offrono un contributo alla crescita dell'intelligenza e allo sviluppo della professionalità, a differenza di procedure routinarie che sono, invece, vere e proprie barriere.

In sostanza, ad un'organizzazione del lavoro burocratica si vuole contrapporre un'organizzazione interattiva e creativa, a un'organizzazione sociale basata su status e ruoli definiti dal reddito o dalla collocazione so-

Il termine "free" che, nella lingua inglese, ha il doppio significato di libertà e gratuità, é inteso nella accezione di libertà e non di prezzo. Il **free software** può anche essere venduto ma permette comunque di capire le modalità di lavoro del programma e di adattarlo alle proprie esigenze: offre la possibilità di redistribuire le copie; aiuta a migliorare il programma stesso; estende le possibilità di distribuire agli altri il contributo del proprio miglioramento; consente il lavoro in collaborazione senza vincoli.

ciale, si contrappone una comunità di eguali basata sul merito.

1.3. La Torre di Babele delle libertà del software: il copyleft

La figura 3 rappresenta schematicamente la complessa torre di Babele delle libertà del software. Al gradino più basso, ossia al minimo livello della libertà, è ovviamente allocato il software proprietario. Subito sopra sta il cosiddetto **freeware**.

Saltando mille piani intermedi si arriva ai massimi livelli della libertà.

Il **freeware** è software distribuito gratuitamente, ma disponibile soltanto in formato eseguibile. La diffusione di freeware è una tecnica di "marketing" talora adottata da aziende produttrici di software proprietario per promuovere un nuovo prodotto. La logica del freeware è l'opposto di quella del software libero, perché il prefisso "free" fa riferimento al prezzo e non alla libertà.

In teoria, il livello massimo della libertà con cui può essere distribuito il software è quello che corrisponde alla denominazione di "pubblico dominio", che da molti anni è spesso adottata nella comunità degli informatici. Un prodotto software di pubblico dominio può anche essere utilizzato per la realizzazione di software proprietario, cosi come è avvenuto anche per molti programmi liberi che essendo distribuiti con licenze permissive sono stati trasformati in prodotti chiusi e proprietari.

Tuttavia, Stallman non ritenne opportuno rendere GNU di pubblico dominio, anche se quella soluzione avrebbe consentito la massima diffusione del prodotto: un programma completamente libero avrebbe rischiato di estinguersi e di limitarsi ad un'entusiasmante esperienza creativa, una volta che le sue varianti fossero state utilizzate per un uso proprietario. In tal modo la catena del lavoro e dell'uso cooperativo si sarebbe spezzata e avrebbe interrotto anche il valore insito nel prodotto iniziale, in quanto l'ultima versione aggiornata avrebbe avuto molte più probabilità di diffondersi vanificando tutto il lavoro precedente.

Se sul piano astratto distribuire GNU nel sistema di pubblico dominio avrebbe potuto consentire una diffusione più ampia, sul piano concreto nel tempo ne avrebbe tratto vantaggio solo il sistema proprietario. In un certo senso il fine di una diffusione quantitativa senza preservare il contenuto e trascurando le modalità della diffusione avrebbe costituito un ostacolo alla creazione del software libero e alla costruzione di una comunità di liberi programmatori.

La scelta fu quella di proteggere il prodotto con un tipo nuovo di licenza, formalmente denominata G.P.L. (*General Public Licence*) ma scherzosamente chiamata "copyleft" (la parola non compare su alcun dizionario della lingua inglese) o "diritto di copia di sinistra", in contrapposizione al più noto "*copyright*", interpretato come "diritto di copia di destra". Ognuno può modificare e distribuire il prodotto, ma non si possono apporre restrizioni individuali sul prodotto redistribuito. Il "copyleft", che Stallman chiama anche "permesso d'autore", consente a chi acquista un programma di utilizzarlo in un numero indefinito di copie, di modificarlo a suo piacimento, di distribuirlo nella forma originale o modificata, gratuitamente o a pagamento, alle sole condizioni di distribuirlo in formato sorgente e di imporre a chiunque acquisisca il prodotto di firmare lo stesso tipo di contratto.

FIGURA 3
La torre di Babele delle libertà del software

Il modello del *copyleft* permette di dare un fondamento giuridico a un mercato costruito sulla, non mera, appropriazione privata della proprietà intellettuale. Inoltre, la soluzione del *copyleft* fornisce uno stimolo a contribuire alla crescita del software libero, in quanto costituisce una garanzia di fiducia sulla stabilità di un patto di libera circolazione del software e anche un modo per generare un meccanismo di creazione di risorse finanziarie.

1.4. Linus Torwalds e Linux

Nell'arco di circa sette anni, la FSF realizzò un'enorme mole di programmi: "*compilatori*", ossia traduttori, da un linguaggio sorgente di alto livello come il C nel linguaggio di macchina; "*text editor*", ossia programmi per la compilazione e la correzione di documenti; "*debugger*", ossia strumenti per l'analisi di programmi con l'obiettivo di identificare i bachi; interfacce varie; altri strumenti di utilità generale. Nel 1990 il sistema GNU era quasi completo, ma mancava ancora il "*kernel*" o "nucleo", ossia l'insieme dei programmi di base che consentono la gestione delle risorse fondamentali, come l'unità di calcolo e la memoria centrale. Il nucleo era certamente la parte più importante di GNU, ma la sua realizzazione era stata rinviata in attesa della promessa liberalizzazione come software libero di un micronucleo, sviluppato dalla Carnegie Mellon University e successivamente ampliato dall'università dello Utah.

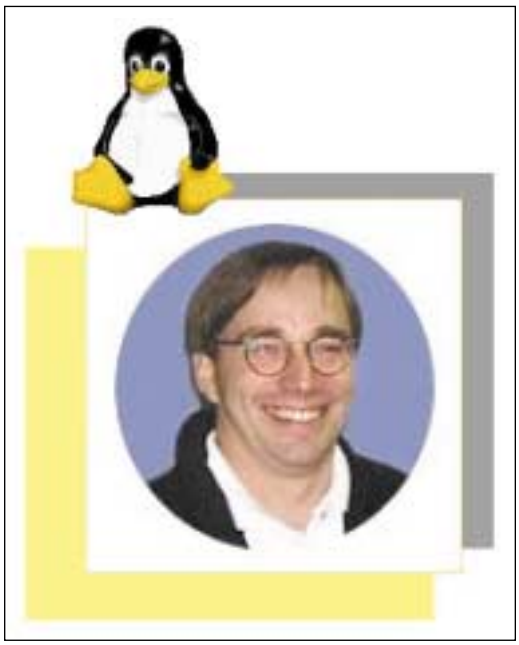

FIGURA 4
Linus Torwalds e il pinguino, simbolo di Linux, il suo sistema operativo

Fortunatamente a questo punto una nuova storia e un nuovo protagonista si intrecciano e portano a compimento l'iniziativa di Stallman (Figura 4).
Nel 1990 uno studente ventenne di informatica dell'università di Helsinki, Linus Torvalds, che si diletta a programmare il calcolatore trascurando lo studio, decide di comprarsi un calcolatore nuovo. Ovviamente, gli elaboratori della classe del gigantesco "*mainframe*" dell'università su cui ha imparato a programmare, sono fuori dalla sua disponibilità finanziaria di studente, mentre il vecchio Commodore attaccato al televisore, che usa a casa da tempo, non gli consente di andare oltre il programmino giocattolo. I nuovi personal computer che montano il microprocessore Intel 386 sembrano rappresentare un ottimo compromesso fra costo e prestazioni, ma il sistema operativo che su di essi viene installato, il vecchio DOS (*Disk Operating System*) di Microsoft, non gli consente di sviluppare software di alto livello, non permettendo, in particolare, di programmare "processi" concorrenti, ossia i moduli software fatti per operare in parallelo.
L'ideale sarebbe stato installare sul personal computer il tradizionale Unix, uno dei più diffusi nel mondo, ma i 5.000 dollari di costo lo rendono inaccessibile. Così, Linus decide di scrivere da solo, partendo da Minix, un sistema operativo didattico molto diffuso nelle università, il nucleo di un nuovo sistema operativo, un clone di Unix, per dotare il personal computer delle funzionalità di base di un elaboratore di fascia alta.
Nella primavera del 1991 il nucleo del nuovo sistema operativo, versione 0.01, é pronto. Gestisce i *file*, ossia i documenti, e il *file system*, ossia l'organizzazione gerarchica dei documenti in cartelline e grandi cartelle, con la stessa logica di Unix, é dotato della funzionalità di emulazione di un terminale e contiene alcuni *driver* di base per pilotare le unità periferiche. Sostituendo la consonante finale del proprio nome con la "x" di Unix e adottando il pinguino come suo simbolo, Linus battezza il suo prodotto "Linux", e fa così una prima scelta felice. Ancora più felice e importante é la seconda

scelta, quella di diffondere il nuovo sistema operativo su Internet, mettendolo a disposizione di chiunque sia interessato a utilizzarlo, senza chiedere altra contropartita oltre alla collaborazione per migliorarlo ed espanderlo.
Il suo invito é raccolto da centinaia di giovani programmatori in tutto il mondo, che nell'arco di pochi anni, in un telelavoro collettivo guidato da quello splendido organizzatore che si rivelò Linus, trasformano un interessante prototipo scientifico in una vera e propria linea di prodotti industriali [5, 9].
Come si vedrà meglio più avanti, oggi Linux é operante non soltanto sull'architettura del personal computer Intel, ma anche su tutte le piattaforme più importanti, ed è installato su milioni di calcolatori.
Linux é diventato un sistema molto utile e diffuso soprattutto fra la popolazione più acculturata del mondo degli informatici programmatori e accademici ed é utilizzato da molte società di servizi come le poste negli Stati Uniti. Recentemente il governo cinese ha deciso di adottare estensivamente Linux (una variante specifica: TurboLinux) per tutto il suo sistema amministrativo. Molte sono le ditte che vendono Linux e offrono assistenza tecnica. Tuttavia Linux é disponibile gratuitamente.
La fortuna di Linux non é solo quella di un prodotto largamente portabile e disponibile, ma anche quella di essere caratterizzato da forti principi di progettazione e da un solido modello di sviluppo, basato su una stretta collaborazione di una moltitudine di progettisti e programmatori.
Linus Torvalds accetta la logica del *copyleft* e l'inserimento non banale del suo sistema operativo nella grande cornice di GNU: nasce così un sistema operativo completo, il GNU/Linux, pienamente compatibile con lo Unix proprietario, ma completamente libero. Nel giro di tre anni GNU/Linux diviene competitivo per affidabilità e sicurezza con le versioni commerciali più importanti di Unix.

1.5. Internet

Un'analisi attenta della sua storia mostrerebbe con chiarezza che Internet, oltrechè madre è stata anche figlia del software libero. Mi limito a ricordare due esempi importanti.
Quando la rete conteneva migliaia di nodi, i *router* erano configurati a mano. Ciò significa che il programmatore di un router doveva precisare per ogni indirizzo-destinazione di un pacchetto in arrivo, su quale linea di uscita, e quindi verso quale destinazione intermedia, si dovesse instradare lo stesso pacchetto di ingresso. In presenza di milioni di nodi e milioni di indirizzi diversi, questa soluzione non è più attuabile.
Le soluzioni ideate per configurare dinamicamente i router e definire automaticamente i percorsi ottimali sono il frutto di un'intelligenza collettiva di livelli qualitativi e quantitativi così elevati da indurre alla convinzione che quelle soluzioni molto difficilmente sarebbero potute nascere nell'ambito di una sola azienda, anche molto importante.
Un secondo esempio importante è rappresentato dal noto meccanismo del "World Wide Web", la ragnatela che avvolge il mondo, quello che consente ad un utilizzatore che stia leggendo una pagina proveniente dagli Stati Uniti di "cliccare" su una parola sottolineata e ottenere istantaneamente e automaticamente un'altra pagina, proveniente forse dal Giappone. Questo meccanismo semplice e immediato, che si deve al CERN (*Conseil European pour la Recherché Nucleaire*) di Ginevra e, in particolare, ai due fisici Tim Berners-Lee e Robert Cailliau, può essere appreso in pochi minuti anche da un bambino e pertanto ha clamorosamente migliorato la fruibilità della Rete, determinando la sua attuale diffusione in ogni strato della popolazione.
La Rete è stata progettata da migliaia di ricercatori e programmatori di tutto il mondo mettendo in comune un enorme patrimonio di intelligenze, conoscenze, risorse. Questo sforzo collettivo è stato coordinato da un unico organismo, l'IETF (*Internet Engineering Task Force*), un'associazione libera di alcune migliaia di studiosi, aperta a chiunque sia interessato ai progetti di Internet, come dimostra il fatto che all'IETF ci si iscrive a titolo personale, e non come rappresentanti di qualche istituzione pubblica o privata.
Un enorme patrimonio di conoscenze è contenuto in migliaia di documenti o R.F.C. (*Request For Comments*), generalmente formulati in modo informale per migliorare la chia-

rezza, in ossequio a una raccomandazione altrettanto chiara: "Scriveteli nel cesso, ma scriveteli semplici e chiari".
Oggi l'IETF è composta da 150 gruppi di lavoro, che coprono tutte le aree scientifiche e tecnologiche: dalla rete e dai servizi di utente, alla crittografia e autenticazione, dalle soluzioni per il routing ai problemi della gestione. Il frutto dell'enorme lavoro di quei gruppi, dalla documentazione scientifica al software sviluppato, dalle proposte di standard agli standard veri e propri, è disponibile gratuitamente in Internet e può essere liberamente utilizzato anche a fini commerciali. I processi produttivi e decisionali, la documentazione scientifica, il codice scritto, gli standard sono open source ossia sono non soltanto, e non tanto, gratuiti, quanto piuttosto "trasparenti", in modo che siano ben chiari sia i principi di base sia la logica operativa, per consentire ulteriori miglioramenti e progressi.
Internet è cresciuta più rapidamente di ogni altra tecnologia nella storia, molto più rapidamente delle ferrovie, della telefonia, della radio e della televisione. Il merito fondamentale di quel rapido progresso è da attribuirsi allo spirito della collaborazione che anima l'IETF e all'adozione di due principi fondamentali: "*rough consensus and running code*" e "*fly before buy*". In altri termini: "la fantasia al potere", come predicavano gli studenti nel '68, ma anche molta concretezza, "cose che funzionano, e non soltanto idee".
Narra Giovanni nel suo Vangelo "*Andò Simon Pietro, e tirò a terra la rete piena di 153 grossi pesci. E sebbene erano tanti, la rete non si strappò...*". È l'annunciazione di Internet.
Internet è il frutto di due miracoli. Il primo è il miracolo tecnologico del collegamento simultaneo di 153 milioni di calcolatori. Il secondo è il miracolo di un'invenzione molto complessa ed importante nata fuori della logica del mercato, della competizione e della gerarchia.
Forse un giorno valuteremo Internet come una delle più importanti invenzioni dell'Uomo, perché mai nella storia dell'umanità è stato disponibile uno strumento così efficace per la diffusione delle conoscenze e la crescita del sapere.

2. QUALITÀ, ASSISTENZA E SICUREZZA

2.1. Qualità del software libero e di quello proprietario

L'autore di questo articolo ha tentato molte volte con poco successo di convincere imprenditori e responsabili di aziende italiane ad adottare software libero per costruire i vari programmi applicativi. – "È vero che non dovrò pagare le royalties" – gli osservavano – "ma chi mi garantirà la qualità del prodotto? chi mi assisterà nella manutenzione? –.
E ancora, su sollecitazione del solito grande esperto della nota società internazionale di consulenza: "Il software libero è noto a tutti, e quindi "hacker" e "cracker" ci sguazzano. Chi mi proteggerà dalle intrusioni?"
Esaminiamo separatamente le tre questioni della qualità, dell'assistenza tecnica e, infine, della sicurezza, iniziando dalla prima e probabilmente la più importante, la qualità.
È molto difficile formulare un giudizio obiettivo sulla qualità di un prodotto software. Probabilmente come conseguenza della soggettività del giudizio il dibattito sul confronto della qualità del software proprietario e del software libero è cosí infuocato. Da un lato, i produttori del software proprietario sostengono che non può offrire alcuna garanzia di qualità il frutto di un lavoro collettivo svolto con modalità caotiche e anarcoidi da una pluralità di lavoratori disseminati sulla superficie dell'intero pianeta, riuscendo facilmente con questa argomentazione a convincere la grande maggioranza degli imprenditori e dei manager nostrani.
Sull'altro fronte, i sostenitori del software libero rilevano la qualità eccezionale dei prodotti più noti del loro mondo - Linux, Apache, BSD (*Berkeley Software Distribution*, UNIX) - e ne deducono una legge universale sulla superiorità del software libero.
Uno dei più importanti esponenti di questa scuola di pensiero, artefice anche della *Debian Society*, è Eric Raymond, che in un saggio molto noto nella comunità mondiale degli informatici. *La cattedrale* e *il bazar*, [7] ha raccontato e analizzato la storia di Fetchmail, un programma libero per la gestione remota della posta elettronica, traendo dal successo di quell'esperienza indicazioni di carattere generale sull'intero universo del software libero.

Nel suo saggio Raymond confronta il processo di produzione del software caratteristico di un'azienda importante, ove ogni atto è parte di un rituale collettivo preordinato, preciso, gerarchico, con il processo caratteristico del software libero, ove come in un bazar una moltitudine di soggetti si scambiano beni in modo caotico, senza disciplina e autorità superiori. Diciannove regole d'oro costituiscono secondo Raymond le chiavi del successo di un prodotto software; quasi tutte sono applicabili soltanto nel bazar.
Sinceramente, molte delle argomentazioni proposte da Raymond suscitano, a giudizio di chi scrive, qualche perplessità. Complessivamente, la cattedrale sembra, a chi scrive, un luogo più idoneo del bazar allo sviluppo del software.
Tuttavia, due ragioni fondamentali potrebbero spiegare la presunta superiorità dei migliori prodotti del software libero rispetto ai corrispondenti prodotti del software proprietario.
La prima è la possibilità di buttare in campo centinaia di utenti programmatori nella fase di debugging.
Raymond attribuisce a Torvalds il seguente enunciato, noto appunto come "Teorema di Linus": "Given enough eyeballs all bugs are shallow" ossia " Date tante pupille, tutti i bachi diventano bazzecole".
La seconda ragione è sintetizzata dall'osservazione che nove donne in un mese non fanno un bambino. I programmatori liberi lavorano per la soddisfazione personale e la stima degli amici, senza fretta avendo la qualità come obiettivo centrale. I programmatori delle aziende che vendono software su licenza devono produrre una nuova versione ogni due anni, per non uscire dal mercato e produrre nuovo fatturato. Ma la complessità dei nuovi prodotti cresce molto rapidamente e soltanto sacrificando la qualità è possibile superare le sfide della complessità.

2.2. Assistenza tecnica

Dal punto di vista industriale la questione dell'assistenza tecnica è uno dei fattori più importanti che giocano a favore dei grandi a danno dei piccoli. La piccola software house di Paperino, ad esempio, che producesse un prodotto di grande interesse per il mercato non riuscirebbe a diffonderlo perché non disporrebbe delle risorse finanziarie per creare un'adeguata rete di assistenza. Per contro, Paperone ha attuato un meccanismo per l'assistenza ai propri prodotti che, secondo alcuni fanatici sostenitori del software libero, rivela la sua natura diabolica ma che più razionalmente, si deve riconoscere, dimostra il genio della finanza. Infatti, la multinazionale di Paperone incarica aziende specializzate di organizzare corsi di specializzazione per aspiranti tecnici di assistenza e incassa denaro per l'iscrizione ai corsi, l'organizzazione degli esami e la concessione dei diplomi. Così, la creazione di una rete capillare di assistenza basata su competenze di alto livello, che a Paperino costerebbe come mille fatturati annui, contribuisce al riempimento delle leggendarie casseforti di Paperone.
Tuttavia, a dispetto di questi meccanismi finanziari e dell'obiettiva capacità delle multinazionali di creare reti di assistenza di alto livello, la superiorità delle multinazionali del software proprietario rimane discutibile.
Infatti, quando la complessità del problema dell'attuazione della funzionalità particolare o dell'integrazione con altri prodotti supera un certo livello, allora diventa necessario per il tecnico conoscere a fondo cosa fa il prodotto, e come lo fa, e per questa conoscenza è necessaria l'analisi del codice sorgente, che Paperone non consente per non correre il rischio di essere copiato (o scoperto nel caso che uno dei suoi programmatori avesse copiato qualcosa).
Così, alla superiorità economica si contrappone un'intrinseca debolezza tecnica che in prospettiva non potrà non condizionare pesantemente lo sviluppo del software proprietario.

2.3. Sicurezza

Talvolta qualche sostenitore del software proprietario propone un'argomentazione quasi "truffaldina" che all'analisi non troppo meditata di un dilettante potrebbe apparire ragionevole: afferma, ad esempio, che i prodotti chiusi possono contare su livelli di sicurezza più elevati, e in particolare su una migliore crittografia, rispetto ai prodotti aperti, che adottano algoritmi noti a tutti in quanto il loro codice può essere letto e studiato a fondo.
E invece è proprio vero l'esatto contrario, infatti, in linea teorica, la crittografia costruita su chiavi sufficientemente lunghe è assoluta-

mente imbattibile in quanto non è noto, e forse non lo sarà mai, un algoritmo per batterlo. Ma i sistemi crittografici spesso presentano punti deboli, generalmente rappresentati da chiavi troppo corte e da soluzioni tecniche per la distribuzione delle chiavi attaccabili con vari stratagemmi. Cosí, sicurezza e crittografia sono obiettivi molto difficili, forse i più difficili dell'informatica e, per raggiungere una ragionevole tranquillità, occorre sottoporre procedure e algoritmi a verifiche severe condotte da molti studiosi di alto livello, possibilmente appartenenti ad ambienti diversi.
Per sapere se una serratura è sicura, occorre affidarla a uno o più esperti, che la aprano e la studino a fondo per verificare che non possa essere aperta con un idoneo chiavistello. Se la serratura non può essere aperta, perché affogata, ad esempio, in un blocco di cemento, non è possibile verificare se quella è una buona o cattiva serratura. Un ladro professionista potrebbe corrompere il fabbro che ha costruito la serratura, farsi spiegare come funziona e costruire un idoneo chiavistello; viceversa, se la serratura è perfetta, non esiste un chiavistello in grado di aprirla.

3. L'INDUSTRIA DEL SOFTWARE LIBERO O OPEN SOURCE

3.1. Gli operatori puri del software libero

Nei primi anni '90, mentre Linus Torvalds sviluppava le prime versioni di Linux, altre novità importanti si manifestavano in diverse comunità scientifiche.
In California, Bill Jolitz [2] portava a compimento la distribuzione Net/2 dello Unix della Università di California di Berkeley, chiamato *BSD*, con lo stesso obiettivo di Stallman e di Torvalds, ossia offrire al mondo uno Unix libero dal *copyright* della AT&T. Quella prima versione era rivolta ai personal computer della famiglia Intel 80386, ma fu il progenitore di una ricca famiglia di prodotti liberi. Oggi i noti Yahoo! e Hotmail utilizzano uno dei figli recenti di 386BSD, chiamato Free*BSD*, per gestire i loro servizi in rete che sono rivolti a milioni di utenti, mentre altre imprese utilizzano due altri figli prediletti, come Open*BSD* [8], orientato alle esigenze della sicurezza, e Net*BSD*, finalizzato all'ottimizzazione delle prestazioni nella gestione dei protocolli di Internet.
Nel 1994 iniziava lo sviluppo di Apache, quello che è oggi considerato il più importante e diffuso "www server" a livello mondiale.
In quel tempo, il mercato dei server era dominato da due prototipi, quello sviluppato al CERN di Ginevra, il laboratorio comunitario finalizzato alle ricerche sulle particelle ove un paio di anni prima era nato il protocollo HTML (*Hyper Text Markup Language*) per la distribuzione dell'informazione su Internet, e quello del NCSA (*National Center for Supercomputer Applications*) dell'Università dell'Illinois.
Il leader dello sviluppo di Apache fu Brian Behlendorf, un tecnico e programmatore allora ventunenne, come Linus Torvalds tre anni prima, quando aveva iniziato la programmazione di Linux. Behlendorf gestiva uno dei primi *Internet Service Provider* (ISP) commerciali e sentiva il bisogno di continui miglioramenti del servizio, ma i ricercatori dell'NCSA recepivano lentamente i suoi suggerimenti: in parte, perché il gruppo originario che aveva sviluppato il server si era staccato per fondare Netscape e sviluppare un nuovo *browser* e in parte, perché in quel tempo l'Università dell'Illinois stava trattando la vendita delle licenze del proprio *server* con alcune società commerciali.
Per questa ragione Behlendorf e altri sei giovani progettisti e programmatori costituirono un gruppo autonomo che rapidamente si espanse e produsse quell'autentica meraviglia che è Apache. La diffusione di questo prodotto, prima nella comunità scientifica e successivamente anche nelle imprese, è stata eccezionale. I server Apache, che erano il 31% del totale nel '96 sono saliti al 55% nel '99 ed oggi, inizio del 2002, si valuta che essi siano più del 60%.
Nel 1987 Larry Wall, un programmatore della Burroughs, un'azienda, ora confluita nella Unisys, sentì il bisogno di uno specifico linguaggio di programmazione che consentisse come il C lo sviluppo di algoritmi anche molto complessi e nello stesso tempo svolgesse funzioni di gestione del sistema operativo. Nacque così il Perl (*Practical Extraction and Reporting Language*) che fu subito distribuito in rete ed ora è diffuso nell'ambiente di programmatori di server Apache.
Linux, BSD, Apache e Perl sono i quattro progetti più noti che sono nati negli anni '90 e sono stati coronati dal successo. Non sono co-

munque i soli, perché non vi è area dell'elaborazione e della trasmissione dei dati che non abbia visto nascere almeno un progetto significativo volto allo sviluppo di software libero.
Mentre in ambiente accademico o molto vicino all'accademia si sviluppavano questi importanti progetti, nascevano e si espandevano rapidamente le prime imprese per l'utilizzazione commerciale del software libero.
Red Hat è forse la più nota fra le imprese che hanno il software libero come "core business". Distribuisce in tutto il mondo il sistema operativo Linux, programmi vari associati a Linux e la documentazione relativa. Inoltre, vende servizi di assistenza, progettazione e consulenza. Infine, ha creato una consociata, Red Hat Advanced Development Labs, ove operano alcuni programmatori ad alto livello, ai quali è stato affidato il compito di collaborare a progetti internazionali come il noto GNOME (*GNU Network Object Model Environment*) [1], nato in ambiente messicano con l'obiettivo di sviluppare il corrispondente software libero della suite Office di Microsoft per le applicazioni dell'ufficio.
Una storia parallela è quella di VA Research, che vende prodotti hardware sui quali sono stati installati moduli software della famiglia GNU/Linux. Red Hat e VA Research non sono le uniche aziende che abbiano scelto il software libero come oggetto preminente della loro attività industriale. Tra le altre si ricordano TurboLinux, Caldera, Linuxcare, Cignus, Abisoft, BitWizard, Digital Creations; anche alcune piccole aziende italiane, come la milanese Prosa e la pisana Icube, operano esclusivamente o prevalentemente nell'area del software libero.

3.2. Il software libero alla guerra

Oltre agli operatori che potremmo definire "istituzionali" in quanto operano soltanto nel settore del software libero rispettando completamente le sue regole - distribuzione del codice sorgente e libertà di duplicazione e modifica, in primis -, come Red Hat, Caldera, Debian, TurboLinux, VALinux, anche i grandi, tradizionali protagonisti dell'informatica sono entrati, o stanno entrando, nel nuovo comparto industriale.
Il più importante di questi è certamente IBM, la madre dell'informatica mondiale, che continua a dominare il mercato dei *mainframe*, i grossi calcolatori che costituiscono tuttora l'intelligenza centrale delle grandi organizzazioni pubbliche e private, dai ministeri alle banche, dalle grandi imprese produttive alle società della distribuzione e dei trasporti.

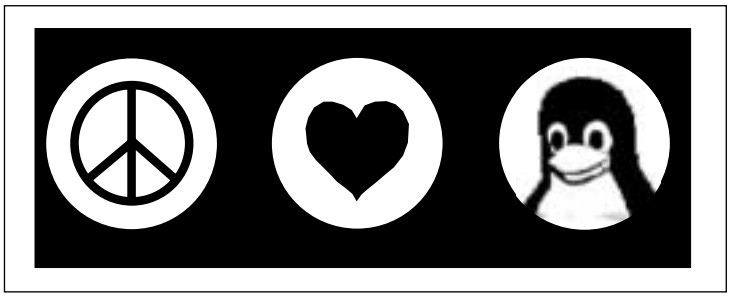

FIGURA 5
Il trittico "Peace, Love and Linux" proposto da IBM

Oggi, la gelosia della progenitrice e forse il timore della propria scomparsa induce i manager dell'IBM a sognare che il software libero possa erodere il mercato e il potere di Microsoft. In particolare, i responsabili di IBM sognano che Linux soppianti Windows nella ricca prateria dei personal computer e dei mini. Per questa ragione, tutte le linee di progetti IBM, dai personal computer ai server, dai cluster ai mini, dai mainframe agli elaboratori paralleli di alte prestazioni, supportano Linux.
Per promuovere questa scelta, la stessa IBM ha lanciato una campagna pubblicitaria basata sul trittico "peace, love and Linux" (Figura 5). Lo slogan è piaciuto ai pittori di strada americani che hanno ridipinto quel trittico sulle pareti e sui marciapiedi di molte città.
Molto diversa, anche se finalizzata alla stessa tipologia di business, appare la politica di Sun Microsystems. Questa società trae la maggioranza dei propri profitti dalla vendita di minielaboratori utilizzati prevalentemente come server nelle reti aziendali o in Internet. Su queste macchine installa uno Unix proprietario, chiamato Solaris, che è seriamente minacciato da Linux. Per fronteggiare questo pericolo, i manager di Sun hanno recentemente deciso di distribuire liberamente il codice sorgente di Solaris, ma le vendite di questo sistema operativo sono ancora proposte sulla base di *royalty* per copia, senza diritto di duplicazione o ridistribuzione.
A metà degli anni '90, in concomitanza con l'esplosione dell'importanza di Internet, Sun ha proposto al mercato un interessante linguaggio di programmazione, chiamato Java. Oltre al valore concettuale derivante dalle proprietà

formali, e in particolare dall'orientamento alla cosiddetta "programmazione ad oggetti" che consente migliore qualità e maggiore produttività nello sviluppo del software, Java è finalizzato alla realizzazione di programmi applicativi per Internet. I programmi di navigazione, i cosiddetti *browser*, insieme alle pagine WWW (*World Wide Web*) possono estrarre dai server remoti opportuni programmi chiamati "applet", scritti, appunto, in Java.
Per incentivare l'impiego di Java e l'utilizzo dei propri prodotti hardware e software, Sun ha promosso un'iniziativa mondiale *open source*, chiamando a raccolta i programmatori liberi di tutto il mondo. La libertà di questi non è comunque quella massima rappresentata dalla licenza *GPL* di Stallman, in quanto Sun si è riservata il compito del coordinamento globale e il ruolo dei programmatori liberi è confinato agli strati applicativi. In sostanza, il sogno di Sun è rappresentato da un'enorme terra emersa di programmi *free* galleggianti sul mare della tecnologia proprietaria di Sun.
Un terzo ricco neofita del credo del software libero è Oracle, l'azienda mondiale che domina l'importante mercato dei D.B.M.S. (*Data Base Management System*), ossia dei sistemi software per la gestione degli archivi di dati.
Il Presidente di Oracle, Larry Ellison, è un nemico giurato di Bill Gates per antipatia personale e feroci scontri sul mercato.
Appoggiando Linux, Larry Ellison conta di indebolire Microsoft, come è nel disegno di IBM. Inoltre, la diffusione di Linux consentirebbe l'indipendenza dei DBMS di Oracle da sistemi operativi di concorrenti pericolosi, come Solaris di Sun. In sintesi, il sogno di Oracle è, come quello di Sun, rappresentato da una terra ferma proprietaria, il suo DBMS, galleggiante su un mare libero e gratuito.
Anche Intel, il più importante produttore mondiale di microcircuiti in generale, e di microprocessori in particolare, si sta innamorando di Linux. Per due decenni la "santa alleanza" nota come Wintel, ossia Windows + Intel ha retto a ogni tentazione disgregatrice, condizionando nel bene e nel male lo sviluppo dell'informatica e delle sue applicazioni. Nel bene, in quanto ha reso possibile la diffusione mondiale di centinaia di milioni di personal computer a basso costo. Nel male, come associazione perversa: Intel, realizzando microprocessori sempre più potenti, ha reso possibile l'utilizzo di sistemi operativi e programmi applicativi di Microsoft sempre più ricchi e complessi, mentre la crescita della complessità di questi ultimi ha reso necessario l'impiego di microprocessori sempre più veloci e di memorie sempre più capaci. Negli ultimi anni, però, i vincoli di sangue impressi il giorno della nascita di Microsoft si sono allentati. Probabilmente a cercare l'indipendenza per prima è stata prevalentemente Microsoft che, ad esempio, ha aperto il sistema operativo Windows NT alle architetture non Intel (come Power PC di IBM, Motorola ed Apple, ALFA di Digital Equipment Corp., SPARC di Sun) e recentemente ha proposto il nuovo sistema operativo WindowsCE per le piccole unità di calcolo, come i *palm top computer* (i calcolatori che stanno sul palmo della mano) e le varie *Internet Appliances*, le micro-unità per navigare in rete. In compenso, Intel non ha ripudiato Microsoft ma non nasconde più la propria simpatia per Linux.

3.3. La guerra civile delle licenze

Durante gli anni '80 il tipo di licenza adottato prevalentemente nella distribuzione del software libero fu la *GPL* di Stallman, di cui sono state individuate le caratteristiche principali nel primo capitolo di questo articolo.
Negli anni '90 è prevalsa invece, nell'ambito della comunità dei programmatori del software libero, una linea politica più liberale e pragmatica volta a promuovere la diffusione del software *open source* anche a costo del sacrificio dei principi di Stallman, compreso il "diritto di copia di sinistra" e il rigore della GPL. In particolare, l'organizzazione Debian, nata nel 1995 per promuovere la diffusione di Linux, propose le cosiddette "*Debian Free Software Guidelines*" che consentivano una grande flessibilità nell'uso del software libero, compresa la possibilità di fondere software libero e software proprietario.
Queste *guideline* furono adottate e chiarite un paio d'anni dopo dalla cosiddetta "Open Source Iniziative" in un documento noto come "Open Source Definition". Quel documento precisava che le licenze non debbono porre vincoli sul software che viene distribuito insieme al software *open source* e che pertanto software libero e software proprietario

possono essere liberamente intermescolati.
Contro i pragmatici *dell'Open Source Initiative*, sono violentemente insorti Stallman e i puristi della Free Software Foundation. La figura 1 rappresenta Stallman fotografato ad una conferenza negli abiti e con l'aureola (la superficie attiva di un hard disk della prima generazione) di Saint I-GNU-tius. Gli anatemi, che un tempo erano rivolti a Microsoft, erano diretti quel giorno ai fedifraghi della *Open Source Initiative*.
Ne è nata una guerra civile che tuttora imperversa nella comunità dei produttori e degli utilizzatori del software libero.

3.4. Tempo reale e automazione

Negli ultimi mesi l'importanza di Linux come strumento tecnico e come prodotto di mercato è considerevolmente aumentata. Le ultime versioni si sposano perfettamente con tutte le unità hardware più diffuse - processori, memorie, video, controllori di dischi, lettori dei nuovi dischi ottici DVD, schede di rete e altri apparati di comunicazione -, adottano interfacce grafiche avanzate, possono essere installate e configurate anche dai meno esperti. La versione server consente la cifratura dei documenti ed è aperta alle più importanti funzionalità della Rete. Secondo la IDC (*International Data Corporation*), un'importante azienda che opera nel settore delle valutazioni commerciali, i server Linux hanno raggiunto il 27% del mercato, sempre più vicini ai server Microsoft attestati al 41%.
Comunque, il mercato dei prodotti informatici classici, quelli che ammiriamo nelle banche, nelle pubbliche amministrazioni, negli uffici, è soltanto la punta emergente di un iceberg molto più grande di quanto vediamo. Una delle aree tecnicamente ed economicamente più importanti dell'"altra" informatica è quella dell'automazione del lavoro. Sono microcalcolatori o microcontrollori quelli che controllano le lavorazioni e le movimentazioni elementari; sono "personal computer" o PLC (*Controllori Logici Programmabili*) quelli che coordinano tutte le attività di un'isola di lavorazione; sono elaboratori di fascia più alta quelli che controllano tutti i flussi produttivi e il coordinamento di questi flussi con gli approvvigionamenti, i magazzini, le filiali di vendita.
Non soltanto i prodotti, ma la stessa industria pesante di oggi, ossia l'industria degli strumenti della produzione, è divenuta più leggera dell'aria leggera come il pensiero (Figura 6). Milioni di istruzioni software, dieci piani di morbidezza, a bordo dei microcontrollori, dei controllori logici programmabili, degli elaboratori di stabilimento e di azienda, sono diventati gli strumenti fondamentali di produttività, qualità, competitività.

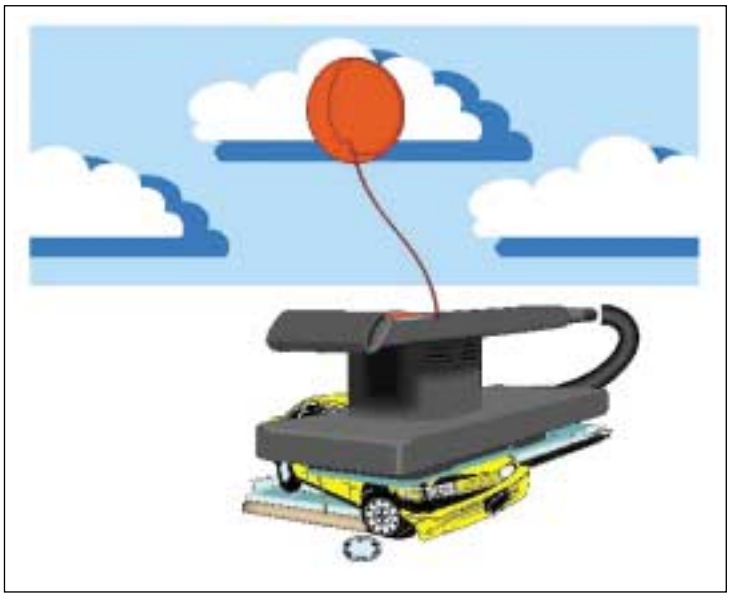

FIGURA 6
L'industria pesante, ossia l'industria degli strumenti della produzione, è più leggera dell'aria

Il nocciolo di questa montagna di software è il "sistema operativo", che deve operare in "tempo reale", ossia deve dare comandi e risposte in tempi predefiniti, prevedibili e relativamente brevi. Un segnale d'allarme che provenga dal sensore di pressione di una caldaia deve poter essere elaborato in un tempo così breve da consentire l'apertura della valvola di sicurezza prima dello scoppio della caldaia.
Sino a pochi mesi fa tutto il software di automazione e, in particolare, il sistema operativo in tempo reale erano prodotti da aziende specializzate che vendevano software "proprietario", generalmente su commessa. Il loro compito era diventato sempre più complesso e difficile, perché il ritmo con cui nuovi componenti hardware venivano introdotti sul mercato e la rapidità di obsolescenza dei vecchi componenti richiedevano investimenti sempre più onerosi. Per questa ragione, molti operatori del settore stanno puntando sul software libero e su Linux in particolare.
Linux offre potenti strumenti per la gestione del sistema, una ricca dote di programmi di

servizio per la gestione delle unità periferiche, affidabilità e robustezza. Soprattutto, a differenza di Windows, Linux è intrinsecamente modulare, per cui può essere variamente configurato in funzione delle specifiche esigenze. La disponibilità del codice sorgente e la trasparenza che ne deriva consentono di ritagliare i singoli moduli in modo non soltanto da attuare la configurazione ottimale, ma anche da consentire una modellazione matematica del suo comportamento e verificare la rispondenza alle specifiche esigenze del tempo reale.

3.5. Il Software libero "incastonato"

Più del 95% dei microprocessori che sono prodotti ogni anno sono destinati non all'industria dei calcolatori e degli apparati per le telecomunicazioni, ma agli altri comparti produttivi. Qualunque oggetto di produzione recente si prenda in mano, dal giocattolino per i bambini alla bilancia della cucina, contiene, nascosto al suo interno, almeno un microprocessore e molto software.
Secondo i calcoli di qualche esperto di valutazioni finanziarie, la dimensione economica di tutto questo software "incastonato", come chi scrive ama tradurre la denominazione inglese "*embedded*", è dell'ordine di mille miliardi di Euro ogni anno.
Proprio questa enorme dimensione economica sta inducendo migliaia di aziende manifatturiere di tutto il mondo a introdurre il software libero nei propri prodotti. In questo comparto, Linux sta giocando la parte del leone, in virtù dei suoi pregi di affidabilità, qualità, modularità, configurabilità e, soprattutto, basso costo. Sono le stesse ragioni che stanno determinando il successo di questo gioiello del software libero nell'area delle applicazioni in tempo reale cui è stato dedicato il paragrafo precedente, e il legame fra le due aree applicative diviene ogni giorno più forte. Infatti, molti sistemi operanti in tempo reale appartengono alla categoria dell'informatica "incastonata" e la maggioranza dei prodotti di questa categoria deve generalmente operare in tempo reale.

3.6. Software libero e Telecomunicazioni

Attualmente, il mondo delle tecnologie e dei prodotti per le telecomunicazioni è diviso in due emisferi. Il primo è costituito dall'antico continente che ha il suo baricentro nella terraferma della tradizionale telefonia analogica. Questo emisfero è dominato da soluzioni antiche, basate su tecnologie e software prevalentemente proprietari. Il secondo emisfero è costituito dal grande oceano di Internet, dominato da tecnologie e software liberi, su cui galleggiano piccoli arcipelaghi di prodotti proprietari - tipicamente quelli della telefonia cellulare - emersi frettolosamente negli ultimi anni.
Gli studiosi dell'ambiente ci hanno avvisato che il livello degli oceani sta salendo e che molte delle terre emerse rischiano di essere inghiottite dalle acque. Vi sono chiari segni che lo stesso fenomeno si sta manifestando nel mondo delle tecnologie e dei prodotti per le telecomunicazioni, con l'oceano del software libero che sta iniziando a sommergere le terre ferme delle soluzioni proprietarie.
Motorola, l'azienda che domina il settore della telefonia mobile ove il software è sempre stato proprietario, ha dichiarato, ad esempio, l'intenzione di liberalizzare il proprio software e, come primo passo, ha firmato un accordo di collaborazione con Collabnet, che progetta e costruisce strumenti per sviluppi cooperativi basati su software libero. Nell'ambito di questa collaborazione, Collabnet pone un sito e una linea di strumenti specifici a disposizione della comunità dei programmatori intenzionati a collaborare al progetto. Motorola fornisce, invece, i propri strumenti e il proprio ambiente di sviluppo per realizzare programmi applicativi per una nuova linea di apparati di comunicazione e elaborazione, chiamati iDEN (I*ntegrated Digital Enhanced Network*), che saranno presto adottati da alcuni operatori negli Stati Uniti e in Canada.

4. UN NUOVO MODELLO DELLO SVILUPPO?

4.1. Modelli di business del software libero e di quello proprietario

I sostenitori del software libero sognano un futuro in cui la maggioranza delle imprese del settore operi esclusivamente sul software libero. È realistico questo sogno? Come potranno queste imprese produrre fatturati ed utili adeguati?
L'argomento è stato ed è ancora oggetto di un ampio dibattito in ambito economico.

Chi scrive ritiene che gli studi economici dedicati ai modelli di business dei produttori di software libero siano sostanzialmente corretti, ma abbiano il difetto di non porre in adeguata evidenza il seguente rilievo. Oggi, la grande maggioranza degli operatori del software lavora su commessa, pagati sostanzialmente dal cliente in funzione del tempo dedicato ad ogni specifica attività. I loro modelli di business consistono nello sviluppo di software specifico per l'applicazione del cliente, nell'installazione di hardware e programmi, nella gestione di sistemi informativi, nella produzione di manuali, nella formazione. Soltanto una piccola minoranza degli addetti al software, lo produce per venderlo poi su licenza, sulla base di un valore ben definito per ogni copia venduta o per ogni installazione prevista, secondo il modello economico delle note multinazionali del settore. In sostanza, i modelli di business del software libero coincidono con i modelli adottati dalla grande maggioranza degli operatori del settore del software, compresi quelli che non credono nel software libero e utilizzano esclusivamente il software proprietario.

4.2. La ragione più importante della convenienza del software libero

Chi scrive è convinto che l'importanza del software libero vada al di là della pur rilevante dimensione economica del comparto industriale associato; vede, inoltre, nel software libero il simbolo e la sostanza di una nuova rivoluzione tecnologica ed industriale, che è andata maturando nell'arco di un ventennio e che è letteralmente esplosa negli ultimi due anni.
In sintesi, tre sono le caratteristiche salienti del nuovo mondo. In primo luogo, i nuovi prodotti hanno un prevalente contenuto di conoscenza teorica, indipendentemente dalla struttura fisica del supporto. In particolare, il software è conoscenza pura.
In secondo luogo, il volume di conoscenza antica o recente incorporata in quei prodotti è enorme, difficilmente realizzabile "ex-novo" da una sola impresa, anche se dotata di imponenti strutture di ricerca e sviluppo.
In terzo luogo, l'intreccio delle conoscenze incorporate in qualunque prodotto è così stretto e complesso, e la varietà di queste conoscenze è così ampia, da aggiungere un'ulteriore difficoltà alla loro ricostruzione nell'ambito di una sola azienda, essendo oggi ogni impresa caratterizzata da una forte specializzazione in una determinata area.
Il nuovo scenario ingloba immagini molto luminose che inducono all'ottimismo. I modelli di business del software libero, più simili a quelli dell'artigianato che non a quelli della grande industria, potrebbero rendere meno iniqua la competizione fra imprese piccole e grandi, fra Paesi poveri e ricchi. La complessità dell'universo delle conoscenze, potrebbe rendere la collaborazione più conveniente della competizione. La solidarietà potrebbe manifestarsi come una nuova, rivoluzionaria, opportunità.

Bibliografia

[1] Berra M, Meo AR: *Informatica solidale*. Bollati Boringhieri 2001.

[2] Chalmers R: *The unknown hackers. Bill and Lynne Jolitz may be the most famous programmers you've never heard of*. http://www.salon.com/tech/feature/2000/05/17/386bsd

[3] Chiccarelli S, Monti A: *Spaghetti hacker: storie, tecniche e aspetti giuridici dell'hacking in Italia*. Apogeo, Milano 1997.

[4] Di Bona C, Ockman S, Stone M: (a cura di), *Open Source: Voices from the Open Source Revolution. O'Reilly* Press., Cambridge 1999.

[5] Linuxjournal: http://www.linuxjournal.com.

[6] Mastrolilli P: *Hackers: i ribelli digitali*. GLF Laterza, Bari 2001.

[7] OpenBSD: *Three years without a remote hole in the default install!* - http://www.openbsd.org/.

[8] Raymond E: *The Catedral and the Bazaar*. Paperback Edition, 2001.

[9] Slashdot: http://www.slashdot.org.

Meo Angelo Raffaele Dal 1970 è docente di Reti di Calcolatori presso il Politecnico di Torino.
Dal 1970 al 1999 ha diretto il Centro per l'Elaborazione Numerale dei Segnali del C.N.R.
Dal 1979 al 1985 ha diretto il Progetto Finalizzato Informatica.
Dal 1991 al 1996 ha diretto il Centro di Supercalcolo del Piemonte.
È autore di oltre cento pubblicazioni scientifiche su riviste nazionali e internazionali.
È socio nazionale dell'Accademia delle Scienze.
e-mail: meo@polito.it