COMPENDIO DI LIBERTA' INFORMATICA E CULTURA OPEN

a cura di Simone Aliprandi

con testi di
Simone Aliprandi, Emmanuele Bello,
Marco Biagiotti, Massimo Carboni,
Donato Molino, Bruce Perens,
Alessandro Rubini, Richard M. Stallman

AVVISO PER
COPISTERIE E CENTRI DI STAMPA

Questo libro si può fotocopiare
e riprodurre integralmente
senza violare alcuna norma
sul copyright e senza dover
corrispondere nulla alla SIAE.

AVVISO PER I SINGOLI UTENTI

Nonostante il particolare regime
di copyright di questo libro ti
consenta di riprodurlo liberamente,
considera l'ipotesi di acquistarne
una copia originale.
Il prezzo di copertina
particolarmente basso
fa sì che l'originale costi
meno di una copia artigianale.
Avresti così un oggetto
più gradevole e maneggevole
e contemporaneamente sosterresti
questo tipo di editoria.

AVVISO PER TUTTI

Se vuoi sostenere e diffondere
la cultura dell'informatica libera
e della condivisione delle conoscenze,
collabora alla distribuzione di questo libro.

Se militi in qualche associazione
o lavori per enti pubblici e centri culturali,
contattaci all'indirizzo **info@copyleft-italia.it**
per organizzare conferenze di presentazione,
ordinare copie a prezzi scontati,
segnalarci librerie o società di distribuzione.
Oppure puoi scrivere una recensione del libro
o più semplicemente fare un link dal tuo sito
a **www.copyleft-italia.it**.

Ogni occasione di visibilità è preziosa.

EDIZIONE:
PrimaOra soc. coop. a r.l. (Lodi) - www.primaora.it

STAMPA:
Grafiche Lama (Piacenza)

PROMOSSO DA
www.copyleft-italia.it

IN COLLABORAZIONE CON
Comune di Modena - Assessorato alle Politiche Giovanili
Prosa - Progettazione Sviluppo Aperto

DISCLAIMER

INDICE

** contributi completamente inediti*

Presentazione

Se qualcuno si aspetta qualcosa di particolarmente innovativo da queste pagine, ha sbagliato libro. Lo dico subito, onde evitare equivoci.

Questa è un'antologia di articoli e documenti già pubblicati altrove; e i pochi testi inediti qui contenuti trattano temi già abbastanza noti ai cultori dell'informatica libera.

Eppure sono fermamente (e un po' spavaldamente) convinto della grande utilità di questo libro. Dalla fine degli anni 90 ad oggi ormai si è scritto molto su questa cultura, ma durante questi due anni di mia attività di sensibilizzazione (partecipando ad eventi, progetti e mailing list) e ricerca privata in questo settore, mi sono accorto di quanto ci sia bisogno in Italia di **informazione di base** su certi temi. La gente che muove i primi passi in questo nuovo affascinante mondo della libera condivisione delle conoscenze e della creatività è molto curiosa di sapere da dove è partito il tutto; e purtroppo le occasioni di divulgazione sono davvero poche e spesso rivolte a pubblici settoriali e ristretti (principalmente informatici, o economisti e giuristi).

Questo libro cerca di fare un primo passo per rispondere a questa esigenza diffusa che statisticamente è proiettata a crescere sempre di più nei prossimi anni. Ho voluto raccogliere e commentare i testi che a mio avviso fanno luce meglio di altri sui concetti fondamentali; concetti che infatti danno il nome alle varie sezioni di questo libro (Free Software, Open Source, Linux, Copyleft, Opencontent) e che ho già cercato di approfondire nel mio primo libro "Copyleft & opencontent". Qui ho cercato di fare un passo indietro, ma

con un'ottica più matura e con una maggiore consapevolezza del fenomeno in questione; ho cercato di rispolverare le letture che ho fatto in questi anni e di interrogarmi con il "senno di poi" sul loro impatto. Le mie considerazioni sono contenute di volta in volta nelle presentazioni introduttive che precedono ogni articolo e ogni documento: l'augurio è che possano risultare utili.

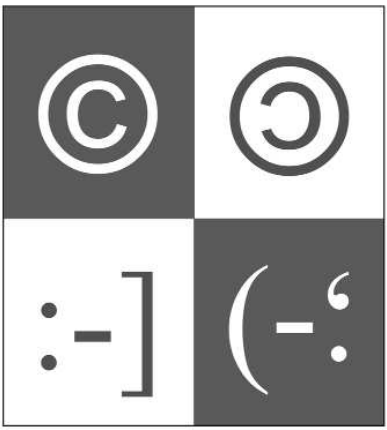
Inoltre tengo a precisare (ancora in modo vergognosamente spavaldo) che questo libro incarna al meglio lo spirito dell'opencontent. Grazie a questa filosofia di distribuzione delle opere dell'ingegno che altri illustri autori prima di me hanno voluto sposare, mi è stato possibile realizzare quest'alchimia di voci autorevoli senza dover muovere un dito di troppo per l'acquisizione dei diritti. Spero che ciò possa fungere da esempio concreto (una sorta di urlo alla Frankenstein Junior: "si può fare!) e quindi possa aprire la strada in Italia ad un tipo di editoria illuminata e innovativa, come quella che da due anni a questa parte cerco di sostenere direttamente.

Ovviamente devo ringraziare tutti coloro che mi hanno sostenuto nel perseguimento di questo obbiettivo: gli enti che hanno sponsorizzato e promosso le mie prime pubblicazioni, senza i quali non avrei mai potuto raggiungere un certo grado di visibilità; i docenti e ricercatori che mi hanno coinvolto in convegni e workshop, permettendomi di misurarmi sempre con pubblici diversi e quindi di sondare le diverse esigenze d'informazione; gli amici della società PrimaOra, che ancora una volta mi hanno consentito di gestire l'iniziativa editoriale con la massima libertà; ma soprattutto gli enti promotori di questa nuova pubblicazione: è infatti grazie al loro contributo che il libro può essere venduto ad un prezzo di copertina inferiore agli standard di mercato (tra l'altro rendendone così più conveniente l'acquisto in originale rispetto alla stampa artigianale, comunque consentita dalla licenza). Infine un grazie anticipato a coloro che collaboreranno alla distribuzione di quest'opera editoriale, anche in canali alternativi (associazioni, fiere, eventi) cioè al di là della tradizionale rete di vendita (librerie, edicole, megastores) che resta - per motivi spesso pretestuosi - ancora meta difficilmente raggiungibile per questo tipo di editoria.

A questo punto, buona lettura e... buona condivisione!

Simone Aliprandi
(gennaio 2006)

Parte prima
PER FARSI UN’IDEA...



Nella pagina precedente:
Il copyleft: l'altra faccia del copyright (di Simone Aliprandi)

Breve introduzione al modello copyleft
(di Simone Aliprandi)

Con questo testo ho cercato di condensare in poche pagine tutti i concetti fondamentali della nuova cultura della condivisione delle conoscenze, seguendo a grandi linee la struttura tematica del mio libro "Copyleft & opencontent". Quest'opera si è sviluppata in più fasi ed è stata pubblicata già in altre occasioni, pur con alcune leggere differenze testuali.

[S. Aliprandi]

In principio era il software

Il software nasce nel secondo dopoguerra come uno strumento direttamente funzionale all'applicazione tecnologica e dunque strettamente legato all'hardware: si può dire infatti che inizialmente ogni calcolatore contenesse le istruzioni utili a farlo funzionare. Quando i calcolatori iniziarono a diventare macchine più complesse e contemporaneamente strumenti di lavoro non più elitari ma diffusi anche al di fuori dei centri di ricerca, ecco che si pose il problema di creare dei sistemi operativi standardizzati che permettessero una maggiore fruibilità anche da parte di utenti medi. Il software divenne così un'entità a sè e di lì a poco (cioè con la diffusione di massa del computer) un prodotto commerciabile: era infatti possibile acquistare un computer (nel senso di solo hardware) e in separata sede installarvi il sistema operativo e gli applicativi che servivano all'utente (software).

Nacque così la nuova esigenza di "pacchettizzare" il software e metterlo a disposizione degli utenti in una rete di distribuzone a sè stante; e fu in questa fase che le imprese avventuratesi in questo nuovo mercato iniziarono ad avvertire l'esigenza di tutelare il proprio lavoro, servendosi degli strumenti classici che il diritto industriale metteva a disposizione: il segreto industriale, il brevetto o il diritto d'autore. Per una serie di motivi e di considerazioni dottrinali su cui non esiste tuttora un consenso unanime (e su cui non è il caso di soffermarsi in questa sede), la scelta cadde sulla tutela d'autore, chiamata "copyright" nel suo contesto originario, ovvero quello statunitense: fu infatti il legislatore americano (con il Software Copyright Act del 1980) a fare il primo passo ufficiale in questa direzione, per essere poi seguito nel decennio successivo dai legislatori degli altri paesi industrializzati.

Copyleft in ambito software

Tuttavia, un gruppo sempre più folto di informatici di vecchio stampo (i cosiddetti "hacker" nel senso però neutrale del termine, cioè di appassionati della libera programmazione), quelli per cui il software doveva rimanere uno strumento di sviluppo tecnologico più che un oggetto di marketing, opposero resistenza a questo trend ispirato ad un'ottica unicamente di profitto, volendo dimostrare al mondo che il software poteva (anzi, doveva) rimanere uno strumento il più possibile libero da vincoli giuridici e fruibile da chiunque volesse intervenire sulla sua struttura e sulle sue funzionalità.

A questo scopo era fondamentale poter sempre disporre del codice sorgente, ovvero il codice in linguaggio di programmazione grazie al quale è possibile capire la struttura del software ed eventualmente modificarlo e correggerlo. Le imprese di software invece, sfruttando i diritti esclusivi del regime di copyright, distribuivano il software solo in linguaggio macchina (codice binario), criptando il codice sorgente e rendendo così ogni operazione di modifica, aggiornamento e adattamento impossibile o quantomeno difficile.

Essendo il software ormai un'opera sottoposta a copyright ed essendo il copyright un regime di tutela che si applica automaticamente con la creazione dell'opera, gli sviluppatori che avessero voluto distribuire le loro opere liberamente avevano solo due scelte: rilasciarle esplicitamente in un regime di public domain (ma questa scelta sarebbe stata controproducente poichè chiunque avrebbe potuto appropriarsi dell'opera e sfruttarla anche a fini commerciali, per di più criptando successivamente il codice sorgente); oppure rilasciarle sotto un particolare regime giuridico cristallizzato in una licenza nella quale l'autore, sempre fondandosi sui principi del copyright,

disciplinava le modalità di utilizzo e distribuzione dell'opera.

Tale particolare regime, nato nell'ambito del progetto GNU, prese il nome emblematico di copyleft e i suoi principi cardine furono condensati in un'apposita licenza chiamata GNU General public license (GPL): essa garantiva che il software fosse liberamente eseguibile, copiabile e modificabile, e soprattutto che chiunque ridistribuisse copie di quel software o creasse altro software derivato da quel codice mantenesse il medesimo regime di licenza. Uno scaltro escamotage che garantiva la persistenza all'infinito delle libertà caratteristiche del software libero.

Dispute terminologiche: "copyleft"

L'espressione "copyleft" nasce dalla prassi goliardica di alcuni sviluppatori di software che distribuivano copie dei loro lavori riportanti la dicitura "copyleft - all rights reversed" (con una © rovesciata). In effetti il termine è molto significativo poichè racchiude un duplice gioco di parole: "left" è appunto il participio passato di "leave" (lasciare, permettere) e comunica l'idea di un regime più libero; ma è anche l'opposto di "right" (destra) e comunica un'idea di ribaltamento dei principi.

Dopo il 1989 (anno di nascita della GPL) comparvero altre licenze ispirate alla stessa filosofia, ma gli ideologi/porta-voce del progetto GNU si preoccuparono di dare al termine copyleft una configurazione piuttosto netta: ovvero nell'accezione originaria è considerata vera licenza copyleft quella che impone il mantenimento all'infinito del medesimo regime (nel linguaggio Creative Commons, la cosidetta clausola "share-alike", cioè "condividi allo stesso modo").

Questo nuovo modello di gestione dei diritti d'autore ha avuto fin da subito grande rilevanza socio-culturale e col tempo l'espressione "copyleft", forse per la sua particolare efficacia semantica, è stata usata per indicare più ampiamente tutto questo fenomeno giuridico di rivisitazione del modello tradizionale di gestione dei diritti d'autore. E nonostante le critiche dei puristi del movimento, questo allargamento semantico è ormai un dato di fatto in gran parte della documentazione e saggistica in materia.

Dispute terminologiche: "free software o open source?"

Dato che in inglese l'aggettivo "free" significa contemporaneamente "libero" e "gratuito", è spesso passato l'equivoco che software libero fosse tutto ciò che veniva regalato. Ma a questa stregua sarebbero rientrati in tale categoria anche i software "trial-version" oppure "freeware", distribuiti a scopi puramente commerciali e comunque senza disponibilità del codice

sorgente. Tale confusione era assolutamente da evitare. Inoltre la diffusione di questo messaggio da un lato sviliva il software libero che appariva come il "fratello povero" del software proprietario (quando invece si trattava il più delle volte di software di grande pregio e affidabilità); e d'altro canto incuteva un timore di fondo nei confronti di alcune imprese che avrebbero voluto investire risorse economiche anche nello sviluppo di software libero.

Fu così che nel 1998 alcuni attivisti del settore cercarono di dare un nuovo volto al fenomeno rendendolo in un certo senso più appetibile al mondo imprenditoriale. L'idea era quella di puntare non più tanto su aspetti etici di libertà e condivisione quanto piuttosto sulle caratteristiche e i vantaggi tecnici di questo tipo di software. Nacque dunque il termine indubbiamente efficace "open source" (cioè "codice sorgente aperto") e la Open Source Initiative, un progetto guidato da Eric Raymond che si sarebbe occupato di vigilare sul corretto uso di questo termine e dunque di verificare che le varie licenze emerse in quegli anni mantenessero alcuni parametri di base.

Si creava così una dicotomia, tuttora insanata, fra conservatori (fedeli al paradigma originario voluto dalla Free Software Foundation) e innovatori (aperti alle nuove prospettive di marketing). Una divisione spesso puramente teorica e basata su argomentazioni etico-filosofiche, dato che nella maggior parte dei casi il software "open source" è anche "free software" e ad ogni modo si tratta di due mondi paralleli che tra l'altro di dirigono nella stessa direzione.

Copyleft in ambito content: la documentazione tecnico-informatica

Con la diffusione del software libero e del software open source anche in un circuito commerciale e di massa, ci si è spesso trovati di fronte ad un paradosso: tutta la documentazione (istruzioni tecniche, manuali, presentazioni) relativa al software libero e prodotta dagli stessi sviluppatori, veniva editata in un regime di copyright tradizionale. Molti autori, soprattutto i "guru" del movimento (primo fra tutti Richard Stallman) pubblicavano i loro articoli d'informazione e sensibilizzazione accompagnati da una breve nota di copyleft che suonava più o meno così: "è permessa la copia letterale dell'opera con ogni mezzo a condizione che venga riportata questa nota". In questo laconico disclaimer si condensa in effetti molto efficacemente il senso pratico del modello copyleft persistente; dal punto di vista giuridico però tale laconicità poteva essere foriera di abusi e interpretazioni fuorvianti. Tra l'altro l'uso di questa nota nel caso di documentazione poteva non essere particolarmente appropriato poichè non si contemplava la possibilità di modifica dei contenuti dell'opera: possibilità determinante trattandosi

di manuali di software liberamente modificabile, oltre che liberamente copiabile. Alcuni autori scelsero di applicare la GPL anche alle opere di documentazione, ma come è già emerso si tratta di una licenza pensata e palesemente riferita ad un'opera tecnico-funzionale come il software. Ecco che nel 2000 nacque (sempre in seno al progetto GNU) la Free Documentation License: una licenza appositamente pensata per le opere letterarie, dunque una delle prime licenze copyleft in ambito content e non solo strettamente software.

Copyleft in ambito content: le opere artistico-espressive in generale

Sulla scia di questo nuovo spiraglio apertosi in ambito informatico e più in generale della diffusione massiccia di Internet, in quegli anni (cioè dalla fine degli anni Novanta) si attivarono alcuni progetti di promozione della libera circolazione delle informazioni e delle opere creative. Ogni progetto propose la propria "ricetta" per sdoganare i principi del copyleft anche in quell'ambito non più strettamente informatico: nacquero così alcune licenze come - per citarne solo alcune - la Open Publication License (del progetto OpenContent), la OpenAudio License (della Electronic Frontier Foundation), la OpenMusic License (del progetto tedesco OpenMusic), Licence Art Libre (del progetto francese Art Libre). Fu però un gruppo di giuristi di Stanford (capitanati dal professor Lawrence Lessig) a fare il passo più determinante in questo senso, con l'attivazione del progetto Creative Commons e la diffusione nel 2002 delle relative licenze: queste licenze erano pensate in modo da poter funzionare per tutti i tipi di opere creative e in modo da poter essere tradotte e possibilmente adattate ai vari ordinamenti giuridici. Tra l'altro la loro struttura si articolava in clausole modulari che permettevano all'autore di decidere quali usi consentire per la sua opera, a quali condizioni e in quali contesti: in poche parole, consentivano all'autore di graduare la libertà di utilizzo dell'opera, chiarendone le condizioni.

Il senso del copyleft in sintesi.

Cercando di dare una definizione semplice e chiara al concetto di copyleft, possiamo dire che si tratta di un modello alternativo di gestione dei diritti d'autore, che opera - a differenza del modello tradizionale - in un'ottica non esclusiva e non standardizzata e che deriva originariamente dalla libera scelta dell'autore. Esso si realizza in concreto grazie all'applicazione di alcuni contratti-licenza che disciplinano la diffusione dell'opera e chiariscono a quali condizioni essa può essere condivisa, modificata, commercializzata.

I principali effetti di tale modello sono:

- *disintermediazione,* nel senso che è l'autore stesso a decidere a priori alcune regole relative alla diffusione della sua opera e tali regole sono rivolte a tutta la comunità degli utenti, senza più necessità di un soggetto imprenditoriale che si occupi della distribuzione e commercializzazione dell'opera (editore, produttore, etc.);

- *riequilibrio*, nel senso che, qualora ci fosse comunque l'interazione di un soggetto imprenditoriale (cosa spesso auspicabile), gli equilibri contrattuali relativi ai diritti sull'opera verrebbero ampiamente ridefiniti, spostandosi maggiori prerogative nella sfera dell'autore e creandosi maggiori libertà nella sfera dell'utente finale;

- *elasticità e differenziazione*, nel senso che in questo modello ogni opera ha un suo specifico regime giuridico e tale regime può strutturarsi in modo decisamente più elastico e dinamico, così da adattarsi meglio alla nuova (e in continua evoluzione) compagine del mondo della comunicazione multimediale;

- *sostenibilità economica*: grazie a queste sue caratteristiche, tale paradigma riesce a realizzare un ideale di modello economico sostenibile, tanto nel mondo informatico (ne è la prova da più di un decennio il successo anche economico del software open source) quanto nel mondo della produzione intellettuale in generale.

Parte seconda
FREE SOFTWARE

Nella pagina precedente:
La testa di gnu, simbolo-mascotte del Progetto GNU

Il Progetto GNU
(di Richard M. Stallman)

Richard M. Stallman è riconosciuto unanimemente come il pioniere della cultura della libera condivisione delle conoscenze e come il primo porta-voce della comunità di sviluppo di software libero. Per primo ha avuto la geniale intuizione di creare un nuovo modello di gestione dei diritti d'autore che - per suo battesimo - va sotto il nome di copyleft. Artefice del Progetto GNU e fondatore (nonchè attuale presidente) della Free Software Foundation, ama definirsi un vero e proprio "hacker" (nel senso buono e corretto del termine); ma oltre ad essere leader e informatico eccellente, è anche autore di saggi davvero efficaci e interessanti, tutti rilasciati con un permesso di copia letterale. Quello che ho scelto di riportare è probabilmente il più completo, quello in cui Stallman riesce a presentare tutte le tematiche a lui care: un vero e proprio manifesto del Progetto GNU, ma anche un'utile spiegazione delle motivazioni che stanno dietro quella particolare filosofia. [S. Aliprandi]

La prima comunità di condivisione del software

Quando cominciai a lavorare nel laboratorio di Intelligenza Artificiale del MIT nel 1971, entrai a far parte di una comunità in cui ci si scambiavano i programmi, che esisteva già da molti anni. La condivisione del software non si limitava alla nostra comunità; è un cosa vecchia quanto i compu-

ter, proprio come condividere le ricette è antico come il cucinare. Ma noi lo facevamo più di quasi chiunque altro.

Il laboratorio di Intelligenza Artificiale (AI) usava un sistema operativo a partizione di tempo (timesharing) chiamato ITS (Incompatible Timesharing System) che il gruppo di hacker1 del laboratorio aveva progettato e scritto in linguaggio assembler per il Digital PDP-10, uno dei grossi elaboratori di quel periodo. Come membro di questa comunità, hacker di sistema nel gruppo laboratorio, il mio compito era migliorare questo sistema.

Non chiamavamo il nostro software "software libero", poiché questa espressione ancora non esisteva, ma si trattava proprio di questo. Quando persone di altre università o di qualche società volevano convertire il nostro programma per il proprio sistema e utilizzarlo, erano le benvenute. Se si vedeva qualcuno usare un programma sconosciuto e interessante, si poteva sempre chiedere di vederne il codice sorgente, in modo da poterlo leggere, modificare, o prenderne cannibalizzarne alcune parti per creare un nuovo programma.

La comunità si dissolve

La situazione cambiò drasticamente all'inizio degli anni '80 quando la Digital smise di produrre la serie PDP-10. La sua architettura, elegante e potente negli anni '60, non poteva essere estesa in modo naturale ai più grandi spazi di indirizzamento che si stavano rendendo possibili negli anni '80. Questo significò che quasi tutti i programmi che formavano ITS divennero obsoleti.

La comunità di hacker del laboratorio di Intelligenza Artificiale si era già dissolta non molto tempo prima. Nel 1981 la Symbolics, nata da una costola del laboratorio stesso, gli aveva sottratto quasi tutti gli hacker; l'ormai esiguo gruppo rimasto fu dunque incapace di sostenersi (il libro "Hackers" di Steve Levy narra questi eventi, oltre a fornire una fedele ricostruzione di questa comunità ai suoi inizi). Quando il laboratorio di Intelligenza Artificiale nel 1982 acquistò un nuovo PDP-10, i sistemisti decisero di utilizzare il sistema timesharing non libero della Digital anziché ITS.

I moderni elaboratori di quell'epoca, come il VAX o il 68020, avevano il proprio sistema operativo, ma nessuno di questi era libero: si doveva firmare un accordo di non-diffusione persino per ottenerne una copia eseguibile.

Questo significava che il primo passo per usare un computer era promettere di negare aiuto al proprio vicino. Una comunità cooperante era vietata. La regola creata dai proprietari di software proprietario era: "se

condividi il software col tuo vicino sei un pirata. Se vuoi modifiche, pregaci di farle".

L'idea che la concezione sociale di software proprietario - cioè il sistema che impone che il software non possa essere condiviso o modificato - sia antisociale, contraria all'etica, semplicemente sbagliata, può apparire sorprendente a qualche lettore. Ma che altro possiamo dire di un sistema che si basa sul dividere utenti e lasciarli senza aiuto? Quei lettori che trovano sorprendente l'idea possono aver data per scontata la concezione sociale di software proprietario, o averla giudicata utilizzando lo stesso metro suggerito dal mercato del software proprietario. I produttori di software hanno lavorato a lungo e attivamente per diffondere la convinzione che c'è un solo modo di vedere la cosa.

Quando i produttori di software parlano di "difendere" i propri "diritti" o di "fermare la pirateria", quello che dicono è in realtà secondario. Il vero messaggio in quelle affermazioni sta nelle assunzioni inespresse, che essi danno per scontate; vogliono che siano accettate acriticamente. Esaminiamole, dunque.

Una prima assunzione è che le aziende produttrici di software abbiano il diritto naturale indiscutibile di proprietà sul software, e di conseguenza, abbiano controllo su tutti i suoi utenti. Se questo fosse un diritto naturale, non potremmo sollevare obiezioni, indipendentemente dal danno che possa recare ad altri. È interessante notare che, negli Stati Uniti, sia la costituzione che la giurisprudenza rifiutano questa posizione: il diritto d'autore non è un diritto naturale, ma un monopolio imposto dal governo che limita il diritto naturale degli utenti a effettuare delle copie.

Un'altra assunzione inespressa è che la sola cosa importante del software sia il lavoro che consente di fare - vale a dire che noi utenti non dobbiamo preoccuparci del tipo di società in cui ci è permesso vivere.

Una terza assunzione è che non avremmo software utilizzabile (o meglio, che non potremmo mai avere un programma per fare questo o quell'altro particolare lavoro) se non riconoscessimo ai produttori il controllo sugli utenti di quel programmi. Questa assunzione avrebbe potuto sembrare plausibile, prima che il movimento del software libero dimostrasse che possiamo scrivere quantità di programmi utili senza bisogno di metterci dei catenacci.

Se rifiutiamo di accettare queste assunzioni, giudicando queste questioni con comuni criteri di moralità e di buon senso dopo aver messo al primo posto gli interessi degli utenti, tenendo conto che gli utenti vengono prima di tutto, arriviamo a conclusioni del tutto differenti. Chi usa un calcolatore

dovrebbe essere libero di modificare i programmi per adattarli alle proprie necessità, ed essere libero di condividere il software, poiché aiutare gli altri è alla base della società.

Non c'è modo in questa sede di trattare approfonditamente i ragionamenti che portano a questa conclusione; il lettore interessato può cercare le informazioni in rete a questo indirizzo: http://www.gnu.org/philosophy/why-free.html.

Una difficile scelta morale

Una volta che il mio gruppo si fu sciolto, continuare come prima fu impossibile. Mi trovai di fronte a una difficile scelta morale.

La scelta facile sarebbe stata quella di unirsi al mondo del software proprietario, firmando accordi di non-diffusione e promettendo di non aiutare i miei compagni hacker. Con ogni probabilità avrei anche sviluppato software che sarebbe stato distribuito secondo accordi di non-diffusione, contribuendo così alla pressione su altri perché a loro volta tradissero i propri compagni.

In questo modo avrei potuto guadagnare, e forse mi sarei divertito a programmare. Ma sapevo che al termine della mia carriera mi sarei voltato a guardare indietro, avrei visto anni spesi a costruire muri per dividere le persone, e avrei compreso di aver contribuito a rendere il mondo peggiore.

Avevo già sperimentato cosa significasse un accordo di non diffusione per chi lo firmava, quando qualcuno rifiutò a me e al laboratorio AI del MIT il codice sorgente del programma di controllo della nostra stampante; l'assenza di alcune funzionalità nel programma rendeva oltremodo frustrante l'uso della stampante. Per cui non mi potevo dire che gli accordi di non-diffusione fossero innocenti. Ero molto arrabbiato quando quella persone si rifiutò di condividere il programma con noi; non potevo far finta di niente e fare lo stesso con tutti gli altri.

Un'altra possibile scelta, semplice ma spiacevole, sarebbe stata quella di abbandonare l'informatica. In tal modo le mie capacità non sarebbero state mal utilizzate, tuttavia sarebbero state sprecate. Non sarei mai stato colpevole di dividere o imporre restrizioni agli utenti di calcolatori, ma queste cose sarebbero comunque successe.

Allora cercai un modo in cui un programmatore potesse fare qualcosa di buono. Mi chiesi dunque: c'erano un programma o dei programmi che io potessi scrivere, per rendere nuovamente possibile l'esistenza di una comunità?

La risposta era semplice: innanzitutto serviva un sistema operativo.

Questo è difatti il software fondamentale per iniziare a usare un computer. Con un sistema operativo si possono fare molte cose; senza, non è proprio possibile far funzionare il computer. Con un sistema operativo libero, avremmo potuto avere nuovamente una comunità in cui hacker possono cooperare, e invitare chiunque a unirsi al gruppo. E chiunque sarebbe stato in grado di usare un calcolatore, senza dover cospirare fin dall'inizio per sottrarre qualcosa ai propri amici.

Essendo un programmatore di sistemi, possedevo le competenze adeguate per questo lavoro. Così, anche se non davo il successo per scontato, mi resi conto di essere la persona giusta per farlo. Scelsi di rendere il sistema compatibile con Unix, in modo che fosse portabile, e che gli utenti Unix potessero passare facilmente a esso. Il nome GNU fu scelto secondo una tradizione hacker, come acronimo ricorsivo che significa "GNU's Not Unix" (GNU non è Unix).

Un sistema operativo non si limita solo al suo nucleo, che è proprio il minimo per eseguire altri programmi. Negli anni '70, qualsiasi sistema operativo degno di questo nome includeva interpreti di comandi, assemblatori, compilatori, interpreti di linguaggi, debugger, editor di testo, programmi per la posta e molto altro. ITS li aveva, Multics li aveva, VMS li aveva e Unix li aveva. Anche il sistema operativo GNU li avrebbe avuti.

Tempo dopo venni a conoscenza di questa massima, attribuita a Hillel2:

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?

La decisione di iniziare il progetto GNU si basò su uno spirito simile.

"Free" come libero

Il termine "free software" (N.d.T. il termine free in inglese significa sia gratuito che libero) a volte è mal interpretato: non ha niente a che vedere col prezzo del software; si tratta di libertà. Ecco, dunque, la definizione di software libero: un programma è software libero per un dato utente se:

· l'utente ha la libertà di eseguire il programma per qualsiasi scopo;

· l'utente ha la libertà di modificare il programma secondo i propri bisogni (perché questa libertà abbia qualche effetto in pratica, è necessario avere accesso al codice sorgente del programma, poiché apportare modifiche a un programma senza disporre del codice sorgente è estremamente difficile);

· l'utente ha la libertà di distribuire copie del programma, gratuitamente o dietro compenso;

· l'utente ha la libertà di distribuire versioni modificate del programma,

così che la comunità possa fruire dei miglioramenti apportati.

Poiché "free" si riferisce alla libertà e non al prezzo, vendere copie di un programma non contraddice il concetto di software libero. In effetti, la libertà di vendere copie di programmi è essenziale: raccolte di software libero vendute su CD-ROM sono importanti per la comunità, e la loro vendita è un modo di raccogliere fondi importante per lo sviluppo del software libero. Di conseguenza, un programma che non può essere liberamente incluso in tali raccolte non è software libero.

A causa dell'ambiguità del termine "free", si è cercata a lungo un'alternativa, ma nessuno ne ha trovata una valida. La lingua inglese ha, più termini e sfumature di ogni altra, ma non ha una parola semplice e non ambigua che significhi libero; "unfettered" è la parola più vicina come significato (N.d.T. unfettered è una parola di tono aulico o arcaico che significa libero da ceppi, vincoli o inibizioni). Alternative come "liberated", "freedom" e "open" hanno altri significati o non sono adatte per altri motivi (N.d.T. rispettivamente, liberato, libertà, aperto).

Software GNU e il sistema GNU

Sviluppare un intero sistema è un progetto considerevole. Per raggiungere l'obiettivo decisi di adattare e usare parti di software libero tutte le volte che fosse possibile. Per esempio, decisi fin dall'inizio di usare TeX come il principale programma di formattazione di testo; qualche anno più tardi, decisi di usare l'X Window System piuttosto che scrivere un altro sistema a finestre per GNU.

A causa di questa decisione, il sistema GNU e la raccolta di tutto il software GNU non sono la stessa cosa. Il sistema GNU comprende programmi che non sono GNU, sviluppati da altre persone o gruppi di progetto per i propri scopi, ma che possiamo usare in quanto software libero.

L'inizio del progetto

Nel gennaio 1984 lasciai il mio posto al MIT e cominciai a scrivere software GNU. Dovetti lasciare il MIT, per evitare che potesse interferire con la distribuzione di GNU come software libero. Se fossi rimasto, il MIT avrebbe potuto rivendicare la proprietà del lavoro, e avrebbe potuto imporre i propri termini di distribuzione, o anche farne un pacchetto proprietario. Non avevo alcuna intenzione di fare tanto lavoro solo per vederlo reso inutilizzabile per il suo scopo originario: creare una nuova comunità di condivisione di software. A ogni buon conto, il professor Winston - allora responsabile del laboratorio AI del MIT - mi propose gentilmente di conti-

nuare a utilizzare le attrezzature del laboratorio stesso.

I primi passi

Poco dopo aver iniziato il progetto GNU, venni a sapere del Free University Compiler Kit, noto anche come VUCK (la parola olandese che sta per "free" inizia con la V). Era un compilatore progettato per trattare più linguaggi, fra cui C e Pascal, e per generare codice binario per diverse architetture. Scrissi al suo autore chiedendo se GNU avesse potuto usarlo. Rispose in modo canzonatorio, dicendo che l'università era sì libera, ma non il compilatore. Decisi allora che il mio primo programma per il progetto GNU sarebbe stato un compilatore multilinguaggio e multipiattaforma.

Sperando di evitare di dover scrivere da me l'intero compilatore, ottenni il codice sorgente del Pastel, un compilatore multipiattaforma sviluppato ai Laboratori Lawrence Livermore. Il linguaggio supportato da Pastel, in cui il Pastel stesso era scritto, era una versione estesa del Pascal, pensata come linguaggio di programmazione di sistemi. Io vi aggiunsi un frontend per il C, e cominciai il porting per il processore Motorola 68000, ma fui costretto a rinunciare quando scoprii che il compilatore richiedeva diversi megabyte di memoria sullo stack, mentre il sistema Unix disponibile per il processore 68000 ne permetteva solo 64K.

Mi resi conto allora che il compilatore Pastel interpretava tutto il file di ingresso creandone un albero sintattico, convertiva questo in una catena di "istruzioni", e quindi generava l'intero file di uscita senza mai liberare memoria. A questo punto, conclusi che avrei dovuto scrivere un nuovo compilatore da zero. Quel nuovo compilatore è ora noto come Gcc; non utilizza niente del compilatore Pastel, ma riuscii ad adattare e riutilizzare il frontend per il C che avevo scritto. Questo però avvenne qualche anno dopo; prima, lavorai su GNU Emacs.

GNU Emacs

Cominciai a lavorare su GNU Emacs nel settembre 1984, e all'inizio del 1985 cominciava a essere utilizzabile. Così potei iniziare a usare sistemi Unix per scrivere; fino ad allora, avevo scritto sempre su altri tipi di macchine, non avendo nessun interesse a imparare vi né ed.

A questo punto alcuni cominciarono a voler usare GNU Emacs, il che pose il problema di come distribuirlo. Naturalmente lo misi sul server ftp anonimo del computer che usavo al MIT (questo computer, prep.ai.mit.edu, divenne così il sito ftp primario di distribuzione di GNU; quando alcuni anni dopo andò fuori servizio, trasferimmo il nome sul nostro nuovo ftp

server). Ma allora molte delle persone interessate non erano su Internet e non potevano ottenere una copia via ftp, così mi si pose il problema di cosa dir loro.

Avrei potuto dire: "trova un amico che è in rete disposto a farti una copia". Oppure avrei potuto fare quel che feci con l'originario Emacs su PDP-10, e cioè dir loro: "spediscimi una busta affrancata e un nastro, e io te lo rispedisco con sopra Emacs". Ma ero senza lavoro, e cercavo un modo di far soldi con il software libero. E così feci sapere che avrei spedito un nastro a chi lo voleva per 150 dollari. In questo modo, creai un'impresa di distribuzione di software libero, che anticipava le compagnie che oggi distribuiscono interi sistemi GNU basati su Linux.

Un programma è libero per tutti?

Se un programma è software libero quando esce dalle mani del suo autore, non significa necessariamente che sarà software libero per chiunque ne abbia una copia. Per esempio, il software di pubblico dominio (software senza copyright) è software libero, ma chiunque può farne una versione modificata proprietaria. Analogamente, molti programmi liberi sono protetti da diritto d'autore, ma vengono distribuiti con semplici licenze permissive che permettono di farne versioni modificate proprietarie.

L'esempio emblematico della questione è l'X Window System. Sviluppato al MIT, e pubblicato come software libero con una licenza permissiva, fu rapidamente adottato da diverse società informatiche. Queste aggiunsero X ai loro sistemi Unix proprietari, solo in forma binaria, e coperto dello stesso accordo di non-diffusione. Queste copie di X non erano software più libero di quanto lo fosse Unix.

Gli autori dell'X Window System non ritenevano che questo fosse un problema, anzi se lo aspettavano ed era loro intenzione che accadesse. Il loro scopo non era la libertà, ma semplicemente il "successo", definito come "avere tanti utenti". Non erano interessati che questi utenti fossero liberi, ma solo che fossero numerosi.

Questo sfociò in una situazione paradossale, in cui due modi diversi di misurare la quantità di libertà risultavano in risposte diverse alla domanda "questo programma è libero"? Giudicando sulla base della libertà offerta dai termini distributivi usati dal MIT, si sarebbe dovuto dire che X era software libero. Ma misurando la libertà dell'utente medio di X, si sarebbe dovuto dire che X era software proprietario. La maggior parte degli utenti di X usavano le versioni proprietarie fornite con i sistemi Unix, non la versione libera.

Il permesso d'autore (copyleft) e la GNU GPL

Lo scopo di GNU consisteva nell'offrire libertà agli utenti, non solo nell'ottenere ampia diffusione. Avevamo quindi bisogno di termini di distribuzione che evitassero che il software GNU fosse trasformato in software proprietario. Il metodo che usammo si chiama "permesso d'autore"3.

Il permesso d'autore (copyleft)4 usa le leggi sul diritto d'autore (copyright), ma le capovolge per ottenere lo scopo opposto: invece che un metodo per privatizzare il software, diventa infatti un mezzo per mantenerlo libero.

Il succo dell'idea di permesso d'autore consiste nel dare a chiunque il permesso di eseguire il programma, copiare il programma, modificare il programma, e distribuirne versioni modificate, ma senza dare il permesso di aggiungere restrizioni. In tal modo, le libertà essenziali che definiscono il "free software" (software libero) sono garantite a chiunque ne abbia una copia, e diventano diritti inalienabili.

Perché un permesso d'autore sia efficace, anche le versioni modificate devono essere libere. Ciò assicura che ogni lavoro basato sul nostro sia reso disponibile per la nostra comunità, se pubblicato. Quando dei programmatori professionisti lavorano su software GNU come volontari, è il permesso d'autore che impedisce ai loro datori di lavoro di dire: "non puoi distribuire quei cambiamenti, perché abbiamo intenzione di usarli per creare la nostra versione proprietaria del programma".

La clausola che i cambiamenti debbano essere liberi è essenziale se vogliamo garantire libertà a tutti gli utenti del programma. Le aziende che privatizzarono l'X Window System di solito avevano apportato qualche modifica per portare il programma sui loro sistemi e sulle loro macchine. Si trattava di modifiche piccole rispetto alla mole di X, ma non banali. Se apportare modifiche fosse una scusa per negare libertà agli utenti, sarebbe facile per chiunque approfittare di questa scusa.

Una problematica correlata è quella della combinazione di un programma libero con codice non libero. Una tale combinazione sarebbe inevitabilmente non libera; ogni libertà che manchi dalla parte non libera mancherebbe anche dall'intero programma. Permettere tali combinazioni aprirebbe non uno spiraglio, ma un buco grosso come una casa. Quindi un requisito essenziale per il permesso d'autore è tappare il buco: tutto ciò che venga aggiunto o combinato con un programma protetto da permesso d'autore dev'essere tale che il programma risultante sia anch'esso libero e protetto da permesso d'autore.

La specifica implementazione di permesso d'autore che utilizziamo per la maggior parte del software GNU è la GNU General Public License

(licenza pubblica generica GNU), abbreviata in GNU GPL. Abbiamo altri tipi di permesso d'autore che sono utilizzati in circostanze specifiche. I manuali GNU sono anch'essi protetti da permesso d'autore, ma ne usano una versione molto più semplice, perché per i manuali non è necessaria la complessità della GPL.

La Free Software Foundation

Man mano che l'interesse per Emacs aumentava, altre persone parteciparono al progetto GNU, e decidemmo che era di nuovo ora di cercare finanziamenti. Così nel 1985 fondammo la Free Software Foundation (Fondazione per il software libero), una organizzazione senza fini di lucro per lo sviluppo di software libero. La FSF fra l'altro si prese carico della distribuzione dei nastri di Emacs; più tardi estese l'attività aggiungendo sul nastro altro software libero (sia GNU che non GNU) e vendendo manuali liberi.

La FSF accetta donazioni, ma gran parte delle sue entrate è sempre stata costituita dalle vendite: copie di software libero e servizi correlati. Oggi vende CD-ROM di codice sorgente, CD-ROM di programmi compilati, manuali stampati professionalmente (tutti con libertà di ridistribuzione e modifica), e distribuzioni Deluxe (nelle quali compiliamo l'intera scelta di software per una piattaforma a richiesta).

I dipendenti della Free Software Foundation hanno scritto e curato la manutenzione di diversi pacchetti GNU. Fra questi spiccano la libreria C e la shell. La libreria C di GNU è utilizzata da ogni programma che gira su sistemi GNU/Linux per comunicare con Linux. È stata sviluppata da un membro della squadra della Free Software Foundation, Roland McGrath. La shell usata sulla maggior parte dei sistemi GNU/Linux è Bash, la Bourne Again Shell5, che è stata sviluppata da Brian Fox, dipendente della FSF.

Finanziammo lo sviluppo di questi programmi perché il progetto GNU non riguardava solo strumenti di lavoro o un ambiente di sviluppo: il nostro obiettivo era un sistema operativo completo, e questi programmi erano necessari per raggiungere quell'obiettivo.

Il supporto per il software libero

La filosofia del software libero rigetta una diffusa pratica commerciale in particolare, ma non è contro il commercio. Quando un'impresa rispetta la libertà dell'utente, c'è da augurarle ogni successo.

La vendita di copie di Emacs esemplifica un modo di condurre affari col software libero. Quando la FSF prese in carico quest'attività, dovetti trovare un'altra fonte di sostentamento. La trovai nella vendita di servizi relativi

al software libero che avevo sviluppato, come insegnare argomenti quali programmazione di Emacs e personalizzazione di GCC, oppure sviluppare software, soprattutto adattamento di GCC a nuove architetture.

Oggi tutte queste attività collegate al software libero sono esercitate da svariate aziende. Alcune distribuiscono raccolte di software libero su CD-ROM, altre offrono consulenza a diversi livelli, dall'aiutare gli utenti in difficoltà, alla correzione di errori, all'aggiunta di funzionalità non banali. Si cominciano anche a vedere aziende di software che si fondano sul lancio di nuovi programmi liberi.

Attenzione, però: diverse aziende che si fregiano del marchio "open source" (software aperto) in realtà fondano le loro attività su software non libero che funziona insieme con software libero. Queste non sono aziende di software libero, sono aziende di software proprietario i cui prodotti attirano gli utenti lontano dalla libertà. Loro li chiamano "a valore aggiunto", il che riflette i valori che a loro farebbe comodo che adottassimo: la convenienza prima della libertà. Se noi riteniamo che la libertà abbia più valore, li dovremmo chiamare prodotti "a libertà sottratta".

Obiettivi tecnici

L'obiettivo principale di GNU era essere software libero. Anche se GNU non avesse avuto alcun vantaggio tecnico su Unix, avrebbe avuto sia un vantaggio sociale, permettendo agli utenti di cooperare, sia un vantaggio etico, rispettando la loro libertà.

Tuttavia risultò naturale applicare al lavoro le regole classiche di buona programmazione; per esempio, allocare le strutture dati dinamicamente per evitare limitazioni arbitrarie sulla dimensione dei dati, o gestire tutti i possibili codici a 8 bit in tutti i casi ragionevoli.

Inoltre, al contrario di Unix che era pensato per piccole dimensioni di memoria, decidemmo di non supportare le macchine a 16 bit (era chiaro che le macchine a 32 bit sarebbero state la norma quando il sistema GNU sarebbe stato completo), e di non preoccuparci di ridurre l'occupazione di memoria a meno che eccedesse il megabyte. In programmi per i quali non era essenziale la gestione di file molto grandi, spingemmo i programmatori a leggere in memoria l'intero file di ingresso per poi analizzare il file senza doversi preoccupare delle operazioni di I/O.

Queste decisioni fecero sì che molti programmi GNU superassero i loro equivalenti Unix sia in affidabilità che in velocità di esecuzione.

Donazioni di computer

Man mano che la reputazione del progetto GNU andava crescendo, alcune persone iniziarono a donare macchine su cui girava Unix. Queste macchine erano molto utili, perché il modo più semplice di sviluppare componenti per GNU era di farlo su di un sistema Unix così da sostituire pezzo per pezzo i componenti di quel sistema. Ma queste macchine sollevavano anche una questione etica: se fosse giusto per noi anche solo possedere una copia di Unix.

Unix era (ed è) software proprietario, e la filosofia del progetto GNU diceva che non avremmo dovuto usare software proprietario. Ma, applicando lo stesso ragionamento per cui la violenza è ammessa per autodifesa, conclusi che fosse legittimo usare un pacchetto proprietario, se ciò fosse stato importante nel crearne un sostituto libero che permettesse ad altri di smettere di usare quello proprietario.

Tuttavia, benché fosse un male giustificabile, era pur sempre un male. Oggi non abbiamo più alcuna copia di Unix, perché le abbiamo sostituite con sistemi operativi liberi. Quando non fu possibile sostituire il sistema operativo di una macchina con uno libero, sostituimmo la macchina.

L'elenco dei compiti GNU

Mentre il progetto GNU avanzava, e un numero sempre maggiore di componenti di sistema venivano trovati o sviluppati, diventò utile stilare un elenco delle parti ancora mancanti. Usammo questo elenco per ingaggiare programmatori che scrivessero tali parti, e l'elenco prese il nome di elenco dei compiti GNU. In aggiunta ai componenti Unix mancanti inserimmo nell'elenco svariati progetti utili di programmazione o di documentazione che a nostro parere non dovrebbero mancare in un sistema operativo veramente completo.

Oggi non compare quasi nessun componente Unix nell'elenco dei compiti GNU; tutti questi lavori, a parte qualcuno non essenziale, sono già stati svolti. D'altro canto l'elenco è pieno di quei progetti che qualcuno chiamerebbe "applicazioni": ogni programma che interessi a una fetta non trascurabile di utenti sarebbe un'utile aggiunta a un sistema operativo.

L'elenco comprende anche dei giochi, e così è stato fin dall'inizio: Unix comprendeva dei giochi, perciò era naturale che così fosse anche per GNU. Ma poiché non c'erano esigenze di compatibilità per i giochi, non ci attenemmo alla scelta di giochi presenti in Unix, preferendo piuttosto fornire un elenco di diversi tipi di giochi potenzialmente graditi agli utenti.

La licenza GNU per le librerie

La libreria C del sistema GNU utilizza un tipo speciale di permesso d'autore, la "Licenza Pubblica GNU per le Librerie"6, che permette l'uso della libreria da parte di software proprietario. Perché quest'eccezione?

Non si tratta di questioni di principio: non c'è nessun principio che dica che i prodotti software proprietari abbiano il diritto di includere il nostro codice (perché contribuire a un progetto fondato sul rifiuto di condividere con noi?). L'uso della licenza LGPL per la libreria C, o per qualsiasi altra libreria, è una questione di strategia.

La libreria C svolge una funzione generica: ogni sistema operativo proprietario e ogni compilatore includono una libreria C. Di conseguenza, rendere disponibile la nostra libreria C solo per i programmi liberi non avrebbe dato nessun vantaggio a tali programmi liberi, avrebbe solo disincentivato l'uso della nostra libreria.

C'è un'eccezione a questa situazione: sul sistema GNU (termine che include GNU/Linux) l'unica libreria C disponibile è quella GNU. Quindi i termini di distribuzione della nostra libreria C determinano se sia possibile o meno compilare un programma proprietario per il sistema GNU. Non ci sono ragioni etiche per permettere l'uso di applicazioni proprietarie sul sistema GNU, ma strategicamente sembra che impedirne l'uso servirebbe più a scoraggiare l'uso del sistema GNU che non a incoraggiare lo sviluppo di applicazioni libere.

Ecco perché l'uso della licenza LGPL è una buona scelta strategica per la libreria C, mentre per le altre librerie la strategia va valutata caso per caso. Quando una libreria svolge una funzione particolare che può aiutare a scrivere certi tipi di programmi, distribuirla secondo la GPL, quindi limitandone l'uso ai soli programmi liberi, è un modo per aiutare gli altri autori di software libero, dando loro un vantaggio nei confronti del software proprietario.

Prendiamo come esempio GNU-Readline, una libreria scritta per fornire a Bash la modificabilità della linea di comando: Readline è distribuita secondo la normale licenza GPL, non la LGPL. Ciò probabilmente riduce l'uso di Readline, ma questo non rappresenta una perdita per noi; d'altra parte almeno una applicazione utile è stata resa software libero proprio al fine di usare Readline, e questo è un guadagno tangibile per la comunità.

Chi sviluppa software proprietario ha vantaggi economici, gli autori di programmi liberi hanno bisogno di avvantaggiarsi a vicenda. Spero che un giorno possiamo avere una grande raccolta di librerie coperte dalla licenza GPL senza che esista una raccolta equivalente per chi scrive software pro-

prietario. Tale libreria fornirebbe utili moduli da usare come i mattoni per costruire nuovi programmi liberi, e costituendo un sostanziale vantaggio per la scrittura di ulteriori programmi liberi.

Togliersi il prurito?

Eric Raymond afferma che "ogni buon programma nasce dall'iniziativa di un programmatore che si vuole togliere un suo personale prurito". È probabile che talvolta succeda così, ma molte parti essenziali del software GNU sono state sviluppate al fine di completare un sistema operativo libero. Derivano quindi da una idea e da un progetto, non da una necessità contingente.

Per esempio, abbiamo sviluppato la libreria C di GNU perché un sistema di tipo Unix ha bisogno di una libreria C, la Bourne-Again Shell (bash) perché un sistema di tipo Unix ha bisogno di una shell, e GNU tar perché un sistema di tipo Unix ha bisogno un programma tar. Lo stesso vale per i miei programmi: il compilatore GNU, GNU Emacs, GDB, GNU Make.

Alcuni programmi GNU sono stati sviluppati per fronteggiare specifiche minacce alla nostra libertà: ecco perché abbiamo sviluppato gzip come sostituto per il programma Compress, che la comunità aveva perduto a causa dei brevetti sull'algoritmo LZW. Abbiamo trovato persone che sviluppassero LessTif, e più recentemente abbiamo dato vita ai progetti GNOME e Harmony per affrontare i problemi causati da alcune librerie proprietarie (come descritto più avanti). Stiamo sviluppando la GNU Privacy Guard per sostituire i diffusi programmi di crittografia non liberi, perché gli utenti non siano costretti a scegliere tra riservatezza e libertà.

Naturalmente, i redattori di questi programmi sono coinvolti nel loro lavoro, e varie persone vi hanno aggiunto diverse funzionalità secondo le loro personali necessità e i loro interessi. Tuttavia non è questa la ragione dell'esistenza di tali programmi.

Sviluppi inattesi

All'inizio del progetto GNU pensavo che avremmo sviluppato l'intero sistema GNU e poi lo avremmo reso disponibile tutto insieme, ma le cose non andarono così.

Poiché i componenti del sistema GNU sono stati implementati su un sistema Unix, ognuno di essi poteva girare su sistemi Unix molto prima che esistesse un sistema GNU completo. Alcuni di questi programmi divennero diffusi e gli utenti iniziarono a estenderli e a renderli utilizzabili su nuovi sistemi: sulle varie versioni di Unix, incompatibili tra loro, e talvolta anche

su altri sistemi.

Questo processo rese tali programmi molto più potenti e attirò finanziamenti e collaboratori al progetto GNU; tuttavia probabilmente ritardò di alcuni anni la realizzazione di un sistema minimo funzionante, perché il tempo degli autori GNU veniva impiegato a curare la compatibilità di questi programmi con altri sistemi e ad aggiungere nuove funzionalità ai componenti esistenti, piuttosto che a proseguire nella scrittura di nuovi componenti.

GNU-Hurd

Nel 1990 il sistema GNU era quasi completo, l'unica parte significativa ancora mancante era il kernel. Avevamo deciso di implementare il nostro kernel come un gruppo di processi server che girassero sul sistema Mach. Mach è un microkernel sviluppato alla Carnegie Mellon University e successivamente all'Università dello Utah; GNU Hurd è un gruppo di server (o "herd of gnus": mandria di gnu) che gira su Mach svolgendo le funzioni del kernel Unix. L'inizio dello sviluppo fu ritardato nell'attesa che Mach fosse reso disponibile come software libero, come era stato promesso.

Una ragione di questa scelta progettuale fu di evitare quella che sembrava la parte più complessa del lavoro: effettuare il debugging del kernel senza un debugger a livello sorgente. Questo lavoro era già stato fatto, appunto in Mach, e avevamo previsto di effettuare il debugging dei server Hurd come programmi utente, con GDB. Ma questa fase si rivelò molto lunga, e il debugging dei server multi-thread che si scambiano messaggi si è rivelato estremamente complesso. Per rendere Hurd robusto furono così necessari molti anni.

Alix

Originariamente il kernel GNU non avrebbe dovuto chiamarsi Hurd; il suo nome originale era Alix, come la donna di cui ero innamorato in quel periodo. Alix, che era amministratrice di sistemi Unix, aveva sottolineato come il suo nome corrispondesse a un comune schema usato per battezzare le versioni del sistema Unix: scherzosamente diceva ai suoi amici: "qualcuno dovrebbe chiamare un kernel come me". Io non dissi nulla ma decisi di farle una sorpresa scrivendo un kernel chiamato Alix.

Le cose non andarono così. Michael Bushnell (ora Thomas), principale autore del kernel, preferì il nome Hurd, e chiamò Alix una parte del kernel, quella che serviva a intercettare le chiamate di sistema e a gestirle inviando messaggi ai server che compongono HURD.

Infine io e Alix ci lasciammo e lei cambiò nome; contemporaneamente la struttura di Hurd veniva cambiata in modo che la libreria C mandasse messaggi direttamente ai server, e così il componente Alix scomparve dal progetto. Prima che questo accadesse, però, un amico di Alix si accorse della presenza del suo nome nel codice sorgente di Hurd e glielo disse. Così il nome raggiunse il suo scopo.

Linux e GNU/Linux

GNU Hurd non è pronto per un uso non sperimentale, ma per fortuna è disponibile un altro kernel: nel 1991 Linus Torvalds sviluppò un Kernel compatibile con Unix e lo chiamò Linux. Attorno al 1992, la combinazione di Linux con il sistema GNU ancora incompleto produsse un sistema operativo libero completo (naturalmente combinarli fu un notevole lavoro di per sé). È grazie a Linux che oggi possiamo utilizzare una versione del sistema GNU.

Chiamiamo GNU/Linux questa versione del sistema, per indicare la sua composizione come una combinazione del sistema GNU col kernel Linux.

Le sfide che ci aspettano

Abbiamo dimostrato la nostra capacità di sviluppare un'ampia gamma di software libero, ma questo non significa che siamo invincibili e inarrestabili. Diverse sfide rendono incerto il futuro del software libero, e affrontarle richiederà perseveranza e sforzi costanti, talvolta per anni. Sarà necessaria quella determinazione che le persone sanno dimostrare quando danno valore alla propria libertà e non permettono a nessuno di sottrargliela. Le quattro sezioni seguenti parlano di queste sfide.

Hardware segreto

Sempre più spesso, i costruttori di hardware tendono a mantenere segrete le specifiche delle loro apparecchiature; questo rende difficile la scrittura di driver liberi che permettano a Linux e XFree86 di supportare nuove periferiche. Anche se oggi abbiamo sistemi completamente liberi, potremmo non averli domani se non saremo in grado di supportare i calcolatori di domani.

Esistono due modi per affrontare il problema. Un programmatore può ricostruire le specifiche dell'hardware usando tecniche di reverse engineering. Oppure si può scegliere hardware supportato dai programmi liberi: man mano che il nostro numero aumenta, la segretezza delle specifiche diventerà una pratica controproducente.

Il reverse engineering è difficile: avremo programmatori sufficientemente determinati da dedicarvisi? Sì, se avremo costruito una forte consapevolezza che avere programmi liberi sia una questione di principio e che i driver non liberi non sono accettabili. E succederà che molti di noi accettino di spendere un po' di più o perdere un po' più di tempo per poter usare driver liberi? Sì, se il desiderio di libertà e la determinazione a ottenerla saranno diffusi.

Librerie non libere

Una libreria non libera che giri su sistemi operativi liberi funziona come una trappola per i creatori di programmi liberi. Le funzionalità attraenti della libreria fungono da esca; chi usa la libreria cade nella trappola, perché il programma che crea è inutile come parte di un sistema operativo libero (a rigore, il programma potrebbe esservi incluso, ma non funzionerebbe, visto che manca la libreria). Peggio ancora, se un programma che usa la libreria proprietaria diventa diffuso, può attirare altri ignari programmatori nella trappola.

Il problema si concretizzò per la prima volta con la libreria Motif, negli anni '80. Sebbene non ci fossero ancora sistemi operativi liberi, i problemi che Motif avrebbe causato loro erano già chiari. Il progetto GNU reagì in due modi: interessandosi presso diversi progetti di software libero perché supportassero gli strumenti grafici X liberi in aggiunta a Motif, e cercando qualcuno che scrivesse un sostituto libero di Motif. Il lavoro richiese molti anni: solo nel 1997 LessTif, sviluppato dagli "Hungry Programmers", divenne abbastanza potente da supportare la maggior parte delle applicazioni Motif.

Tra il 1996 e il 1998 un'altra libreria non libera di strumenti grafici, chiamata Qt, veniva usata in una significativa raccolta di software libero: l'ambiente grafico KDE.

I sistemi liberi GNU/Linux non potevano usare KDE, perché non potevamo usare la libreria; tuttavia, alcuni distributori commerciali di sistemi GNU/Linux, non scrupolosi nell'attenersi solo ai programmi liberi, aggiunsero KDE ai lori sistemi, ottenendo così sistemi che offrivano più funzionalità, ma meno libertà. Il gruppo che sviluppava KDE incoraggiava esplicitamente altri programmatori a usare Qt, e milioni di nuovi "utenti Linux" non sospettavano minimamente che questo potesse costituire un problema. La situazione si faceva pericolosa.

La comunità del software libero affrontò il problema in due modi: GNOME e Harmony.

GNOME (GNU Network Object Model Environment, modello di ambiente per oggetti di rete) è il progetto GNU per l'ambiente grafico (desktop). Intrapreso nel 1997 da Miguel de Icaza e sviluppato con il supporto di Red Hat Software, GNOME si ripromise di fornire funzionalità grafiche simili a quelle di KDE, ma usando esclusivamente software libero. GNOME offre anche dei vantaggi tecnici, come il supporto per svariati linguaggi di programmazione, non solo il C++. Ma il suo scopo principale era la libertà: non richiedere l'uso di alcun programma che non fosse libero.

Harmony è una libreria compatibile con Qt, progettata per rendere possibile l'uso del software KDE senza dover usare Qt.

Nel novembre 1998 gli autori di Qt annunciarono un cambiamento di licenza che, una volta operativo, avrebbe reso Qt software libero. Non c'è modo di esserne certi, ma credo che questo fu in parte dovuto alla decisa risposta della comunità al problema posto da Qt quando non era libero (la nuova licenza è scomoda e iniqua, per cui rimane comunque preferibile evitare l'uso di Qt).

Come risponderemo alla prossima allettante libreria non libera? Riuscirà la comunità in toto a comprendere l'importanza di evitare la trappola? Oppure molti di noi preferiranno la convenienza alla libertà, creando così ancora un grave problema? Il nostro futuro dipende dalla nostra filosofia.

Brevetti sul software

Il maggior pericolo a cui ci troviamo di fronte è quello dei brevetti sul software, che possono rendere inaccessibili al software libero algoritmi e funzionalità per un tempo che può estendersi fino a vent'anni. I brevetti sugli algoritmi di compressione LZW furono depositati nel 1983, e ancor oggi non possiamo distribuire programmi liberi che producano immagini GIF compresse. Nel 1998 un programma libero per produrre audio compresso MP3 venne ritirato sotto minaccia di una causa per violazione di brevetto.

Ci sono modi per affrontare la questione brevetti: possiamo cercare prove che un brevetto non sia valido oppure possiamo cercare modi alternativi per ottenere lo stesso risultato. Ognuna di queste tecniche, però, funziona solo in certe circostanze; quando entrambe falliscono un brevetto può obbligare tutto il software libero a rinunciare a qualche funzionalità che gli utenti desiderano. Cosa dobbiamo fare quando ciò accade?

Chi fra noi apprezza il software libero per il valore della libertà rimarrà comunque dalla parte dei programmi liberi; saremo in grado di svolgere il

nostro lavoro senza le funzionalità coperte da brevetto. Ma coloro che apprezzano il software libero perché si aspettano che sia tecnicamente superiore probabilmente grideranno al fallimento quando un brevetto ne impedisce lo sviluppo. Perciò, nonostante sia utile parlare dell'efficacia pratica del modello di sviluppo "a cattedrale", e dell'affidabilità e della potenza di un dato programma libero, non ci dobbiamo fermare qui; dobbiamo parlare di libertà e di principi.

Documentazione libera

La più grande carenza nei nostri sistemi operativi liberi non è nel software, quanto nella carenza di buoni manuali liberi da includere nei nostri sistemi. La documentazione è una parte essenziale di qualunque pacchetto software; quando un importante pacchetto software libero non viene accompagnato da un buon manuale libero si tratta di una grossa lacuna. E di queste lacune attualmente ne abbiamo molte.

La documentazione libera, come il software libero, è una questione di libertà, non di prezzo. Il criterio per definire libero un manuale è fondamentalmente lo stesso che per definire libero un programma: si tratta di offrire certe libertà a tutti gli utenti. Deve essere permessa la ridistribuzione (compresa la vendita commerciale), sia in formato elettronico che cartaceo, in modo che il manuale possa accompagnare ogni copia del programma.

Autorizzare la modifica è anch'esso un aspetto cruciale; in generale, non credo sia essenziale permettere alle persone di modificare articoli e libri di qualsiasi tipo. Per esempio, non credo che voi o io dobbiamo sentirci in dovere di autorizzare la modifica di articoli come questo, articoli che descrivono le nostre azioni e il nostro punto di vista.

Ma c'è una ragione particolare per cui la libertà di modifica è cruciale per la documentazione dei programmi liberi. Quando qualcuno esercita il proprio diritto di modificare il programma, aumentandone o alterandone le funzionalità, se è coscienzioso modificherà anche il manuale, in modo da poter fornire una documentazione utile e accurata insieme al programma modificato. Un manuale che non permetta ai programmatori di essere coscienziosi e completare il loro lavoro non soddisfa i bisogni della nostra comunità.

Alcuni limiti sulla modificabilità non pongono alcun problema; per esempio, le richieste di conservare la nota di copyright dell'autore originale, i termini di distribuzione e la lista degli autori vanno bene. Non ci sono problemi nemmeno nel richiedere che le versioni modificate dichiarino esplicitamente di essere tali, così pure che intere sezioni non possano esse-

re rimosse o modificate, finché queste sezioni vertono su questioni non tecniche. Restrizioni di questo tipo non creano problemi perché non impediscono al programmatore coscienzioso di adattare il manuale perché rispecchi il programma modificato. In altre parole, non impediscono alla comunità del software libero di beneficiare appieno dal manuale.

D'altro canto, deve essere possibile modificare tutto il contenuto tecnico del manuale e poter distribuire il risultato in tutti i formati usuali, attraverso tutti i normali canali di distribuzione; diversamente, le restrizioni creerebbero un ostacolo per la comunità, il manuale non sarebbe libero e avremmo bisogno di un altro manuale.

Gli sviluppatori di software libero avranno la consapevolezza e la determinazione necessarie a produrre un'intera gamma di manuali liberi? Ancora una volta, il nostro futuro dipende dalla nostra filosofia.

Dobbiamo parlare di libertà

Stime recenti valutano in dieci milioni il numero di utenti di sistemi GNU/Linux quali Debian GNU/Linux e Red Hat Linux. Il software libero ha creato tali vantaggi pratici che gli utenti stanno approdando a esso per pure ragioni pratiche.

Gli effetti positivi di questa situazione sono evidenti: maggior interesse a sviluppare software libero, più clienti per le imprese di software libero e una migliore capacità di incoraggiare le aziende a sviluppare software commerciale libero invece che prodotti software proprietari.

L'interesse per il software, però, sta crescendo più in fretta della coscienza della filosofia su cui è basato, e questa disparità causa problemi. La nostra capacità di fronteggiare le sfide e le minacce descritte in precedenza dipende dalla determinazione nell'essere impegnati per la libertà. Per essere sicuri che la nostra comunità abbia tale determinazione, dobbiamo diffondere l'idea presso i nuovi utenti man mano che entrano a far parte della comunità.

Ma in questo stiamo fallendo: gli sforzi per attrarre nuovi utenti nella comunità sono di gran lunga maggiori degli sforzi per l'educazione civica della comunità stessa. Dobbiamo fare entrambe le cose, e dobbiamo mantenere un equilibrio fra i due impegni.

"Open Source"

Parlare di libertà ai nuovi utenti è diventato più difficile dal 1998, quando una parte della comunità decise di smettere di usare il termine "free software" e usare al suo posto "open source".

Alcune delle persone che suggerirono questo termine intendevano evitare che si confondesse "free" con "gratis", un valido obiettivo. D'altra parte, altre persone intendevano mettere da parte lo spirito del principio che aveva dato la spinta al movimento del software libero e al progetto GNU, puntando invece ad attrarre i dirigenti e gli utenti commerciali, molti dei quali afferiscono a una ideologia che pone il profitto al di sopra della libertà, della comunità, dei principi. Perciò la retorica di "open source" si focalizza sul possibilità di creare software di buona qualità e potente ma evita deliberatamente le idee di libertà, comunità, principio.

Le riviste che si chiamano "Linux..." sono un chiaro esempio di ciò: sono piene di pubblicità di software proprietario che gira sotto GNU/Linux; quando ci sarà il prossimo Motif o Qt, queste riviste avvertiranno i programmatori di starne lontano o accetteranno la sua pubblicità?

L'appoggio delle aziende può contribuire alla comunità in molti modi; a parità di tutto il resto è una cosa utile. Ma ottenere questo appoggio parlando ancor meno di libertà e principi può essere disastroso; rende ancora peggiore lo sbilanciamento descritto tra diffusione ed educazione civica.

"Software libero" (free software) e "sorgente aperto" (open source) descrivono più o meno la stessa categoria di software, ma dicono cose differenti sul software e sui valori. Il progetto GNU continua a usare il termine "software libero" per esprimere l'idea che la libertà sia importante, non solo la tecnologia.

Prova!

La filosofia di Yoda ("Non c'è provare") suona bene, ma per me non funziona. Ho fatto la maggior parte del mio lavoro angustiato dal timore di non essere in grado di svolgere il mio compito e nel dubbio, se fossi riuscito, che non fosse sufficiente per raggiungere l'obiettivo. Ma ci ho provato in ogni caso perché nessuno tranne me si poneva tra il nemico e la mia città. Sorprendendo me stesso, qualche volta sono riuscito.

A volte ho fallito, alcune delle mie città sono cadute; poi ho trovato un'altra città minacciata e mi sono preparato a un'altra battaglia. Con l'andar del tempo ho imparato a cercare le possibili minacce e a mettermi tra loro e la mia città, facendo appello ad altri hacker perché venissero e si unissero a me.

Oggigiorno spesso non sono da solo. È un sollievo e una gioia quando vedo un reggimento di hacker che scavano trincee per difendere il confine e quando mi rendo conto che questa città può sopravvivere; per ora. Ma i pericoli diventano più grandi ogni anno, e ora Microsoft ha esplicitamente

preso di mira la nostra comunità. Non possiamo dare per scontato il futuro della libertà; non diamolo per scontato! Se volete mantenere la vostra libertà dovete essere pronti a difenderla.

* La traduzione di questo saggio è stata revisionata e curata, con la supervisione dell'autore, da Lorenzo Bettini, Antonio Cisternino, Francesco Potortì e Alessandro Rubini.

1. L'uso del termine "hacker" nel senso di "pirata" è una confusione di termini creata dai mezzi di informazione. Noi hacker ci rifiutiamo di riconoscere questo significato, e continuiamo a utilizzare la parola nel senso di "uno che ami programmare, e a cui piaccia essere bravo a farlo"

2. Essendo ateo, non seguo alcuna guida religiosa, ma a volte mi trovo ad ammirare qualcosa che qualcuno di loro ha detto.

3. Nel 1984 o 1985, Don Hopkins, persona molto creativa, mi mandò una lettera. Sulla busta aveva scritto diverse frasi argute, fra cui questa: "Permesso d'autore - tutti i diritti rovesciati". Utilizzai l'espressione "permesso d'autore" per battezzare il concetto di distribuzione che allora andavo elaborando.

4. NdT: si tratta di un gioco di parole, che qui viene reso con "permesso di autore": copyright (diritto di autore) è formato dalle parola "copy" (copia) e "right" (diritto, ma anche destra), opposto di "left" (sinistra, ma anche lasciato).

5. "Bourne Again Shell" è un gioco di parole sul nome "Bourne Shell", che era la normale shell di Unix. NdT: "Bourne again" richiama l'espressione cristiana "born again", "rinato" (in Cristo).

6. N.d.T. Nel 1999 la FSF ha cambiato nome alla licenza LGPL che ora si chiama "Lesser GPL", GPL attenuata, per non suggerire che si tratti della forma di licenza preferenziale per le librerie.

La licenza GNU GPL

La licenza GPL è la capostipite delle licenze copyleft. Oltre ad essere la prima comparsa e resa pubblica per il suo libero utilizzo, è la più utilizzata in assoluto. Penso sia attualmente impossibile fare un computo delle righe di codice rilasciate sotto la sua disciplina, ma per capirne la portata basta dire che essa accompagna tutte le distribuzioni GNU/Linux e software come Openoffice, GIMP, E-Mule. Rappresenta la cristallizzazione del modello copyleft come inteso dai suoi ideatori Stallman e Moglen, massimi portavoce della Free Software Foundation; dunque qualsiasi programma informatico che voglia professarsi "free software" nel senso originario del termine deve utilizzare questa licenza, oppure una licenza le cui clausole non contraddicano gli effetti pratici e giuridici della GPL (cioè che sia GPL-compatibile).

Questa licenza è stata da più parti criticata per la sua rigidità e per la sua poca asetticità. Infatti, riguardo alla prima critica, la licenza è "persistente", cioè se modifichiamo e sviluppiamo un'opera sotto GPL, dobbiamo a nostra volta rilasciare le modifiche sotto GPL; ed è "virale", cioè se nella realizzazione di un software misceliamo del codice sotto GPL con altro codice (originariamente non sotto GPL), siamo costretti a rilasciare tutto il codice di quel software sotto GPL. Riguardo alla seconda critica, invece, bisogna rilevare che il testo della licenza in effetti comprende molte

parti di tipo più propagandistico-divulgativo che definitorio-contrattuale e ciò rischia di appesantire uno strumento che di per sé dovrebbe essere puramente giuridico (si pensi ad esempio al lungo preambolo che suona proprio come un manifesto ideologico).

Ciononostante la licenza è usata costantemente ed efficacemente da più di quindici anni: la sua prima versione risale al 1989, nel 1991 è stata pubblicata una seconda versione e di recente la Free Software Foundation ha aperto un grande forum di raccolta di pareri e indicazioni da parte della comunità di utenti per la redazione di una Versione 3. Tutti si aspettano una versione innovativa e, dalla prima bozza comparsa nel gennaio 2006, pare che toccherà esplicitamente le problematiche derivanti dalla brevettazione di software. Ci si auspica inoltre una redazione in più lingue ufficiali sullo stile delle licenze Creative Commons: attualmente infatti tutte le licenze GNU sono rilasciate solo in lingua inglese e le traduzioni che circolano in rete sono tutte a scopo unicamente informativo. [S. Aliprandi]

Questa è una traduzione italiana non ufficiale della Licenza Pubblica Generica GNU. Non è pubblicata dalla Free Software Foundation e non ha valore legale nell'esprimere i termini di distribuzione del software che usa la licenza GPL. Solo la versione originale in inglese della licenza ha valore legale. Ad ogni modo, speriamo che questa traduzione aiuti le persone di lingua italiana a capire meglio il significato della licenza GPL.

This is an unofficial translation of the GNU General Public License into Italian. It was not published by the Free Software Foundation, and does not legally state the distribution terms for software that uses the GNU GPL -- only the original English text of the GNU GPL does that. However, we hope that this translation will help Italian speakers understand the GNU GPL better.

LICENZA PUBBLICA GENERICA (GPL) DEL PROGETTO GNU
Versione 2, Giugno 1991

Traduzione curata da gruppo Pluto, da ILS e dal gruppo italiano di traduzione GNU. Ultimo aggiornamento 19 aprile 2000.
Traduzione tratta dalla pagina web http://www.softwarelibero.it/gnudoc/gpl.it.txt

Preambolo

Le licenze della maggior parte dei programmi hanno lo scopo di togliere all'utente la libertà di condividere e modificare il programma stesso. Viceversa, la Licenza Pubblica Generica GNU è intesa a garantire la libertà di condividere e modificare il software libero, al fine di assicurare che i programmi siano liberi per tutti i loro utenti. Questa Licenza si applica alla maggioranza dei programmi della Free Software Foundation e ad ogni altro programma i cui autori hanno deciso di usare questa Licenza. Alcuni altri programmi della Free Software Foundation sono invece coperti dalla Licenza Pubblica Generica Minore. Chiunque può usare questa Licenza per i propri programmi.

Quando si parla di software libero (free software), ci si riferisce alla libertà, non al prezzo. Le nostre Licenze (la GPL e la LGPL) sono progettate per assicurarsi che ciascuno abbia la libertà di distribuire copie del software libero (e farsi pagare per questo, se vuole), che ciascuno riceva il codice sorgente o che lo possa ottenere se lo desidera, che ciascuno possa modificare il programma o usarne delle parti in nuovi programmi liberi e che ciascuno sappia di potere fare queste cose.

Per proteggere i diritti dell'utente, abbiamo bisogno di creare delle restrizioni che vietino a chiunque di negare questi diritti o di chiedere di rinunciarvi. Queste restrizioni si traducono in certe responsabilità per chi distribuisce copie del software e per chi lo modifica.

Per esempio, chi distribuisce copie di un programma coperto da GPL, sia gratis sia in cambio di un compenso, deve concedere ai destinatari tutti i diritti che ha ricevuto. Deve anche assicurarsi che i destinatari ricevano o possano ottenere il codice sorgente. E deve mostrar loro queste condizioni di licenza, in modo che essi conoscano i propri diritti.

Proteggiamo i diritti dell'utente in due modi: (1) proteggendo il software con un copyright, e (2) offrendo una licenza che dia il permesso legale di copiare, distribuire e modificare il Programma.

Inoltre, per proteggere ogni autore e noi stessi, vogliamo assicurarci che ognuno capisca che non ci sono garanzie per i programmi coperti da GPL. Se il programma viene modificato da qualcun altro e ridistribuito, vogliamo che gli acquirenti sappiano che ciò che hanno non è l'originale, in modo che ogni problema introdotto da altri non si rifletta sulla reputazione degli autori originari.

Infine, ogni programma libero è costantemente minacciato dai brevetti sui programmi. Vogliamo evitare il pericolo che chi ridistribuisce un programma libero ottenga la proprietà di brevetti, rendendo in pratica il programma cosa di sua proprietà. Per prevenire questa evenienza, abbiamo chiarito che ogni brevetto debba essere concesso in licenza d'uso a chiunque, o non avere alcuna restrizione di licenza d'uso.

Seguono i termini e le condizioni precisi per la copia, la distribuzione e la modifica.

LICENZA PUBBLICA GENERICA GNU
TERMINI E CONDIZIONI PER LA COPIA, LA DISTRIBUZIONE E LA MODIFICA

0. Questa Licenza si applica a ogni programma o altra opera che contenga una nota da parte del detentore del copyright che dica che tale opera può essere distribuita sotto i termini di questa Licenza Pubblica Generica. Il termine "Programma" nel seguito si riferisce ad ogni programma o opera così definita, e l'espressione "opera basata sul Programma" indica sia il Programma sia ogni opera considerata "derivata" in base alla legge sul copyright; in altre parole, un'opera contenente il Programma o una porzione di esso, sia letteralmente sia modificato o tradotto in un'altra lingua. Da qui in avanti, la traduzione è in ogni caso considerata una "modifica". Vengono ora elencati i diritti dei beneficiari della licenza.
Attività diverse dalla copiatura, distribuzione e modifica non sono coperte da questa Licenza e sono al di fuori della sua influenza. L'atto di eseguire il Programma non viene limitato, e l'output del programma è coperto da questa Licenza solo se il suo contenuto costituisce un'opera basata sul Programma (indipendentemente dal fatto che sia stato creato eseguendo il Programma). In base alla natura del Programma il suo output può essere o meno coperto da questa Licenza.

1. È lecito copiare e distribuire copie letterali del codice sorgente del Programma così come viene ricevuto, con qualsiasi mezzo, a condizione che venga riprodotta chiaramente su ogni copia una appropriata nota di copyright e di assenza di garanzia; che si mantengano intatti tutti i riferimenti a questa Licenza e all'assenza di ogni garanzia; che si dia a ogni altro destinatario del Programma una copia di questa Licenza insieme al Programma.
È possibile richiedere un pagamento per il trasferimento fisico di una copia del Programma, è anche possibile a propria discrezione richiedere un pagamento in cambio di una copertura assicurativa.

2. È lecito modificare la propria copia o copie del Programma, o parte di esso, creando perciò un'opera basata sul Programma, e copiare o distribuire tali modifiche o tale opera secondo i termini del precedente comma 1, a patto che siano soddisfatte tutte le condizioni che seguono:
a) Bisogna indicare chiaramente nei file che si tratta di copie modificate e la data di ogni modifica.
b) Bisogna fare in modo che ogni opera distribuita o pubblicata, che in parte o nella sua totalità derivi dal Programma o da parti di esso, sia concessa nella sua interezza in licenza gratuita ad ogni terza parte, secondo i termini di questa Licenza.
c) Se normalmente il programma modificato legge comandi interattivamente quando viene eseguito, bisogna fare in modo che all'inizio dell'esecuzione interattiva usuale, esso stampi un messaggio contenente una appropriata nota di copyright e di assenza di garanzia (oppure che specifichi il tipo di garanzia che si offre). Il messaggio

deve inoltre specificare che chiunque può ridistribuire il programma alle condizioni qui descritte e deve indicare come reperire questa Licenza. Se però il programma di partenza è interattivo ma normalmente non stampa tale messaggio, non occorre che un'opera basata sul Programma lo stampi.
Questi requisiti si applicano all'opera modificata nel suo complesso. Se sussistono parti identificabili dell'opera modificata che non siano derivate dal Programma e che possono essere ragionevolmente considerate lavori indipendenti, allora questa Licenza e i suoi termini non si applicano a queste parti quando queste vengono distribuite separatamente. Se però queste parti vengono distribuite all'interno di un prodotto che è un'opera basata sul Programma, la distribuzione di quest'opera nella sua interezza deve avvenire nei termini di questa Licenza, le cui norme nei confronti di altri utenti si estendono all'opera nella sua interezza, e quindi ad ogni sua parte, chiunque ne sia l'autore.
Quindi, non è nelle intenzioni di questa sezione accampare diritti, né contestare diritti su opere scritte interamente da altri; l'intento è piuttosto quello di esercitare il diritto di controllare la distribuzione di opere derivati dal Programma o che lo contengano.
Inoltre, la semplice aggregazione di un'opera non derivata dal Programma col Programma o con un'opera da esso derivata su di un mezzo di memorizzazione o di distribuzione, non è sufficente a includere l'opera non derivata nell'ambito di questa Licenza.

3. È lecito copiare e distribuire il Programma (o un'opera basata su di esso, come espresso al comma 2) sotto forma di codice oggetto o eseguibile secondo i termini dei precedenti commi 1 e 2, a patto che si applichi una delle seguenti condizioni:
a) Il Programma sia corredato del codice sorgente completo, in una forma leggibile da calcolatore, e tale sorgente sia fornito secondo le regole dei precedenti commi 1 e 2 su di un mezzo comunemente usato per lo scambio di programmi.
b) Il Programma sia accompagnato da un'offerta scritta, valida per almeno tre anni, di fornire a chiunque ne faccia richiesta una copia completa del codice sorgente, in una forma leggibile da calcolatore, in cambio di un compenso non superiore al costo del trasferimento fisico di tale copia, che deve essere fornita secondo le regole dei precedenti commi 1 e 2 su di un mezzo comunemente usato per lo scambio di programmi.
c) Il Programma sia accompagnato dalle informazioni che sono state ricevute riguardo alla possibilità di ottenere il codice sorgente. Questa alternativa è permessa solo in caso di distribuzioni non commerciali e solo se il programma è stato ottenuto sotto forma di codice oggetto o eseguibile in accordo al precedente comma B.
Per "codice sorgente completo" di un'opera si intende la forma preferenziale usata per modificare un'opera. Per un programma eseguibile, "codice sorgente completo" significa tutto il codice sorgente di tutti i moduli in esso contenuti, più ogni file associato che definisca le interfacce esterne del programma, più gli script usati per controllare la compilazione e l'installazione
dell'eseguibile. In ogni caso non è necessario che il codice sorgente fornito includa nulla

che sia normalmente distribuito (in forma sorgente o in formato binario) con i principali componenti del sistema operativo sotto cui viene eseguito il Programma (compilatore, kernel, e così via), a meno che tali componenti accompagnino l'eseguibile.
Se la distribuzione dell'eseguibile o del codice oggetto è effettuata indicando un luogo dal quale sia possibile copiarlo, permettere la copia del codice sorgente dallo stesso luogo è considerata una valida forma di distribuzione del codice sorgente, anche se copiare il sorgente è facoltativo per l'acquirente.

4. Non è lecito copiare, modificare, sublicenziare, o distribuire il Programma in modi diversi da quelli espressamente previsti da questa Licenza. Ogni tentativo di copiare, modificare, sublicenziare o distribuire altrimenti il Programma non è autorizzato, e farà terminare automaticamente i diritti garantiti da questa Licenza. D'altra parte ogni acquirente che abbia ricevuto copie, o diritti, coperti da questa Licenza da parte di persone che violano la Licenza come qui indicato non vedranno invalidata la loro Licenza, purché si comportino conformemente ad essa.

5. L'acquirente non è tenuto ad accettare questa Licenza, poiché non l'ha firmata. D'altra parte nessun altro documento garantisce il permesso di modificare o distribuire il Programma o i lavori derivati da esso. Queste azioni sono proibite dalla legge per chi non accetta questa Licenza; perciò, modificando o distribuendo il Programma o un'opera basata sul programma, si indica nel fare ciò l'accettazione di questa Licenza e quindi di tutti i suoi termini e le condizioni poste sulla copia, la distribuzione e la modifica del Programma o di lavori basati su di esso.

6. Ogni volta che il Programma o un'opera basata su di esso vengono distribuiti, l'acquirente riceve automaticamente una licenza d'uso da parte del licenziatario originale. Tale licenza regola la copia, la distribuzione e la modifica del Programma secondo questi termini e queste condizioni. Non è lecito imporre restrizioni ulteriori all'acquirente nel suo esercizio dei diritti qui garantiti. Chi distribuisce programmi coperti da questa Licenza non e' comunque tenuto a imporre il rispetto di questa Licenza a terzi.

7. Se, come conseguenza del giudizio di un tribunale, o di una imputazione per la violazione di un brevetto o per ogni altra ragione (non limitatamente a questioni di brevetti), vengono imposte condizioni che contraddicono le condizioni di questa licenza, che queste condizioni siano dettate dalla corte, da accordi tra le parti o altro, queste condizioni non esimono nessuno dall'osservazione di questa Licenza. Se non è possibile distribuire un prodotto in un modo che soddisfi simultaneamente gli obblighi dettati da questa Licenza e altri obblighi pertinenti, il prodotto non può essere affatto distribuito. Per esempio, se un brevetto non permettesse a tutti quelli che lo ricevono di ridistribuire il Programma senza obbligare al pagamento di diritti, allora l'unico modo per soddisfare contemporaneamente il brevetto e questa Licenza e' di non distribuire affatto il Programma.

Se una qualunque parte di questo comma è ritenuta non valida o non applicabile in una qualunque circostanza, deve comunque essere applicata l'idea espressa da questo comma; in ogni altra circostanza invece deve essere applicato questo comma nel suo complesso.
Non è nelle finalità di questo comma indurre gli utenti ad infrangere alcun brevetto né ogni altra rivendicazione di diritti di proprietà, né di contestare la validità di alcuna di queste rivendicazioni; lo scopo di questo comma è unicamente quello di proteggere l'integrità del sistema di distribuzione dei programmi liberi, che viene realizzato tramite l'uso di licenze pubbliche. Molte persone hanno contribuito generosamente alla vasta gamma di programmi distribuiti attraverso questo sistema, basandosi sull'applicazione fedele di tale sistema. L'autore/donatore può decidere di sua volontà se preferisce distribuire il software avvalendosi di altri sistemi, e l'acquirente non può imporre la scelta del sistema di distribuzione.
Questo comma serve a rendere il più chiaro possibile ciò che crediamo sia una conseguenza del resto di questa Licenza.

8. Se in alcuni paesi la distribuzione o l'uso del Programma sono limitati da brevetto o dall'uso di interfacce coperte da copyright, il detentore del copyright originale che pone il Programma sotto questa Licenza può aggiungere limiti geografici espliciti alla distribuzione, per escludere questi paesi dalla distribuzione stessa, in modo che il programma possa essere distribuito solo nei paesi non esclusi da questa regola. In questo caso i limiti geografici sono inclusi in questa Licenza e ne fanno parte a tutti gli effetti.

9. All'occorrenza la Free Software Foundation può pubblicare revisioni o nuove versioni di questa Licenza Pubblica Generica. Tali nuove versioni saranno simili a questa nello spirito, ma potranno differire nei dettagli al fine di coprire nuovi problemi e nuove situazioni.
Ad ogni versione viene dato un numero identificativo. Se il Programma asserisce di essere coperto da una particolare versione di questa Licenza e "da ogni versione successiva", l'acquirente può scegliere se seguire le condizioni della versione specificata o di una successiva. Se il Programma non specifica quale versione di questa Licenza deve applicarsi, l'acquirente può scegliere una qualsiasi versione tra quelle pubblicate dalla Free Software Foundation.

10. Se si desidera incorporare parti del Programma in altri programmi liberi le cui condizioni di distribuzione differiscano da queste, è possibile scrivere all'autore del Programma per chiederne l'autorizzazione. Per il software il cui copyright è detenuto dalla Free Software Foundation, si scriva alla Free Software Foundation; talvolta facciamo eccezioni alle regole di questa Licenza. La nostra decisione sarà guidata da due finalità: preservare la libertà di tutti i prodotti derivati dal nostro software libero e promuovere la condivisione e il riutilizzo del software in generale.

NON C'È GARANZIA

11. POICHÉ IL PROGRAMMA È CONCESSO IN USO GRATUITAMENTE, NON C'È GARANZIA PER IL PROGRAMMA, NEI LIMITI PERMESSI DALLE VIGENTI LEGGI. SE NON INDICATO DIVERSAMENTE PER ISCRITTO, IL DETENTORE DEL COPYRIGHT E LE ALTRE PARTI FORNISCONO IL PROGRAMMA "COSÌ COM'È", SENZA ALCUN TIPO DI GARANZIA, NÉ ESPLICITA NÉ IMPLICITA; CIÒ COMPRENDE, SENZA LIMITARSI A QUESTO, LA GARANZIA IMPLICITA DI COMMERCIABILITÀ E UTILIZZABILITÀ PER UN PARTICOLARE SCOPO. L'INTERO RISCHIO CONCERNENTE LA QUALITÀ E LE PRESTAZIONI DEL PROGRAMMA È DELL'ACQUIRENTE. SE IL PROGRAMMA DOVESSE RIVELARSI DIFETTOSO, L'ACQUIRENTE SI ASSUME IL COSTO DI OGNI MANUTENZIONE, RIPARAZIONE O CORREZIONE NECESSARIA.

12. NÉ IL DETENTORE DEL COPYRIGHT NÉ ALTRE PARTI CHE POSSONO MODIFICARE O RIDISTRIBUIRE IL PROGRAMMA COME PERMESSO IN QUESTA LICENZA SONO RESPONSABILI PER DANNI NEI CONFRONTI DELL'ACQUIRENTE, A MENO CHE QUESTO NON SIA RICHIESTO DALLE LEGGI VIGENTI O APPAIA IN UN ACCORDO SCRITTO. SONO INCLUSI DANNI GENERICI, SPECIALI O INCIDENTALI, COME PURE I DANNI CHE CONSEGUONO DALL'USO O DALL'IMPOSSIBILITÀ DI USARE IL PROGRAMMA; CIÒ COMPRENDE, SENZA LIMITARSI A QUESTO, LA PERDITA DI DATI, LA CORRUZIONE DEI DATI, LE PERDITE SOSTENUTE DALL'ACQUIRENTE O DA TERZI E L'INCAPACITÀ DEL PROGRAMMA A INTERAGIRE CON ALTRI PROGRAMMI, ANCHE SE IL DETENTORE O ALTRE PARTI SONO STATE AVVISATE DELLA POSSIBILITÀ DI QUESTI DANNI.

FINE DEI TERMINI E DELLE CONDIZIONI

Appendice: come applicare questi termini a nuovi programmi

Se si sviluppa un nuovo programma e lo si vuole rendere della maggiore utilità possibile per il pubblico, la cosa migliore da fare è rendere tale programma libero, cosicché ciascuno possa ridistribuirlo e modificarlo sotto questi termini.

Per fare questo, si inserisca nel programma la seguente nota. La cosa migliore da fare è mettere la nota all`inizio di ogni file sorgente, per chiarire nel modo più efficiente possibile l'assenza di garanzia; ogni file dovrebbe contenere almeno la nota di copyright e l'indicazione di dove trovare l'intera nota.

> *<una riga per dire in breve il nome del programma e cosa fa>*
> *Copyright (C) <anno> <nome dell'autore>*
>
> Questo programma è software libero; è lecito redistribuirlo o modificarlo secondo i termini della Licenza Pubblica Generica GNU come è pubblicata dalla Free Software Foundation; o la versione 2 della licenza o (a propria scelta) una versione successiva. Questo programma è distribuito nella speranza che sia utile, ma SENZA ALCUNA GARANZIA; senza neppure la garanzia implicita di NEGOZIABILITÀ o di APPLICABILITÀ PER UN PARTICOLARE SCOPO. Si veda la Licenza Pubblica Generica GNU per avere maggiori dettagli.
>
> Questo programma deve essere distribuito assieme ad una copia della Licenza Pubblica Generica GNU; in caso contrario, se ne può ottenere una scrivendo alla Free Software Foundation, Inc.,
> 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Si aggiungano anche informazioni su come si può essere contattati tramite posta elettronica e cartacea.

Se il programma è interattivo, si faccia in modo che stampi una breve nota simile a questa quando viene usato interattivamente:

> *Orcaloca versione 69, Copyright (C) anno nome dell'autore*
> Orcaloca non ha ALCUNA GARANZIA; per dettagli usare il comando `show g'.
> Questo è software libero, e ognuno è libero di ridistribuirlo secondo certe condizioni; usare il comando `show c' per i dettagli.

Gli ipotetici comandi "show g" e "show c" mostreranno le parti appropriate della Licenza Pubblica Generica. Chiaramente, i comandi usati possono essere chiamati diversamente da "show g" e "show c" e possono anche essere selezionati con il mouse o attraverso un menù, o comunque sia pertinente al programma.

Se necessario, si deve anche far firmare al proprio datore di lavoro (per chi lavora come programmatore) o alla propria scuola, per chi è studente, una "rinuncia al copyright" per il programma. Ecco un esempio con nomi fittizi:

> Yoyodinamica SPA rinuncia con questo documento ad ogni diritto sul copyright del programma `Orcaloca' (che svolge dei passi di compilazione) scritto da Giovanni Smanettone.
>
> <firma di Primo Tizio>, 1 April 3000
> Primo Tizio, Presidente

I programmi coperti da questa Licenza Pubblica Generica non possono essere incorporati all'interno di programmi proprietari. Se il proprio programma è una libreria di funzioni, può essere più utile permettere di collegare applicazioni proprietarie alla libreria. Se si ha questa intenzione consigliamo di usare la Licenza Pubblica Generica Minore GNU (LGPL) invece di questa Licenza.

La licenza GNU LGPL

I problemi derivati dall'impostazione rigida e "virale" della GPL (come illustrati poco fa nella relativa introduzione) hanno creato alcuni problemi in certe fasi dello sviluppo di software. Ci sono infatti alcuni tipi di software, come le librerie di funzioni, che per la loro natura strumentale devono essere combinati con altro software per poter risultare utili. In questo caso, l'obbligo di rilasciare tutto il pacchetto software sotto GPL poteva risultare un deterrente allo sviluppo di librerie libere. Ecco che la Free Software Foundation ha pensato alla redazione di una sorta di "sorella minore" della GPL, chiamata infatti Lesser GPL, che - in via del tutto straordinaria - risultasse più elastica nelle sue clausole. Ovviamente la Free Software Foundation tiene a precisare l'uso di tale licenza è consigliato solo nei casi in cui non ci sia alternativa e ricorda che la licenza "madre" per il software libero resta la GPL. Ho scelto di riportare solo una versione sintetica come rielaborata da Creative Commons [si veda più avanti] per il fatto che gran parte delle clausole della licenza coincidono con quelle della GPL e il testo ufficiale è davvero lunghissimo; dunque, in questa sede, la versione sintetica è sufficiente per coglierne il senso.

[S. Aliprandi]

GNU Lesser General Public License, Free Software Foundation
(versione "commons deed" in Italiano, rilasciata da Creative Commons)

La GNU Lesser General Public License è una licenza Software Libero. Come tutte le licenze Software Libero ti garantisce le seguenti quattro libertà:

0. La libertà di eseguire il programma, per qualsiasi scopo.
1. La libertà di studiare come funziona il programma e adattarlo alle proprie esigenze.
2. La libertà di ridistribuire copie in modo da aiutare il prossimo.
3. La libertà di migliorare il programma e distribuirne pubblicamente i miglioramenti, in modo tale che tutta la comunità ne tragga beneficio.

Si possono esercitare le libertà qui specificate, a patto di rispettare le condizioni esplicitate in questa licenza. La licenza Gnu-Lgpl è pensata più per le librerie di funzioni che per programmi eseguibili completi. Le condizioni più importanti sono:

- Si deve riprodurre chiaramente su ogni copia un'appropriata nota di copyright e di assenza di garanzia; si devono mantenere intatti tutti i riferimenti a questa licenza e all'assenza di ogni garanzia; si deve dare a ogni altro destinatario del programma una copia della GNU Lesser General Public License insieme al programma. Ogni traduzione della GNU Lesser General Public License deve essere accompagnata dalla GNU Lesser General Public License.

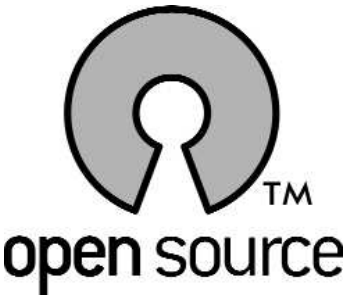
- Se si modifica la propria copia o le proprie copie della Libreria o parte di essa, si può redistribuire la libreria risultante a patto di farlo sotto le condizioni della Licenza Generica Minore Gnu. Ogni traduzione della Licenza Generica Minore Gnu deve essere accompagnata dal testo originale della Gnu Lesser General Public License.

- Se si copia o si distribuisce la Libreria, si deve corredarla del corrispondente codice sorgente completo, in una forma leggibile dal calcolatore o di un'offerta scritta, valida per almeno tre anni, di fornire un copia completa del codice sorgente corrispondente, in una forma leggibile dal calcolatore. Non serve fornire il codice sorgente dei programmi che si collegano alla Libreria.

Se ottieni il permesso dal titolare del diritto d'autore, è possibile rinunciare ad ognuna di queste condizioni.
Le tue utilizzazioni libere e gli altri diritti non sono in nessun modo limitati da quanto sopra.

Questo è un riassunto in lingua corrente dei concetti chiave della licenza completa
(codice legale) Gnu Lesser General Public License
[disponibile al sito http://www.gnu.org/copyleft/lesser.html].

Parte terza
OPEN SOURCE



Nella pagina precedente:
Il logo della Open Source Initiative

The Open Source Definition
(di Bruce Perens)

Verso la fine degli anni 90 alcuni esponenti del mondo free software iniziarono a riflettere su alcune problematiche di fondo. Il mondo dell'informatica di massa era pronto per entrare in contatto con questa realtà emergente e il successo dei sistemi GNU/Linux ne era la conferma. Bisognava però fare i conti con una certa diffidenza dell'imprenditoria software verso alcune caratteristiche del modello proposta dal Progetto GNU. Semplificando (forse troppo) la questione, i punti dolenti erano più o meno questi: l'aggettivo "free" in inglese rende il concetto di "libero" ma anche di "gratuito" e ciò poteva risultare un deterrente all'ingresso di alcune imprese in questo nuovo mercato; l'apparato di principi etici piuttosto monolitico che accompagnava l'operato della Free Software Foundation (e secondo alcuni anche l'approccio un po' "oltranzista" dello stesso Stallman) risultavano scomodi compagni di una promozione del modello freesoftware anche a livello commerciale.

Fu così che nel 1998 Bruce Perens (attivista di prima linea nello sviluppo del progetto Debian) con l'appoggio di altri esponenti del movimento (primo fra tutti Eric Raymond) si preoccupò di ridisegnare le tracce del modello, rendendolo più consono a queste nuove esigenze. Venne

così coniato il termine - indubbiamente efficace - "Open Source" (lett. "sorgente aperto") che appunto cerca di focalizzare l'attenzione sulle caratteristiche tecniche del software (la disponibilità del codice sorgente) piuttosto che sugli aspetti etici (libertà, condivisione).

Questo testo rappresenta appunto la "definizione di Open Source": è il documento che Perens ha redatto come "decalogo" di riferimento per chiarire a priori cosa possa essere ricondotto al concetto di Open Source. Contemporaneamente è stato fondato un ente che coordinasse le attività di promozione, tutela e informazione, cioè la Open Source Initiative (OSI); ed è stato registrato il marchio "OSI certified" (cioè "certificato dall'OSI") di cui possono fregiarsi tutti i software che rispondono alle caratteristiche previste dalla Open Source Definition.

E' un modus operandi *piuttosto diverso rispetto a quello della FSF (e infatti da quest'ultima molto criticato); ma che comunque ha portato i suoi frutti, facendo sì che molte imprese si affacciassero davvero su questo mercato.*

Questo saggio - scritto dallo stesso Perens - spiega l'evoluzione della OSI e commenta nel dettaglio il testo della Open Source Definition. E' il caso di tener presente però che questo saggio risale al 1998 e non esiste una versione aggiornata; mentre il testo della OSD ha subito numerose integrazioni e modifiche. Tuttavia il senso e l'importanza di questo articolo non vengono meno. E' possibile ricostruire la cronologia delle innovazioni intervenute al sito ufficiale www.opensource.org e in nota cercherò di chiarire le modifiche più sostanziali.

[S. Aliprandi]

L'utente tipico di computer possiede una discreta quantità di software che ha comprato nel tempo e che ormai non adopera più. Magari ha aggiornato il computer, o ha cambiato marca, e i vecchi programmi hanno smesso di funzionare. Magari il software è diventato obsoleto. O semplicemente il programma non lo soddisfa. Forse ha comprato due o più computer e non vuole pagare per una seconda copia del software. Quale che sia la ragione, il software per cui ha pagato anni fa non è più adeguato. Tutto questo è inevitabile?

Non sarebbe bello avere diritto a un aggiornamento gratuito ogni volta che il software lo richiede? Se, passando da un Mac a un PC, si potesse cambiare la versione del software gratis? Se, quando il software non fun-

ziona o non è abbastanza potente, si potesse migliorarlo o perfino ripararlo da soli? Se il software continuasse a essere supportato anche quando l'azienda produttrice abbia cessato l'attività? Non sarebbe bello usare il software sulla stazione di lavoro, sul computer di casa e sul portatile, anziché su un solo computer? È probabile che, in quel caso, si starebbe ancora usando il software pagato anni prima. Questi sono alcuni dei diritti che l'Open Source riconosce.

La Open Source Definition è una carta dei diritti dell'utente di computer. Definisce certi diritti che una licenza software deve garantire per poter essere certificata come Open Source. I produttori che non rendono Open Source il loro programmi trovano difficile competere con chi lo fa, dal momento che gli utenti imparano ad apprezzare quei diritti che avrebbero dovuto sempre essere loro. Programmi come il sistema operativo Linux e il browser Web Netscape sono diventati popolarissimi, scalzando altro software sottoposto a licenze più restrittive. Aziende che usano software Open Source godono il vantaggio del suo rapidissimo sviluppo, spesso a opera cooperativa di numerose aziende, e in gran parte fornito da soggetti individuali che operano migliorie sulla base di proprie necessità.

I volontari che hanno reso possibili prodotti come Linux ci sono, e le aziende possono cooperare, solo grazie ai diritti che vengono con l'Open Source. Il programmatore medio si sentirebbe stupido nel riversare tanto lavoro in un programma per vedere poi le sue migliorie vendute dal proprietario senza che lui ne riceva alcun ritorno. Quegli stessi programmatori contribuiscono volentieri all'Open Source perché esso assicura loro questi diritti:

· il diritto di fare copie del programma e di distribuirle;

· il diritto d'accesso al codice sorgente del software, condizione necessaria per poterlo modificare;

· il diritto di apportare migliorie al programma.

Questi diritti sono importanti per coloro che collaborano a un software perché li mantengono tutti al medesimo livello. Chiunque lo voglia è libero di vendere un programma Open Source, così i prezzi rimarranno bassi e sarà rapido lo sviluppo per raggiungere nuovi mercati. Chiunque investa il suo tempo a costruire conoscenza in un programma Open Source lo può supportare, e questo permette agli utenti l'opzione di fornire a loro volta il loro supporto, o l'economia dovuta a un gran numero di fornitori di supporto concorrenti. Qualunque programmatore può adattare un programma Open Source a misura di mercati specifici per raggiungere clienti nuovi. Chi lo fa non è costretto a pagare diritti o concessioni di licenza.

La ragione per il successo di una strategia che può suonare alquanto

comunista proprio nel momento in cui il fallimento del comunismo stesso è visibile ovunque, è nel fatto che l'economia dell'informazione è sostanzialmente diversa da quella degli altri prodotti. I costi della copia di un'informazione come un programma software è infinitesimo. L'elettricità non costa quasi nulla, l'uso dell'attrezzatura poco di più. È come, per fare un paragone, se si duplicasse una pagnotta usando un solo etto di farina.

La storia

Il concetto di free software non è nuovo. Quando le università cominciarono ad adottare i computer, essi erano strumenti per la ricerca. Il software veniva scambiato liberamente e i programmatori venivano pagati per l'atto della programmazione, non per i programmi in sé. Solo più tardi, quando il mondo degli affari e del commercio adottò i computer, i programmatori cominciarono a mantenersi limitando i diritti d'uso del loro software e facendosi pagare per ogni copia. Il free software come idea politica è stato reso popolare da Richard Stallman dal 1984, allorché formò la Free Software Foundation e il progetto GNU a essa collegato. La premessa di Stallman è che la gente dovrebbe avere più libertà e dovrebbe imparare ad apprezzarla. Egli progettò un insieme di diritti che sentiva necessari a ogni utente e li codificò nella GNU Public License o GPL. Stallman battezzò scherzosamente la sua licenza copyleft in quanto lasciava intatto il diritto alla copia. Stallman stesso sviluppò lavori fondamentali di free software quali il compilatore C GNU e GNU Emacs, un editor di testo che alcuni hanno trovato così seducente da concepirne quasi un culto. Il suo lavoro ispirò molti altri a fornire free software sotto la GPL. Per quanto non promossa con il medesimo fervore libertario, la Open Source Definition include molte delle idee di Stallman, e può ben considerarsi un derivato della sua opera.

La Open Source Definition cominciò la sua vita come un documento di linea di condotta della distribuzione Debian GNU/Linux. Debian, uno dei primi sistemi Linux, tuttora popolare, fu costruito interamente con free software. Tuttavia, dal momento che c'erano altre licenze oltre al copyleft che comportavano la gratuità, Debian ebbe qualche problema nel definire che cosa fosse gratis, e i produttori non resero mai chiara la loro politica di free software al resto del mondo. All'epoca, trovandomi a capo del progetto Debian, affrontai questi problemi proponendo un Contratto Sociale Debian e la Guida Debian del Free Software, nel luglio del 1997. Molti sviluppatori Debian inviarono critiche e miglioramenti che io incorporai nei documenti. Il Contratto Sociale documentava l'intenzione di Debian di

costituire il proprio sistema interamente con free software, e la Guida rendeva facilmente possibile la classificazione del software come free software o meno, confrontando la licenza software con la guida stessa.

La Guida Debian fu oggetto di molte lodi nella comunità del free software, specialmente fra gli sviluppatori Linux, che a quel tempo stavano preparando la loro propria rivoluzione software sviluppando il primo vero sistema operativo gratuito. Quando Netscape decise di rendere libero il suo browser Web, contattò Eric Raymond. Raymond, la Margaret Mead del free software, è autore di numerosi articoli di taglio antropologico che illustrano il fenomeno del free software e la cultura che vi è cresciuta intorno: scritti che furono i primi di un genere e che hanno messo sotto la luce dei riflettori questo fenomeno fino ad allora oscuro. La dirigenza di Netscape rimase suggestionata in particolare dal saggio di Raymond "La cattedrale e il bazaar", la cronaca di uno sviluppo free software coronato da successo con volontari non pagati, e gli chiese una consulenza, sotto patto di riservatezza, mentre sviluppavano una licenza per il loro free software. Raymond insisté che la licenza di Netscape dovesse adeguarsi alla guida Debian per poter essere presa sul serio come free software.

Raymond e io ci eravamo incontrati qualche volta all'Hacker Conference, una raduno su invito di programmatori creativi e non convenzionali. Avevamo corrisposto via email su vari argomenti. Mi contattò nel febbraio del 1997 con l'idea per l'Open Source. Raymond temeva che la mentalità conservatrice dell'ambiente degli affari venisse scoraggiata dal grado di libertà di Stallman, che era al contrario popolarissimo fra i programmatori di mentalità più liberale. Era impressione di Raymond che ciò stesse sclerotizzando lo sviluppo di Linux nel mondo business laddove esso fioriva invece nell'ambiente della ricerca. Raymond ebbe incontri con uomini d'affari nell'industria Linux che stava muovendo solo allora i primi passi; insieme, essi concepirono un programma di marketing del free software indirizzato ai colletti bianchi. Furono coinvolti Larry Augustin di VA Research e Sam Ockman (che abbandonò più tardi VA per formare Penguin Computing), nonché altri non di mia conoscenza.

Alcuni mesi prima dell'Open Source, avevo elaborato l'idea dell'Open Hardware, concetto simile rivolto agli strumenti hardware e alle loro interfacce anziché ai programmi software. A tutt'oggi l'Open Hardware non ha avuto il successo dell'Open Source, ma il progetto è ancora attivo; se ne può sapere di più a http://www.openhardware.org.

Secondo Raymond, la Guida Debian era il documento più adatto a definire l'Open Source, ma serviva una denominazione più generale e la rimo-

zione dei riferimenti specifici a Debian. Modificai la Guida Debian fino a ricavarne la Open Source Definition. Avevo formato per Debian un ente chiamato Software in the Public Interest, e mi offrii di registrare un marchio per Open Source in modo da poter associare il suo uso alla definizione. Raymond acconsentì, e io registrai una certificazione (una forma speciale di marchio che potesse applicarsi secondo i termini ai prodotti altrui). Circa un mese dopo la registrazione del marchio, apparve chiaro che Software in the Public Interest avrebbe potuto non essere la dimora migliore per il marchio Open Source, e trasferii dunque la proprietà del marchio a Raymond. Raymond e io abbiamo da allora formato la Open Source Initiative, un'organizzazione esclusivamente destinata alla gestione della campagna Open Source e della sua certificazione di marchio. Mentre scrivo, l'iniziativa Open Source è retta da un comitato di sei componenti scelti fra fornitori di free software di chiara fama, e sta cercando di espandere il suo comitato fino a una decina di persone.

Al momento del suo concepimento, la campagna Open Source fu oggetto di molte critiche perfino da parte del contingente Linux che già aveva approvato il concetto di free software. Molti rilevarono che il termine Open Source era già in uso nel ramo della raccolta di dati per le campagne politiche. Altri pensarono che il termine Open fosse già usurato. Per altri ancora era preferibile il nome Free Software, già consolidato. Io opinai che l'abuso del termine Open sarebbe stato sempre meglio dell'ambiguità di free nella lingua inglese, in cui sta a significare tanto libero quanto gratuito, la seconda accezione essendo di gran lunga la più comune nel mondo del commercio di computer e di software. Più tardi, Richard Stallman obiettò alla mancanza di enfasi sulla libertà che secondo lui la campagna dimostrava, e al fatto che, mentre l'Open Source acquistava popolarità, il suo ruolo nella genesi del free software, e quello della sua Free Software Foundation, venivano ignorati: si lamentò di essere stato "cassato dalla storia". Peggiorò la situazione la tendenza degli operatori del settore di contrapporre Raymond a Stallman, quasi essi proponessero filosofie concorrenti anziché, sia pur con metodi diversi, propagandare lo stesso concetto. Io stesso contribuii probabilmente a esacerbare gli animi mettendo Stallman e Raymond l'uno contro l'altro in dibattiti pubblici alla Linux Expo e alla Open Source Expo. Caratterizzare i due come avversari diventò un'attività tanto consueta che una discussione via email, non destinata alla pubblicazione, apparve sul periodico on-line Salon. A quel punto, chiesi a Raymond di moderare i toni di un dialogo in cui, per la verità, egli non aveva mai inteso entrare.

Quando la Open Source Definition fu scritta, esisteva già un gran numero

di prodotti che potevano rientrare nella categoria. Il problema erano quei programmi che non vi rientravano, e che pure gli utenti trovavano irresistibili.

KDE, Qt e Troll Tech

Il caso di KDE, Qt e Troll Tech è pertinente a questo saggio perché il gruppo KDE e Troll Tech cercarono di porre un prodotto non-Open Source entro l'infrastruttura di Linux, incontrando una resistenza inattesa. Le grida di pubblico scandalo e la minaccia che il loro prodotto venisse rimpiazzato da un altro, completamente Open Source, convinse alla fine Troll Tech a convertirsi a una licenza pienamente Open Source. È un segno interessante dell'accoglienza entusiastica riservata dalla comunità alla Open Source Definition il fatto che Troll dovette adeguare la propria licenza, pena l'insuccesso del suo prodotto.

KDE fu il primo esperimento di un desktop grafico gratuito per Linux. Le applicazioni KDE erano esse stesse sotto GPL, ma dipendevano da una libreria grafica proprietaria nota come Qt, di Troll Tech. I termini della licenza di Qt ne proibivano la modifica o l'uso con qualunque display software che non fosse il senescente X Window System. Ogni uso diverso richiedeva allo sviluppatore una licenza del costo di 1500 dollari. Troll Tech fornì versioni di Qt per Windows di Microsoft e per Macintosh, e questa fu la sua principale fonte d'entrate. La licenza pseudo-gratuita per i sistemi X intendeva indirizzare i contributi degli sviluppatori Linux verso demo, esempi e accessori per i loro costosi prodotti Windows e Mac.

Per quanto i problemi della licenza di Qt apparissero evidenti, la prospettiva di un desktop grafico per Linux era così attraente che molti utenti furono disposti a chiudere un occhio sulla sua natura non-Open Source. I promotori di Open Source trovarono che KDE fosse in difetto perché avevano l'impressione che gli sviluppatori stessero cercando di confondere la definizione di free software allo scopo di includervi elementi solo parzialmente gratuiti, come Qt. Gli sviluppatori KDE replicarono che i loro programmi erano Open Source, anche se non esistevano versioni eseguibili di quei programmi che non richiedessero una libreria non-Open Source. Io e altri sostenemmo che le applicazioni KDE non erano che frammenti Open Source di programmi non-Open Source, e che una versione Open Source di Qt sarebbe stata necessaria prima che ci si potesse riferire a KDE come a un Open Source.

Gli sviluppatori KDE tentarono di risolvere parzialmente il problema della licenza di Qt negoziando con Troll Tech un accordo (KDE Free Qt Foundation) in cui Troll e KDE avrebbero congiuntamente controllato i

rilasci delle versioni gratuite di Qt, e Troll Tech avrebbe rilasciato Qt sotto una licenza conforme a Open Source nel caso che l'azienda venisse acquisita o cessasse l'attività.

Un altro gruppo diede inizio al progetto GNOME, un concorrente interamente Open Source di KDE che mirava a fornire maggiori funzioni e sofisticazioni; un gruppo separato avviò il progetto Harmony per produrre un clone di Qt completamente Open Source che avrebbe supportato KDE. Mentre le dimostrazioni di GNOME avvenivano fra il plauso e Harmony stava per diventare usabile, Troll Tech capì che QT non avrebbe riscosso successo nel mondo Linux se non avesse cambiato licenza. Troll Tech rilasciò dunque una licenza interamente Open Source per Qt, disinnescando il conflitto ed eliminando i motivi alla base del progetto Harmony. Il progetto GNOME continua tuttora, volto adesso a un KDE migliore in termini di funzionalità e di raffinatezza piuttosto che in termini di licenza.

Prima di rilasciare la sua nuova licenza Open Source, Troll Tech me ne fece avere copia perché la verificassi, con la preghiera che rimanesse riservata finché non fossero in grado di annunciarla. Nel mio entusiasmo di far pace con il gruppo KDE e in un imbarazzante gesto di autoinganno, preannunciai con otto ore di anticipo la licenza su una mailing list KDE. Quell'email, per il mio rimorso, fu raccolta immediatamente da Slashdot e da altre riviste online.

La nuova licenza Troll Tech è notevole perché approfitta di una scappatoia nella Open Source Definition che permette ai file patch di essere trattati diversamente dall'altro software. Vorrei provvedere a chiudere questa scappatoia in una revisione a venire della Open Source Definition, ma il nuovo testo non dovrebbe tuttavia collocare Qt al di fuori dell'Open Source.

Al momento in cui scrivo, i promotori di Open Source stanno crescendo in misura esponenziale. I recenti contributi Open Source di IBM e di Ericsson hanno fatto i titoli dei giornali. Due distribuzioni Linux, Yggdrasil e Debian, stanno rilasciando distribuzioni di sistemi Linux completi, ivi incluse molte applicazioni che sono interamente Open Source; e molte altre, fra cui Red Hat, ci sono assai vicine. Quando il sistema GNOME sarà completo, sarà stato realizzato un sistema operativo con desktop GUI Open Source in grado di competere con Microsoft NT.

Analisi della Open Source Definition

Questa sezione presenta nella sua interezza il testo della Open Source Definition, corredata di commenti (in corsivo). La versione canonica della Open Source Definition si trova a http://www.opensource.org/osd.html.

Alcuni pedanti hanno voluto trovare delle ambiguità di poco conto nella Open Source Definition. Mi sono astenuto da rivederla dal momento che ha poco più d'un anno di vita e vorrei che il pubblico la considerasse stabile. Il futuro imporrà qualche adeguamento lessicale, ma quasi nessuna modifica allo scopo del documento.

- -

La Open Source Definition

Open Source non significa solo accesso al codice sorgente. I termini di distribuzione di un programma Open Source devono essere consoni ai criteri seguenti:

Si noti che la Open Source Definition non è propriamente una licenza software. È una specifica di quanto è ammesso in una licenza software perché vi si possa riferire come a un'Open Source. La Open Source Definition non è intesa per essere un documento di valore legale. L'inclusione della Open Source Definition nelle licenze software, quale quella proposta per il Progetto di Documentazione di Linux, sembra suggerirmi la stesura di una versione più rigorosa che sia appropriata per quell'uso.

Ai fini dell'Open Source, devono applicarsi insieme tutti i termini che seguono, in tutti i casi. Per esempio, devono applicarsi alle versioni derivate di un programma così come al programma originale. Non è sufficiente applicarne alcune e non altre, e non è sufficiente se i termini non vengono applicati sistematicamente. Dopo aver dovuto affrontare delle interpretazioni particolarmente "semplici" della Open Source Definition, sono tentato di aggiungere: sto dicendo a voi!

1. RIDISTRIBUZIONE LIBERA

La licenza non può impedire ad alcuna parte in causa la vendita o la cessione del software come componente di una distribuzione di software aggregato che contenga programmi proveniente da sorgenti diverse. La licenza non può richiedere diritti o il pagamento di altre concessioni per tale vendita.

Questo significa che potete fare tutte le copie che volete del software e venderle o cederle, e non dovete pagare nessuno per questo privilegio.

L'espressione "distribuzione di software aggregato che contenga programmi provenienti da sorgenti diverse" era intesa a chiudere una scappatoia nella Licenza Artistica - una licenza piuttosto malfatta, a mio parere -

escogitata in origine per il Perl. Oggi, quasi tutti i programmi che usano la Licenza Artistica sono disponibili anche sotto GPL. Quella clausola non è più necessaria e sarà probabilmente tolta da una futura versione della Open Source Definition.

> 2. CODICE SORGENTE
> Il programma deve includere il codice sorgente e deve consentire la distribuzione tanto in codice sorgente che in forma compilata. Laddove una qualunque forma del prodotto non sia distribuita corredata del codice sorgente, devono essere disponibili mezzi ben pubblicizzati per scaricare il codice sorgente, senza costi addizionali, via Internet. Il codice sorgente deve essere la forma preferenziale nella quale un programmatore modifichi un programma. Codice deliberatamente offuscato non è ammesso. Forme intermedie quali l'output di un preprocessore o di un traduttore non sono ammesse.

Il codice sorgente è un preliminare necessario alla riparazione o alla modifica di un programma. L'intento qui è che il codice sorgente sia distribuito con l'opera iniziale e con tutte le opere derivate.

> 3. OPERE DERIVATE
> La licenza deve permettere modifiche e opere derivate e deve consentire la loro distribuzione sotto i medesimi termini della licenza del software originale.

Il software serve a poco se non se ne può fare la manutenzione (riparazione dei bug, porting su nuovi sistemi, migliorie) e la modifica è indispensabile alla manutenzione. L'intento è qui di permettere modifiche d'ogni sorta. Deve essere permessa la distribuzione di un'opera modificata sotto gli stessi termini di licenza dell'opera originale. Tuttavia, non è richiesto che ogni produttore di un'opera derivata debba usare gli stessi termini di licenza, ma solo che possa farlo qualora lo voglia. Diverse licenze si esprimono diversamente in materia: la licenza BSD vi permette di mantenere private le modifiche, la GPL no.

Alcuni autori di software ritengono che questa clausola possa consentire a persone prive di scrupoli di modificare il loro software in maniera che possa causare imbarazzo all'autore originale. Quello che temono è che qualcuno possa deliberatamente provocare un malfunzionamento del software in modo che l'autore originale appaia un programmatore scadente. Altri paventano un possibile uso criminale del software tramite l'aggiunta di funzioni-cavallo di Troia o di tecnologie illegali in alcuni Paesi, come la

crittografia. Tutti questi atti, tuttavia, sono coperti dal codice penale. Un comune fraintendimento a proposito delle licenze è che esse debbano specificare ogni cosa, per esempio "questo software non va usato per compiere delitti". Dovrebbe tuttavia essere chiaro che nessuna licenza ha esistenza valida al di fuori del corpo del diritto civile e penale. Considerare una licenza come qualcosa separato dal corpo delle leggi applicabili è tanto sciocco quanto considerare un documento in lingua inglese separato dal vocabolario di quella lingua, un caso in cui nessuna parola avrebbe un significato definito.

> 4. Integrità del codice sorgente dell'autore
> La licenza può proibire che il codice sorgente venga distribuito in forma modificata solo se la licenza permette la distribuzione di "patch file" con il codice sorgente allo scopo di modificare il programma al momento della costruzione.

Alcuni autori temevano che altri potessero distribuire codice sorgente con modifiche che sarebbero state percepite come opera dell'autore originale e quindi avrebbero potuto gettare ombra su di lui. Questa clausola dà loro un modo di imporre una separazione fra le modifiche e la loro opera, senza proibire le prime. C'è chi considera antiestetico che le modifiche debbano venir distribuite in un file "patch" separato dal codice sorgente, anche se distribuzioni Linux come Debian e Red Hat usano questa procedura per tutte le modifiche apportate ai programmi che distribuiscono. Esistono programmi per riversare automaticamente le patch nel sorgente principale, e questi programmi si possono eseguire automaticamente quando si scompatta un pacchetto di sorgente. Questa clausola, dunque, dovrebbe causare poca o nessuna difficoltà.

Si noti anche che questa clausola dice che, nel caso di file patch, la modifica avviene quando si fa il build del programma. Questa scappatoia è impiegata nella Licenza Pubblica di Qt per prescrivere una diversa, anche se meno restrittiva, licenza per i file patch, in contraddizione con la sezione 3 della Open Source Definition. C'è una proposta per chiudere questa scappatoia nella definizione e mantenere nello stesso tempo Qt entro i confini dell'Open Source.

> La licenza deve permettere esplicitamente la distribuzione di software costruito da codice sorgente modificato. La licenza può richiedere che le opere derivate vadano sotto nome o numero di versione differenti da quelli del software originale.

Questo significa che Netscape, per esempio, può insistere per poter essa sola chiamare una versione del programma Netscape Navigator (tm), mentre tutte le versioni gratuite del programma debbano chiamarsi Mozilla o in altro modo.

5. NESSUNA DISCRIMINAZIONE CONTRO PERSONE O GRUPPI
La licenza non deve discriminare alcuna persona o gruppo di persone.

Una licenza fornita dai Rettori dell'Università della California a Berkeley proibiva l'uso di un programma di progettazione elettronica da parte delle forze di polizia del Sud Africa. Apprezzato come merita questo sentimento in tempi di apartheid, va detto che esso non ha più senso oggi. Alcune persone si trovano ancora con software acquistato sotto quella licenza, e le loro versioni derivate devono portare la stessa restrizione. Le licenze Open Source non devono contenere tale clausola, indipendentemente dalla nobiltà dell'intento.

6. NESSUNA DISCRIMINAZIONE DI SETTORI.
La licenza non deve proibire ad alcuno l'uso del programma in uno specifico campo o per un determinato proposito. Per esempio, non può impedire che il programma venga usato a scopi commerciali o nella ricerca genetica.

Il software dev'essere impiegabile allo stesso modo in una clinica che pratichi aborti e in un'organizzazione antiabortista. Queste discussioni politiche sono di pertinenza del Congresso degli Stati Uniti, non delle licenze del software. Alcuni trovano questa mancanza di discernimento gravemente offensiva!

7. DISTRIBUZIONE DELLA LICENZA
I diritti relativi al programma devono applicarsi a tutti coloro ai quali il programma sia ridistribuito, senza necessità di esecuzione di una licenza aggiuntiva da parte di questi.

La licenza dev'essere automatica, senza la richiesta di alcuna firma. Purtroppo, negli Stati Uniti non ci sono dati validi precedenti giudiziari del potere della licenza senza firma quando questa venga passata da una seconda a una terza parte. Tuttavia, questo argomento considera la licenza come facente parte della legge sul contratto, mentre qualcuno obietta che dovrebbe essere considerata come legge di copyright, campo in cui si danno più prece-

denti per quel tipo di licenza. Un buon precedente ci sarà senz'altro nei prossimi anni, data la popolarità del questa licenza e il boom dell'Open Source.

> 8. La licenza non dev'essere specifica a un prodotto.
> I diritti relativi a un programma non devono dipendere dall'essere il programma parte di una particolare distribuzione software. Se il programma è estratto da quella distribuzione e usato o distribuito entro i termini della licenza del programma stesso, tutte le parti a cui il programma sia ridistribuito dovrebbero avere gli stessi diritti che vengono garantiti in unione alla distribuzione software originale.

Questo significa che non si può impedire a un prodotto identificato come Open Source di essere gratuito solo se lo si usa con una marca particolare di distribuzione Linux, ecc. Deve rimanere gratuito se anche lo si separa dalla distribuzione software da cui proviene.

> 9. La licenza non deve contaminare altro software
> La licenza non deve porre restrizioni ad altro software che sia distribuito insieme a quello licenziato. Per esempio, la licenza non deve pretendere che tutti gli altri programmi distribuiti sullo stesso media siano software Open Source.

Una versione di GhostScript (programma di rendering PostScript) richiede che i media sui quali viene distribuito contengano solo programmi software gratuiti. Questo non è consentito dalla licenza Open Source. Per fortuna, l'autore di GhostScript distribuisce un'altra versione del programma (un po' più vecchia) sotto una licenza Open Source genuina.

Si noti che c'è differenza fra derivazione e aggregazione. Derivazione è quando un programma incorpora di fatto in sé parti di un altro programma. Aggregazione è quando due programmi vengono inclusi sullo stesso CD-ROM. Questa sezione della Open Source Definition riguarda l'aggregazione, non la derivazione. La sezione 4 riguarda la derivazione.

> 10. Licenze esemplari
> Le licenze GNU GPL, BSD, X Consortium e Artistica sono esempi di licenze da considerarsi conformi alla Open Source Definition. Altrettanto dicasi della MPL. *[vedi nota alla fine del saggio]*

Questo sarebbe una fonte di guai nel giorno in cui una di queste licenze si modificasse e non fosse più Open Source: dovremmo pubblicare immediatamente una revisione della Open Source Definition. Ciò è pertinente

per la verità al testo esplicativo, non alla Open Source Definition in sé.

- -

Analisi delle licenze e loro conformità all'Open Source

Per comprendere la Open Source Definition dobbiamo esaminare alcune pratiche comuni nelle licenze in quanto si riferiscono all'Open Source.

Public Domain

La diffusa convinzione che molto del free software sia di dominio pubblico è errata. Ciò avviene perché l'idea di free software o Open Source confonde molti, che quindi definiscono erroneamente questi programmi come di pubblico dominio perché è il concetto più prossimo a quanto è loro familiare. I programmi, tuttavia, sono molto chiaramente protetti da diritti e sottoposti a licenza: solo, si tratta di una licenza che dà al pubblico più diritti di quelli a cui sia abituato.

Un programma di pubblico dominio è un programma sul quale l'autore abbia rinunciato a tutti di suoi diritti di copyright. Non si può esattamente dire che sia dotato di una licenza; è proprietà personale, per usarlo come meglio si crede. Dal momento che si può trattarlo come personale proprietà, con un programma di pubblico dominio si può fare quello che si vuole. Si può perfino ri-licenziare un programma di pubblico dominio, rimuovendo quella versione dal pubblico dominio, o togliendo il nome del suo autore e trattarlo come opera propria.

Se si sta spendendo molto lavoro su un programma di pubblico dominio, si consideri la possibilità di applicarvi il proprio copyright e di rimetterlo sotto licenza. Per esempio, se non si desidera che una terza parte operi delle modifiche che possa poi mantenere riservate, si applichi la GPL o una licenza simile alla propria versione del programma. La versione da cui si è partiti rimarrà nel pubblico dominio, ma la propria versione sarà sotto una licenza che dovrà essere osservata da chi la usa o ne derivi altre.

Un programma di pubblico dominio si rende privato facilmente, dichiarando un copyright e applicandovi la propria licenza, oppure semplicemente dichiarando "Tutti i diritti riservati".

Le licenze Free Software in generale

Se si ha una raccolta di free software come un disco Linux, si potrebbe credere che il programma su quel disco sia proprio. Ma questo non è del tutto vero. I programmi coperti da copyright sono proprietà di chi detiene

il copyright, anche quando arrivano con una licenza Open Source come la GPL. La licenza del programma garantisce alcuni diritti, e altri si hanno sotto la definizione di uso corretto nella legge sul copyright.

È importante notare che un autore non deve necessariamente limitarsi a porre una sola licenza su un programma che pubblica. Un programma può essere posto sotto GPL, e una versione può anche essere venduta con una licenza commerciale, non-Open Source. Proprio di questa strategia si valgono molti che desiderano creare un programma Open Source e allo stesso tempo guadagnarci qualche cosa. Chi non vuole una licenza Open Source può pagare per il privilegio, fornendo all'autore una fonte d'entrate.

Tutte le licenze che esamineremo hanno una caratteristica comune: declinano qualunque garanzia. Lo scopo è quello di proteggere il proprietario del software da qualunque responsabilità connessa al programma. Appare una richiesta ragionevole, dato che il programma viene ceduto a costo zero: l'autore non riceve dal programma una fonte d'entrata sufficiente per sostenere un'assicurazione sulle responsabilità ed eventuali spese legali.

Se gli autori di free software perdessero il diritto di declinare tutte le garanzie e si trovassero a essere citati in tribunale in base alle prestazioni dei programmi che hanno scritto, smetterebbero di fornire software gratuito al mondo. È nel nostro interesse di utenti aiutare gli autori a proteggere questo diritto.

La GNU General Public License

Si veda l'Appendice B per il testo completo della GPL. La GPL è un manifesto politico tanto quanto è una licenza software, e la maggior parte del testo è inteso a spiegare la motivazione teorica dietro la licenza. Questo dibattito politico ha allontanato alcuni e fornito alcune delle ragioni per cui sono state scritte altre licenze per il free software. Tuttavia, la GPL è stata stilata con l'assistenza di giuristi ed è per questo assai meglio scritta della maggior parte delle licenza di quella famiglia. Io consiglio caldamente di usare la GPL, o la sua variante per librerie LGPL, ogni volta che sia possibile. Sesi sceglie un'altra licenza, o se ne stila una nuova, ci devono essere delle buone ragioni per farlo. Chi formula la propria licenza dovrebbe sapere bene che non e un passo da fare con leggerezza. Le complicazioni inaspettate di una licenza affrettata possono affliggere gli utenti di un software per molti anni a venire.

Il testo della. GPL non e a sua volta sotto GPL. La sua licenza è semplice: Chiunque può copiare e distribuire copie esatte di questo documento di licenza, ma non ne sono ammesse modifiche. Un punto importante, qui, è

che il testo delle licenze di software Open Source di solito non è Open Source esso stesso. Ovviamente, una. licenza non porrebbe offrire protezione di alcun tipo se a chiunque fosse consentito apportarvi delle modifiche.

Le clausole della GPL soddisfano la Open Source Definition. La GPL non richiede alcuna delle clausole consentite dal Paragrafo 4 della Open Source Definition. Integrità del codice sorgente dell'autore.

La GPL non permette di mantenere private le modifiche apportate. Le modifiche devono essere distribuite sotto la GPL. In questo modo, l'autore di un programma sotto GPL ha maggiori probabilità di ricevere modifiche da altri, comprese società commerciali che modificano il suo software per i propri scopi.

La GPL non ammette l'incorporazione di un programma sotto GPL in un programma proprietario. La definizione di GPL di programma proprietario lo indica come ogni programma con una licenza che non dia tanti diritti quanti la GPL.

Esistono alcune scappatoie nella GPL che permettono di usarla in un programma non interamente Open Source. Le librerie software che vengono normalmente distribuite con il compilatore o con il sistema operativo che si usa possono essere collegate a software GPL: ne risulta un programma parzialmente libero. Chi detiene il copyright (di norma l'autore del programma) e la persona che mette il programma sotto GPL e ha il diritto di violare la propria licenza. Questa scappatoia e stata usata dagli autori di KDE per distribuire il loro programma Qt prima che Troll Tech ponesse su Qt una licenza Open Source. Tuttavia, questo diritto non si estende ad alcuna terza parte che ridistribuisca il programma: esse devono seguire tutti i termini della licenza, anche quelli che vengono violati dal detentore del copyright, il che rende problematico ridistribuire un programma che contenga Qt sotto GPL. Gli sviluppatori KDE sembrano inclini a rimediare a questo problema applicando al loro software la LGPL piuttosto che la GPL.

La retorica politica presente nella GPL non e gradita a tutti. Non manca chi ha scelto, per il suo software, licenze non altrettanto adatte per semplice avversione alle idee di Richard Stallmann, pur di non aver voluto vederle ripetute nei propri pacchetti software.

La GNU Library Public License

La LGPL è una derivazione della GPL escogitato per le librerie software. A differenza della GPL, un programma sotto LGPL può venire incorporato entro un programma proprietario. La libreria di linguaggio C fornita con i sistemi Linux e un esempio di software sotto LGPL: essa può essere

usata per costruire programmi proprietari, diversamente Linux risulterebbe utile solamente agli autori di free software.

Una copia di un programma sotto LGPL può essere convertita in qualunque momento in una sotto GPL. Una volta che ciò succede, quella copia non e più riconvertibile in un programma sotto LGPL, e altrettanto dicasi di qualunque suo derivato.

Le rimanenti clausole della LGPL sono simili a quelle della GPL: di fatto essa include la GPL facendovi riferimento.

Le licenze X, BSD e Apache

La licenza. X e le sue affini BSD e Apache sono molto diverse dalla. GPL e dalla LGPL. Queste licenze consentono di fare quasi tutto ciò che si vuole con il software 'licenziato' sotto di esse, e questo perché il software originariamente coperto dalle licenze X e BSD era sovvenzionato con sussidi del Governo degli Stati Uniti. Dal momento che i cittadini statunitensi avevano già pagato il software con i soldi delle tasse, fu loro garantito il diritto di fare del software tutto ciò che volessero.

La concessione più importante, assente dalla GPL, e che si può mantenere private le modifiche licenziate sotto licenza X. In altre parole, si può ottenere il codice sorgente di un programma sotto X, modificarlo e poi vendere versioni binarie del programma senza distribuire il codice sorgente delle modifiche e senza applicarvi la licenza X. Tutto ciò rimane comunque Open Source, poiché la Open Source Definicion non richiede che le modifiche debbano sempre ricadere sotto la licenza originale.

Molti altri sviluppatori hanno adottato la licenza X e le sue varianti, compresi i progetti BSD (Berkeley System Distribution) e Apache Web server. Un dettaglio molesto della licenza BSD è costituito da una clausola che prescrive che ogni volta si faccia cenno a una caratteristica di un programma sotto BSD in una sua pubblicità, si menzioni (generalmente in una nota a pie di pagina) il fatto che il software è stato sviluppato all'Università della California. Ora, tener traccia di quale software abbia quella licenza in una cosa immensa come una distribuzione Linux, e ricordare quindi di menzionare l'Università della California ogni volta che uno di questi programmi venga citato in una pubblicità, e un vero mal di testa per i gestori commerciali del progetto, Nel momento in cui scrivo, la distribuzione Debian GNU/Linux contiene oltre 2500 pacchetti software, e se anche solo una piccola parte di essi fosse sotto BSD, la pubblicità per un sistema Linux come Debian dovrebbe contenere molte pagine solo di note! Tuttavia, la licenza dell'X Consortium non ha quella clausola della pubblicità. Se si pensa di

usare una licenza tipo BSD si usi invece una licenza X.

La Licenza Artistica

Sebbene questa licenza sia stata in origine sviluppata per il Perl, e stata dopo allora adoperata per altro software. A mio parere si tratta di una licenza formulata con grande sciattezza, in quanto impone dei requisiti e fornisce poi delle scappatoie che rendono facile aggirarli. Forse e questa la ragione per cui quasi tutto il software sotto Licenza Artistica, ha oggi una seconda licenza, offrendo la scelta fra la Licenza Artistica e la GPL.

La. Sezione 5 della Licenza Artistica vieta la vendita del software, ma permette che sia venduta una distribuzione di software aggregato di più di un programma. In questo modo, se raggruppate un programma sotto Licenza Artistica con un Helloworld.c di cinque righe di codice, potete vendere il bundle. Questa caratteristica della Licenza Artistica e stata la sola causa della scappatoia dell'"aggregato" nel primo paragrafo della Open Source Definition. Dal momento che l'uso della Licenza Artistica e in netto declino, stiamo pensando di cogliere quella scappatoia. Ciò renderebbe la Licenza Artistica una licenza non-Open Source. Non è questo un passo che faremo leggermente, e ci vorrà probabilmente più di un anno di riflessione e di dibattito prima che questo accada,

La Licenza Artistica richiede che le modifiche siano rese gratuite, ma fornisce poi una scappatoia (nella Sezione 7) che permette di mantenerle private e perfino di porre sotto dominio pubblico parti del programma sotto Licenza Artistica!

La Netscape Public License e la Mozilla Public License

La NPL e stata sviluppata da Netscape quando rese Open Source il suo prodotto Netscape Navigator. Per la precisione, la versione Open Source si chiama Moizilla; Netscape si riserva il marchio Navigator per il suo prodotto. Eric Raymond ed io agimmo come consulenti a titolo gratuito durante lo sviluppo di questa licenza. Io cercai, senza. successo, di persuadere Netscape a usare la GPL, e quando essa declinò, contribuii a comporre una licenza che si conformasse alla Open Source Definifion.

Una. caratteristica importante della. NPL è che contiene privilegi speciali che si applicano a Netscape e a nessun altro. Essa da a Netscape il privilegio di rilicenziare le modifiche fatte al suo software. Netscape può mantenere private quelle modifiche, migliorarle, e rifiutarsi di restituire il risultato. Questa clausola si e resa necessaria perché, quando Netscape decise per l'Open Source, aveva contratti con altre aziende che la impegnavano a

fornir loro Navigator sotto una licenza non Open Source.

Netscape ha. creato la MPL o Mozilla Public License per rimediare a questa situazione. La MPL e molto simile alla NPL, ma non contiene la clausola che permette a Netscape di rimettere le modifiche sotto licenza.

La NPL e la MPL consentono di mantenere private le modifiche apportate.

Molte aziende hanno adottato per i loro programmi una variante della MPL. Non e una buona cosa, perché la NPL era stata progettata per la particolare situazione contrattuale in cui Netscape si trovava. nel momento in cui la licenza veniva scritta, e non e detto che sia altrettanto adatta a usi diversi. Dovrebbe restare la licenza di Netscape e di Mozilla, e altri dovrebbero usare le licenze GPL o X.

Scegliere una licenza

Non conviene formulare una licenza nuova se e possibile usarne una di quelle qui elencate. La propagazione di molte licenze diverse e incompatibili opera a detrimento del software Open Source, perché frammenti di un programma non possono essere usati in un altro programma sotto licenza incompatibile.

Ci si tenga alla larga dalla. Licenza Artistica, a meno che non si intenda studiarla fondo ed eliminarne le scappatoie. Fatto ciò, è tempo di prendere delle decisioni.

1. Si vuole che il pubblico possa. mantenere private le modifiche, o no? Se vuole che chi ha apportato modifiche al proprio software ne rimandi il codice sorgente, si applichi una licenza che lo prescriva. La GPL e la LGPL sono delle buone scelte. Se non dispiace che il pubblico mantenga private le modifiche, si usino la licenza X o la licenza Apache.

2. Si vuole consentire a qualcuno di far confluire il proprio programma nel suo software proprietario? Se si, si usi la LGPL, che lo permette esplicitamente senza consentire al pubblico di rendere privato il codice, oppure si usi le licenza X o Apache, che permettono che le modifiche siano mantenute private.

3. Si desidera che chi lo voglia possa comprare sotto licenza commerciale versioni non Open Source del proprio programma? Se si, si doti il software di doppia licenza. Io consiglio la GPL come licenza Open Source; si può trovare una licenza commerciale adatta all'uso in libri come "Copyright Your Software" ediito da Nolo Press.

4. Si vuole che chiunque usi il proprio software debba pagare per il privilegio? Se le cose stanno cosi, forse l'Open Source non e adatta. Se basta

che solo alcune persone paghino, si può mantenere Open Source il programma. La maggior parte degli autori Open Source considerano i loro programmi come contributi al bene pubblico, e non badano al fatto di essere pagati oppure no.

Questa che segue è una tabella comparativa delle licenze pubbliche:

Licenze	*Può essere miscelato con software commerciale?*	*Le modifiche possono essere mantenute private e non restituite all'autore originale?*	*Può essere ri-licenziato da chiunque?*	*Contiene privilegi speciali sulle modifiche per chi detiene il copyright originale?*
GPL	no	no	no	no
LGPL	**sì**	no	no	no
BSD	**sì**	**sì**	no	no
NPL	**sì**	**sì**	no	**sì**
MPL	**sì**	**sì**	no	no
Dominio Pubblico	**sì**	**sì**	**sì**	no

Il Futuro

Al momento in cui questo saggio andava in stampa, IBM e entrava nel mondo Open Source e la. comunit;à dei venture capital lo sta scoprendo. Intel e Netscape hanno investito in Red Hat, un distributore Linux. VA Research, integratore di server Linux e hardware per workstation, ha annunciato l'ingresso di un investitore esterno. Sendmail Inc., creata per commercializzare l'onnipresente programma di posta elettronica Sendmail, ha annunciato la disponibilità di fondi per sei milioni di dollari. L'applicazione di posta protetta Postfix di IBM ha una licenza Open Source, e un altro prodotto IBM, il compilatore Java Jikes, ha una licenza che, nell'istante in cui scrivo, mira, per il momento con parziale successo, a soddisfare le specifiche dell'Open Source Definition. Parrebbe che IBM intenda modificare la licenza di Jikes perché sia per intero Open Source, e che a questo scopo stia raccogliendo pareri nella comunità.

Due promemoria interni della Microsoft, noti sono il nome di Halloween Documents, sono trapelati al pubblico online. Questi promemoria mostrano in modo inequivocabile come Microsoft si senta minacciata da Open Source e da Linux, e che MS lancerà un'offensiva contro di loro per proteggere i suoi mercati. E' chiaro che dobbiamo prepararci a vivere tempi interessanti. Credo che vedremo Microsof usare due principali strategie: interfacce sotto copyright e brevetti. Microsoft esrenderà i protocolli di

rete, che contengono caratteristiche proprietarie Microsoft in quelli che non verranno resi disponibili al free software. Essa, con altre aziende, farà ricerca aggressivamente in nuove direzioni dell'informatica e brevetterà tutto quanto potrà prima che noi si possa. sfruttare quelle tecniche nel free software; quindi ci chiuderà fuori con le concessioni sui diritti di brevetto. Sono autore di un saggio per la webzine Linux World su come si possano battere i nemici dell'Open Source sul fronte dei brevetti.

La buona notizia e che Microsoft i spaventata! Nel secondo degli Halloween Documents, un membro dello staff Microsoft racconta della sua sensazione d'euforia nel vedere come poteva modificare facilmente parti del sistema Linux perché facesse esattamente quello che voleva, e com'era più facile per un impiegato Microsoft fare questo su Linux di quanto non lo fosse modificare NT.

I tentativi di nuocerci provenienti dall'interno sono i più pericolosi. Credo che vedremo altri sforzi per diluire la definizione di Open Source fino a includervi prodotti parzialmente gratuiti, come abbiamo visto avvenire con la libreria Qt in KDE prima che Troll tech vedesse la luce e rilasciasse una licenza Open Source. Microsoft e altri potrebbero danneggiarci rilasciando un sacco di software gratuito quel tanto da attrarre utenti, ma senza avere le piene libertà dell'Open Source. Non è impensabile che essi possano stronca.re lo sviluppo di certe categorie di software Open Source rilasciando soluzioni "abbastanza valide", "abbastanza quasi-gratis". Tuttavia, la forte reazione che si e avuta contro il progetto KDE prima che la libreria Qt divenisse completamente Open Source, non è di buon augurio per imprese analoghe di MS e compagnia.

Finora abbiamo scampato i cavalli di Troia. Supponiamo che qualcuno che ci vuol male fornisca del software che contiene un cavallo di Troia un espediente per sconfiggere la protezione in un sistema Linux. Supponiamo, poi, che la medesima persona resti in attesa che il software con il cavallo di Troia sia largamente distribuito e quindi ne pubblicizzi la vulnerabilità agli attacchi alla sicurezza. Il pubblico si sarà per allora accorto che il nostro sistema Open Source può lasciarci più vulnerabili a questa sorta di attacchi che non il sistema chiuso di Microsoft; questo potrebbe ridurre la fiducia generale nel software Open Source. Potremmo obiettare che Microsoft ha la sua parte di bug di sicurezza anche se non lascia possibilità di inserirli a persone esterne e che il modello a codice sorgente aperto dell'Open Source rende più facile scoprire questi bug. Qualunque bug del genere che comparisse in Linux sarebbe riparato il giorno dopo essere stato scoperto, mentre un omologo in Windows o non sarebbe mai scoperto o dovrebbe aspettare

il rimedio per anni. Ma dobbiamo rinforzare ancora la nostra difesa contro i cavalli di Troia. Identificare con sicurezza chi contribuisce alla creazione di software e delle modifiche e la difesa migliore di cui disponiamo, dal momento che ci permette di valerci del diritto penale contro chi escogita cavalli di Troia. Quando ero dirigente della distribuzione GNU/Linux di Debian, istituimmo un sistema che consentiva di identificare in modo affidabile tutti i manutentori del software e permetteva loro di partecipare a loro volta a una rete a crittografia a chiave pubblica che ci avrebbe consentito di verificare da chi proveniva il nostro software. Questo tipo di sistema si deve espandere fino a comprendere tutti gli sviluppatori Open Source.

Enormi sono i miglioramenti da intraprendere prima che Linux sia davvero alla. portata dell'utente medio. L'interfaccia grafica per gli utenti e chiaramente qualcosa che manca, e a questo sono rivolti i progetti KDE e GNOME. La prossima frontiera è l'amministrazione di sistema linusconf vi sta parzialmente provvedendo, ma si trova ben lungi dall'essere uno strumento completo d'amministrazione di sistema per l'utente sprovveduto. Se il sistema COAS di Caldera avrà successo, potrebbe diventare la base per una soluzione completa al problema dell'amministrazione di sistema. Tuttavia, Caldera ha avuto dei problemi nel mantenere un'allocazione di risorse sufficienti a COAS per terminarne lo sviluppo, e altri sviluppatori hanno abbandonato la partita perché non notavano progressi.

La pletora di distribuzioni Linux appare oggi in pieno rivolgimento, con Red Hat percepita come vincitrice e Caldera come seconda. Red Hat ha mostrato finora un solido impegno verso il concetto di Open Source, ma un nuovo presidente e voci di un'offerta pubblica iniziale (Initial Pubblic Offering, IPO) potrebbero significare un indebolimento di quest'impegno, specialmente se concorrenti come Caldera, molto più tiepidi verso l'Open Source, riusciranno a inserirsi nel mercati di Red Hat. Se l'impegno delle distribuzioni Linux commerciali verso l'Open Source diventerà problematico, questo genererà probabilmente uno sforzo per rimpiazzarle con tentativi Open Source simili al GNU/Linux di Debian ma più diretti al mercato commerciale di quanto non sia stata Debian.

Malgrado queste sfide, io predico la vittoria dell'Open Source. Linux è divenuto strumento di test per gli studenti d'informatica, che, una volta laureati, porteranno con sé quegli strumenti nei loro posti di lavoro. Molti laboratori di ricerca hanno adottato il modello Open Source in quanto la condivisione delle informazioni è essenziale al metodo scientifico, e l'Open Source consente al software di essere condiviso facilmente. Il mondo business sta adottando il modello Open Source perché consente a gruppi di

aziende di collaborare nella risoluzione di un problema senza la minaccia di una causa anti-trust, e per l'impulso di cui gode quando i contributi pubblici di programmazione rendono gratuite le migliorie al software. Alcune gran- di società hanno adottato l'Open Source come strategia per combattere Microsoft e per scongiurare l'avvento di un'altra. Microsoft a dominare il settore informatico. Ma l'indicazione più affidabile sul futuro dell'Open Source viene dal suo passato: in pochi anni, dal niente siamo arrivati ad avere un robusto corpus di software che è in grado di risolvere tanti problemi diversi e che si avvia a raggiungere il milione di utenti. Non c'e ragione di rallentare la corsa. Proprio adesso.

NOTA:

La sezione 10 è quella che più di tutte ha creato problemi, come ho già spiegato nel mio libro “Copyleft & opencontent”, di cui si riporta un passo:

> [...] Un discorso a parte merita invece la ’Sezione 10’, recentemente novellata per intero a causa dei problemi di poca lungimiranza ed elasticità che il suo dettato denotava. Nella versione originale essa faceva cenno ad alcune "licenze esemplari" da considerarsi modelli conformi alla OSD, che erano la GNU GPL, la BSD, la X Consortium e la Licenza Artistica; questa elencazione avrebbe però creato problemi di interpretazione della OSD nel caso (nemmeno molto improbabile) che una di queste fosse stata modificata così da risultare invece incompatibile col concetto di Open Source. Si è deciso così di eliminare ogni richiamo preciso ad alcune licenze e di sostituirlo con una nuova sezione del tutto diversa ma stavolta molto lungimirante e scaltra: s’introduce il concetto (finora non ufficialmente contemplato dai "manifesti" del movimento Opensource) della neutralità della tecnologia. Si proibisce infatti di usare la licenza di un software Open Source per creare eventuali privilegi in ambito hardware; la libertà del software diventa quindi un by-pass per toccare un altro tema scottante: le implicazioni col diritto industriale fra hardware, software e disciplina antitrust. Infatti, per il già citato fenomeno delle network externalities, i diritti di proprietà intellettuale possono avere ’effetti di rete’ limitativi della libertà di scelta dell.utente. Questa dunque la breve enunciazione della ’Sezione 10’: "**La licenza dev’essere tecnologicamente neutrale. Nessuna condizione della licenza può essere prevista per qualche particolare tecnologia o tipo di interfaccia**."
> [cfr. ALIPRANDI, *Copyleft & opencontent - l’altra faccia del copyright*, PrimaOra, marzo 2005; www.copyleft-italia.it/libro]

Una versione completa, aggiornata e tradotta in Italiano della Open Source Definition è riportata in appendice al libro "Copyleft & opencontent" oppure alla pagina web www.copyleft-italia.it/documenti.

Al sito www.opensource.org è disponibile anche una cronologia delle modifiche che il documento ha subito nel corso degli anni.

La cattedrale e il bazar
(di Massimo Carboni)

Oltre a Bruce Perens, l'altro artefice della Open Source Initiative è Eric Raymond il quale in un suo famoso saggio "The cathedral and the bazar" esponeva - usando una sagace metafora - quali fossero secondo lui le differenze di fondo dei due 'modus operandi': quello del Progetto GNU e del free software ("la cattedrale") contro quello di Linux e dell'Open Source ("il bazar"). In questo articolo si dà un'idea di massima del pensiero di Raymond; si consiglia tuttavia la lettura del suo saggio, accessibile anche dalla pagina web www.copyleft-italia.it/documenti. [S. Aliprandi]

"Ero in ascensore, in occasione della manifestazione Agenda 2000. D'un tratto sale un tipo azzimato, con l'aria un po' tronfia. Era Craig Mundie, vicepresidente Microsoft, tuttavia io non lo sapevo, non lo avevo mai incontrato prima anche se istintivamente sentivo che era uno dei tanti 'soldatini' di una grossa corporate. Lo guardo e non riesco a resistere dal chiedergli 'Scusi lei è della Microsoft?'. Quello mi guarda di sottecchi, e con aria di sufficienza mi risponde 'Sì… e lei cosa fa nella vita?'. Lo disse come se stesse rivolgendosi ad una persona di basso rango, mi diede fasti-

dio, 'Guarda com'è pieno di sé questo qui', pensai. Ricambiai subito lo sguardo e dissi con aria solenne: 'Io… sono il tuo peggiore incubo!'".

Ride divertito Eric Raymond mentre rammenta questo divertente aneddoto che ha fatto un po' il giro del mondo e rappresenta, insieme al suo libro "La Cattedrale e il bazar", il suo manifesto.

Ma Eric Raymond è, soprattutto, una delle figure di spicco dell'Open Source, uno che ha voluto permeare il movimento di profonde venature filosofiche e teoriche, ipotizzando modelli di sviluppo davvero alternativi.

"La Cattedrale e il Bazar è molto di più di un semplice saggio. E' un'analisi antropologica delle cause che hanno consentito lo sviluppo e il successo dell'Open Source. Uno studio dei processi che hanno portato al successo del free software seguendo strade che sono contrarie a tutti i principi dell'ingegneria informatica. Nel libro metto a confronto due stili ben diversi. Il primo è il classico stile di sviluppo chiuso che definisco 'Cattedrale'. Questo stile è caratterizzato da rigide specificazioni degli obiettivi e da piccoli gruppi di sviluppo del progetto gestiti in modo autoritario e gerarchico. Qui, tra una release e l'altra trascorrono lunghi intervalli di tempo. Dall'altro lato c'è quello che secondo me accade nel mondo Linux, cioè una struttura decentralizzata basata su rapporti paritari, collaborativi e tutti allo stesso livello come avviene in un bazar (da noi sarebbe un mercato) e in cui l'intervallo di tempo tra un release e l'altra è nettamente più breve grazie alle continue sollecitazioni di persone che sono estranee al progetto e che portano ad un continuo lavoro di revisione e ottimizzazione del codice.

Questo sviluppo indipendente, di tipo paritario, determinato dagli input di molti e non coordinato gerarchicamente, rappresenta, per me proprio il motivo del successo dell'Open Source."

I 9 "comandamenti"

Lo stesso Raymond è stato assieme a Perens l'ideatore della "Definizione di Open Source", ovvero delle 9 linee guida che racchiudono un po' tutta la filosofia di questa complessa realtà.

"Fin dall'inizio pensammo di avere bisogno di una definizione, di un specie di meta-licenza che definisse il termine di Open Source. Elaborammo così il documento Open Source Definition che si ispira alle Debian Free Software Guideline originariamente scritte da Bruce Perens. La 'definizione di Open Source' implica i famosi nove punti che rappresentano i postulati imprescindibili del software aperto".

Raymond è soprattutto un filosofo, uno che filtra il vissuto attraverso

una personalissima visione della vita, non solo quella del ristretto ambito, informatico. Rispetto al "nemico" Microsoft pare non prendere mai una posizione di aperto conflitto, come se fosse consapevole che una guerra aperta, un'ostilità manifesta fosse il miglior modo per far apparire Microsoft come una specie di icona ingiustamente perseguitata. Invece della sciabola, preferisce il fioretto, che usa con arguzia a maestria.

"Paradossalmente Microsoft ha usato Linux come arma di difesa, specie all'inizio, per poter affermare, nei processi di antitrust, che esisteva un'alternativa al software Microsoft, e questo impediva, di fatto, di parlare di monopolio della società di Bill Gates proprio perché l'esistenza di quest'altro software poteva scalzarla dal primato acquisito (risatina). Naturalmente il giudice non l'ha bevuta e se fossi uno della Microsoft mi augurerei che i miei ingegneri siano effettivamente più realisti e concreti di quanto non lo siano stati gli avvocati difensori del buon Bill."

Comunista a chi?

Naturalmente anche Raymond è spesso rimasto vittima del retaggio intellettuale che vuole il software aperto come una cosa politicamente a "sinistra".

*"Quando sento che l'Open Source è 'di sinistra' vado davvero su tutte le furie. La politica non c'entra un c***o, si tratta di una scelta improntata ai più sani principi della libertà. Se condividere qualcosa è comunista, allora credo la maggior parte delle persone lo siano…"*

- -

Intervista rilasciata in occasione del "Python UK conference 2003"

L'ultima volta che ha manoscritto una lettera reale era...
Oh… intorno al 1975, almeno credo di ricordare…

Per cosa nutre una sincera antipatia?
La stupidità, a qualunque livello. Odio la televisione, la musica commerciale e i dolci. Trovo deprecabile il vittimismo…

Quando ha capito di aver intrapreso la giusta carriera?
Quando, con un certo stupore, ho cominciato a constatare che mi pagavano per quello che facevo.

Gli artisti dovrebbero sempre...
Ricordarsi che se non riescono a raggiungere un pubblico, fanno solo della "masturbazione intellettuale". Giusto?

La cosa più bella che le è capitata nella vita?
Mia moglie Catherine.

Hobby?
Amo le piante. Ho un pollice verde. Sì, lo so che è dispari in un hacker…

Non potrei vivere senza...
Um. Alimento? Acqua? Ossigeno?

In quale sport andava peggio a scuola?
In tutti…

Qual è la cosa più importante al di fuori del mondo del software?
Anche qui la Libertà…

NB: Parte del materiale di questo articolo è stato liberamente tratto dal libro (con incluso CD) Revolution OS della Apogeo (www.apogeonline.com).

Parte quarta
LINUX

Nella pagina precedente:
Tux, il pinguino simbolo di tutto ciò che ruota attorno al mondo Linux

L’alternativa vincente
(di Marco Biagiotti)

Ero in cerca di un articolo che presentasse in modo chiaro, sintetico ed efficace l’innovazione portata nel mondo dell’informatica dalla diffusione dei sistemi GNU/Linux e l’ho trovato sfogliando una rivista con cui ho collaborato per un certo periodo. Si tratta del mensile Open Source, che per una coraggiosa e illuminata scelta editoriale è stata pubblicata interamente sotto licenza GNU FDL. Ciò mi ha permesso di riportare questo articolo e di ridonargli una nuova vita con la visibilità che a mio avviso merita, dimostrando in concreto la forza dell’opencontent. [S. Aliprandi]

Alla ricerca di un'alternativa

La comunità hacker alla fine degli anni '80, nonostante i progressi delle filosofie GNU e software libero (http://www.gnu.org) Richard M. Stallman (http://www.stallman.org), era abbastanza scettica sulla diffusione futura di buoni programmi e di validi sistemi operativi: i tentativi di commercializzazione su larga scala di un prodotto efficiente quale UNIX erano stati resi vani, oltre che dall'elevato costo dell'hardware necessario e del prodotto stesso, anche a causa delle molteplici versioni proprietarie esistenti.

Ma quello che più risultava negativamente stupefacente era la difficoltà di Unix, tecnologicamente superiore sotto tutti gli aspetti a MS-DOS, di non riuscire a contrastare anche negli ambienti professionali, dove i soldi per l'acquisto di hardware c'erano, l'opera di Microsoft (http://www.microsoft.com). Tanto di cappello agli uomini di Bill Gates (http://www.microsoft.com/billgates/) che sono riusciti sicuramente a rappresentare uno caso di marketing da prendere, almeno in parte, come esempio: cosa che non è stata fatta, evidentemente, dai vari rappresentanti delle società e/o organizzazioni detentrici delle molteplici versioni di Unix.

Eric S. Raymond (http://www.catb.org/~esr) descrive quei tempi con una frase che sintetizza i concetti appena espressi nel modo migliore: “I detentori di Unix proprietario diedero prova di tanta lentezza e inettitudine nel campo del marketing, che Microsoft fu in grado di inglobare la maggior parte della loro fetta di mercato con la tecnologia del sistema operativo Windows, incredibilmente inferiore a quella Unix.”.

L'orientamento puramente tecnico dei detentori di Unix e la scarsa considerazione del marketing da parte degli stessi rappresentava un limite enorme: concentrarsi su operazioni di marketing non vuol dire, come alcuni credevano e credono tuttora, di cercare di arricchirsi vendendo prodotti qualitativamente inferiori a prezzi elevati, significa saper evidenziarne le qualità nel modo giusto ed al momento giusto; di far percepire i vantaggi insiti nel prodotto o progetto stesso.

Nei primi anni novanta questa mancanza di attenzione alla comunicazione del prodotto rischiava di far naufragare definitivamente il progetto a vantaggio di chi, invece, aveva capito quanto in quel momento fosse più importante puntare sul marketing che sulla qualità tecnica vera e propria non ancora percepibile dalla gran parte degli utilizzatori di computer.

Questa osservazione non vuol certo essere interpretata come "meglio pensare a vendere che a sviluppare un buon prodotto", tutt'altro; ma sicuramente come "visto che ho sviluppato un prodotto superiore a tutti gli altri perché non provo a diffonderlo come si deve?"; sembra scontato osservare che c'è stato chi pur non avendo sviluppato un sistema superiore abbia deciso (e ci sia riuscito!) a diffonderlo come si deve...

In questa situazione di rischio di “regresso informatico”, molti hacker si chiusero nelle proprie stanze a programmare per puro divertimento, altri indossarono gli abiti blu della grande sorella del software dominata da uno che almeno in teoria era stato fino a pochi anni prima uno come loro; si noti: “almeno in teoria”. L'orizzonte era semplicemente questo: industria del software voleva dire Microsoft e non c'erano soluzioni di rilievo in

grado di rappresentare una valida alternativa ai suoi prodotti almeno in alcuni specifici settori di applicazione; tra l'altro le alternative a Microsoft erano costituite dalle costosissime versioni proprietarie di Unix disponibili soltanto per macchine non accessibili a tutte le tasche.

Il progetto GNU capitanato da Richard M. Stallman stava dando dei buoni frutti e molto software libero era stato sviluppato in breve tempo; tuttavia ciò che realmente mancava al progetto era un kernel.

Sostanzialmente, per “liberarsi” sia di Microsoft che di Unix, tutto quello che stava intorno al kernel (compilatori, librerie, editor, applicazioni, ecc.) era pronto ma mancava proprio il cuore del sistema in quanto gli sforzi di sviluppo che si stavano concentrando intorno al progetto GNU/Hurd (ossia il kernel voluto da RMS) non davano i risultati sperati.

L'alternativa esiste!

Tuttavia, agli inizi degli anni '90, un giovane studente finlandese dell'Università di Helsinki (http://www.cs.helsinki.fi/) di nome Linus Benedict Torvalds (http://www.cs.helsinki.fi/u/torvalds/), classe 1969 (28 dicembre), si divertiva, da buon hacker cresciuto a pane e Commodore fin dall'età di dieci anni, a “mettere le mani sopra” ad un piccolo sistema Unix commerciale chiamato Minix (http://www.cs.vu.nl/~ast/minix.html) ed inizialmente allegato ad un libro del suo corso: "Operating Systems: Design and Implementation" (1987) di Andrew S. Tanenbaum (http://www.cs.vu.nl/~ast/), professore alla Vrije Universiteit ad Amsterdam e di Albert Woodhull (in seguito fu venduto come prodotto editoriale distribuito tramite floppy disk).

Tale sistema, ricordato come il primo kernel Unix-like (ossia creato secondo gli standard Posix che definiscono i sistemi operativi compatibili con Unix) per macchine Intel 386, aveva un enorme pregio che agevolava enormemente il lavoro del giovane Linus: a causa degli scopi didattici per i quali era stato creato era distribuito con i sorgenti!

Sotto lo sguardo perplesso e talvolta particolarmente negativo del professor Tanenbaum, Linus continuava imperterrito nella sua opera di miglioramento ed implementazione del piccolo Minix finché non decise di creare un sistema Unix-like ex-novo. Torvalds era infatti profondamente insoddisfatto dei risultati ottenibili con Minix, per cui l'idea di sviluppare un sistema operativo completamente nuovo che consentisse l'utilizzo di macchine Intel (il cui prezzo stava scendendo vistosamente) senza essere obbligati ad utilizzare un sistema Microsoft, non all'altezza delle necessità di un buon hacker, od un sistema estremamente costoso come UNIX trovò immediata-

mente terreno fertile.

Il 25 agosto su comp.os.minix annunciava le proprie intenzioni "Sto lavorando a un (gratuito) sistema operativo (solo per hobby) per i cloni 386(486)AT".

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

Le motivazioni di questa, apparentemente, utopistica iniziativa?

> "I couldn't afford some of the commercial OSes and I didn't want to run DOS or Windows -- I don't even know, did Windows really exist then?".

Era il settembre del 1991 quando il buon Torvalds rilasciava la versione 0.01 (10.239 righe di codice per un totale di 0,2 MB) del proprio sistema operativo o, per essere più precisi del proprio kernel, anche se, come non mancò di osservare immediatamente Tanenbaum, questa release altro non era che una versione di Minix "modificata" neanche troppo bene.

Tanenbaum, infatti, criticava fortemente, e fin da subito, la scelta di Torvalds di optare per una struttura monolitica di kernel (molti sistemi operativi tra i quali anche Windows si affidano invece a microkernel); all'indirizzo http://www.dina.dk/~abraham/Linus_vs_Tanenbaum.html è ancora visibile l'acceso dibattito tra i due nel quale si inserirono anche altre importanti personalità dell'informatica di quei tempi.

In pratica, secondo Tanenbaum, questo sistema nasceva già obsoleto poiché rappresentava una pura e semplice riscrittura di un qualcosa esistente da oltre 20 anni; il professore era, infatti, un fautore di soluzioni basate su microkernel, nelle quali assumono importanza fondamentale i processi separati dal kernel che risulta di dimensioni particolarmente ridotte.

A dire il vero, nello stesso periodo e quindi agli inizi degli anni '90, anche William e Lynne Jolitz stavano iniziando il porting di Unix BSD sul 386 con risultati tecnici anche superiori a quelli ottenuti dal giovane Torvalds (tutti pensavano che il primo successo su pc da parte di Unix sarebbe stata proprio la creatura degli Jolitz): ma, come in tutte le storie “straordinarie”, c'era un ma...

Linus, come precedentemente osservato, voleva un'alternativa al DOS che fosse più “potente” di Minix o, per meglio dire, un sistema tipo UNIX per macchine nelle quali, in quel periodo, poteva girare solo il DOS, ossia

lo stesso risultato più o meno ricercato da William e Lynne: ma Linus era solo e da solo non avrebbe potuto fare niente.

Infatti, fino a quel momento ogni software particolarmente “complicato”, come un sistema operativo, non era stato sviluppato se non in modo attentamente coordinato da un ristretto gruppo di persone più o meno efficientemente collegate tra di loro: 386BSD non era immune da questa strategia di produzione. Ma si ricordi: Linus era solo e non faceva parte di alcun team di sviluppo; c'erano lui, un Intel 386 e tanta voglia di avere un Sistema operativo con la “S” maiuscola (o per lo meno un kernel) che fosse a misura di hacker in grado di funzionare correttamente anche su un semplice pc alla portata di tutte le tasche.

Non c'era altra soluzione per cui decise di chiedere aiuto agli sviluppatori di tutto il mondo attraverso la condivisione del codice: gli aiuti non mancarono fin dagli inizi e dalla prima versione stabile (la ormai storica 0.10 del dicembre 1991, 17.750 righe di codice per un totale di 0,4 MB) all'ultima, passando dal rilascio della 1.0 nel marzo 1994 (176.250 righe di codice per un totale di 4,7 MB), molti anni sono passati ma la filosofia che anima il progetto è sempre la stessa ossia quella di condividere il proprio sapere in cambio di quello degli altri; tutto ciò è reso possibile dalla licenza GNU GPL (General Public License, http://www.gnu.org/copyleft/gpl.html) con la quale il sistema è distribuito fin dalla versione 1.0 (le versioni 0.XX sono “Copyrighted by Linus Torvalds”). Dalla versione 0.12 (utilizzabile da utenti anche non eccessivamente esperti) in poi, la crescita del sistema iniziò a diventare progressiva e dirompente, sia come numero di sviluppatori, sia come utilizzatori.

La svolta “filosofica” di Linus, ha fatto sì che il progetto GNU di Richard Stallman potesse avere un kernel completamente funzionante (e non ancora nello stato embrionale di Hurd); questa svolta ha agevolato gli sviluppatori di tutto il mondo a creare programmi dedicati o a sperimentare porting di software già esistente.

Fin dalla nascita, quindi, questo sistema è stato adottato da centinaia di hacker pronti a dare il proprio contributo alla ricerca del continuo miglioramento; questo sistema non è determinato da rigidi standard che ne possano limitare lo sviluppo; questo sistema seleziona le idee sulla base dei vantaggi tecnici e funzionali che queste possono apportare e non sulla base di meri vantaggi economici; questo sistema si chiama GNU/Linux e dal 1996 (versione 2.0) ha come logo un pinguino di nome Tux (il logo ufficiale è stato disegnato da Larry Ewing, http://www.isc.tamu.edu/~lewing/linux/, mentre il nome che simboleggia "Torvalds UniX" è attribuito da James Hughesin)...

Eppur si muove...

Da sempre nella storia dell'umanità le scoperte e/o innovazioni scientifiche e tecnologiche sono spesso prese in giro e danno adito a continue perplessità nel mondo accademico e non; le visioni utopistiche degli inventori sono sempre state mal comprese dai più tanto da portare addirittura alle condanne a morte.

L'idea di creare un nuovo sistema che potesse soppiantare quello che da sempre ha rappresentato quasi un sinonimo di personal computer ossia MS-DOS prima e Windows (molto di più, a dire il vero) poi sembrava una boutade dell'ultim'ora di un simpatico "smanettone" con l'hobby dell'informatica: in realtà questa idea ha originato il primo modello di sviluppo basato sulla disinteressata collaborazione di migliaia di sviluppatori tramite Internet. E non solo il progetto non è imploso in se stesso come molti pensavano, ma fin dalla versione 1.0 il kernel compare nei siti FTP in rete e sui CD-ROM e soprattutto funziona egregiamente.

Addirittura, verso la fine del 1993, Linux era già in grado di competere per stabilità e affidabilità, con molti Unix commerciali, ed alcune software house iniziarono a progettare e/o realizzare i primi porting delle proprie applicazioni per questo strepitoso sistema. Nato come "passatempo", sviluppato da programmatori sparsi in tutto il mondo senza la certificazione delle competenze, non dotato di supporti finanziari adeguati, non gestito in modo centralizzato da un'azienda del tipo Linux Inc., non agevolato dalla scarsità di informazioni provenienti dai produttori di hardware, senza alcun supporto di marketing... "eppur si muove!"

Il successo fu grande anche se limitato ad un gruppo di utenti sensibile all'importanza di un buon sistema operativo e non agli utilizzatori in generale: comunque sia, aziende che lo distribuivano accompagnato da altri software (le cosiddette "distribuzioni") iniziarono a nascere e, soprattutto in ambienti server, iniziò ad affermarsi sempre più.

Linus, evidentemente, è stato importante nella nascita del sistema e nel coordinamento dei programmatori ma il vero "padre" di Linux è rappresentato dalla comunità di sviluppatori che, animata da finalità ben descritte dalla filosofia GNU, ha sempre portato avanti e migliorato in continuazione il progetto. E questa filosofia trova terreno fertile in Torvalds: "I backup su hard disk sono da primitivi. I veri uomini caricano i loro dati in un sito FTP in modo tale da condividerli".

Con l'esplosione del web e la maggiore facilità di connessione la condivisione non può che rendere il tutto più semplice! Anche per questo motivo, pur se il progetto iniziale di sviluppo era rivolto a processori Intel e

compatibili, ora supporta moltissimi altri tipi di processori tra i quali PowerPc, ARM, Alpha e Sparc.

... e continua a muoversi decisamente bene!

Questo è ciò che è stato, adesso cerchiamo di capire cosa è e cosa verosimilmente sarà.

Il periodo attuale è, se vogliamo, interlocutorio: Linux è cresciuto, si è sviluppato, ha beneficiato di tutti quegli strumenti GNU o commerciali o, ancora, semi-commerciali che gli hanno consentito di implementare nuovi utilizzi in ambiti professionali e non.

Vale la pena di ricordare, a titolo di non esaustivo esempio, sia i progetti a livello di server (apache, mysql, php, bind, jabber, qmail, postfix) che desktop (openoffice.org, kde, gnome, gimp, mozilla) senza dimenticare i passi in avanti delle varie distribuzioni che sono riuscite a realizzare strumenti di installazione e di sviluppo semplici ed intuitivi ma anche potenti e flessibili. Le problematiche legali sorte attorno alla Microsoft hanno portato sempre più spesso l'occhio decisamente poco esperto dell'utente medio a posarsi su questo "strano" sistema gratuito, efficiente e liberamente modificabile.

Alcuni hanno deciso si sperimentarlo, altri si sono arresi quando il winmodem del proprio computer ha smesso, senza apparente motivo, di connettersi ad Internet o la propria fotocamera digitale non ne ha voluto sapere più di trasferire le foto per la stampa; altri ancora hanno tentato di superare questi problemi e spesso ci sono riusciti anche grazie al costante, disinteressato, aiuto di centinaia di programmatori, sviluppatori, amministratori di sistema o dei semplici ex-utenti alle prime armi.

L'importanza di avvicinare l'utente "normale" a GNU/Linux è determinante per il presente ed il futuro di questo sistema. In quest'ottica le distribuzioni che consentono il boot e l'esecuzione (seppur talvolta limitata) del sistema direttamente da cdrom sono un elemento fondamentale per avvicinare gli utenti più timorosi. Ne esistono moltissime anche se vale la pena ricordare Suse Live (http://www.suse.com), Knoppix (http://www.knoppix.org)e Morphix (http://morphix.sourceforge.net).

Quante volte un utente si trova a dover modificare una foto delle proprie vacanze ma ha perso il CD di installazione (originale?) di Photoshop oppure non riesce più ad avviare correttamente il programma dopo aver installato un versione di prova dell'ultimo videogame? Quale migliore occasione per far partire una distribuzione da CD, aprire Gimp, risolvere il problema in un secondo e dimostrare quanto GNU/Liunx non sia così "ostico" come

spesso viene, impropriamente, descritto!?! O cosa ci può essere di meglio di Openoffice.org per aprire un vecchio file .doc che il nuovo fiammante MS-Office di un nostro amico non riesce più "inspiegabilmente" ad aprire e dimostrare la potenza e la versatilità del sistema del pinguino?

Il sistema GNU/Linux è efficiente sia a livello server che desktop anche se in realtà solo nel primo riesce ad ottenere successi ed affermazioni nonostante l'affaire SCO dell'anno scorso e tutt'ora in corso; il settore desktop è sicuramente ancora da affinare ma non tanto dal punto tecnico e stilistico (anche se ogni implementazione è sempre ben accetta), quanto da quello "comunicazionale": l'utente continua a percepire "questo" Linux come un qualcosa di difficile e complesso adatto soltanto a smanettoni e bambini prodigio; "Windows è più facile e si installa da solo", "Con Windows posso vedere i DVD"... sono queste frasi (tutte da verificare!) le chiavi di volta!

Detto questo potrebbe essere puerile ma soprattutto sterile confrontarsi nuovamente in una diatriba "Windows vs. Linux" e sicuramente non è l'obiettivo di questo articolo; quello che vuol essere evidenziato è che il gap esistente soprattutto tra gli utenti "domestici" tra Linux e Windows non è determinato da facilità di utilizzo o potenza tecnica ma dalla stessa identica differenza esistente fin dagli inizi del confronto UNIX-MSDOS/Windows: la capacità di marketing del team di Redmond. In effetti la curva di apprendimento di KGX (Kde/Gnu/linuX) non è troppo diversa da quella di Windows XP o MAC OS X: sicuramente chi ha utilizzato versioni precedenti di Windows sarà avvantaggiato nel passaggio verso XP, ma per tutti gli altri non è detto.

Alcune aziende big nel mondo Linux (prima tra tutte Red Hat, http://www.redhat.com) hanno preferito abbandonare il settore desktop per concentrarsi nel più redditizio (al momento!) ambiente server e professionale; altre si sono buttate a capofitto, per contro, nel mercato dell'utenza domestica (Lindows, http://www.lindows.com, e Libranet, http://www.libranet.com, tra tutti); il tempo darà torto o ragione a queste scelte senza escludere un futuro "antartico" con pinguini ovunque tra sale server, scrivanie, uffici pubblici e salotti domestici...

Secondo i più autorevoli esponenti internazionali del movimento open-source e software libero il futuro prossimo di Linux sarà determinato dal potenziamento delle applicazioni da ufficio e, soprattutto, dalle implementazioni in ambito groupware nonché dall'adozione di questo sistema nella pubblica amministrazione (Brasile, Sud Africa e Sud-Est asiatico, ma anche la nostra Toscana, nel suo piccolo, sono i primi piacevoli esempi di questo trend). Altri fattori di probabile successo possono essere individuati nel cre-

scente malumore dell'utenza nei confronti delle pratiche Microsoft (si osservi: non di Windows tout-court), al rampante supporto dei grandi rivenditori nonché nella grande diffusione di sistemi Linux-based in prodotti di tutti i giorni (televisori, lettori DVD, sistemi per automobili ecc.).

Detto questo, è probabile che nei prossimi mesi (o anni) Linux riuscirà ad ottenere una quota di utilizzatori desktop particolarmente esigenti dal punto di vista tecnico più ampia di quella attuale sul genere del pubblico Apple per fare un facile paragone.

Ma colta la prima "mela" non resterà che affacciarsi alla "finestra"...

Breve cronologia di un successo informatico distribuito senza precedenti

- 1991: Inizia l'avventura di Linux;
- 1992: Nascono le prime distribuzioni (MCC Linux e SLS);
- 1993: Vengono effettuati i primi porting di software commerciali;
- 1994: Viene rilasciata la prima versione definitiva 1.0; nascono RedHat, Debian, SUSE; Linux diventa GPL; sorgono i primi LUGs (Linux User Groups);
- 1995: Iniziano a diffondersi le prime distribuzioni esclusivamente commerciali come Caldera Linux;
- 1996: Viene rilasciata la versione 2.0; Linux è tradotto in più lingue; nasce la mascotte TUX;
- 1997: Torvalds resta coordinatore ma lascia la Finlandia per Santa Clara (Silicon Valley) per lavorare in Transmeta al fine di realizzare microprocessori a basso consumo (ma era questo il vero obiettivo?);
- 1998: Sorge "Google", motore di ricerca destinato a diventare il leader, e si basa su un sistema Linux;
- 1999: Esce il "rivoluzionario" kernel 2.2 che rappresenta un enorme passo avanti rispetto alla diffusione del sistema;
- 2000: IBM annuncia che Linux è disponibile per i suoi sistemi mainframe S/390;
- 2001: Esce il kernel 2.4.0 che contempla notevoli innovazioni rispetto al precedente soprattuto riguardo al supporto hardware di livello professionale; dispone infatti del supporto ai recenti dispositivi hardware e di innovative soluzioni software (file system journaled, logical volume manager, devfs, ipv6);
- 2002: Si inizia a parlare di "alternativa" a Windows soprattutto per utilizzi professionali anche nel campo desktop;
- 2003: SCO denuncia il fatto che Linux utilizza del codice UNIX del quale detiene i diritti (tali affermazioni sono ancora legalmente da verificare); Linus Torvalds si dimette da Transmeta per dedicarsi interamente in OSDL allo sviluppo del kernel di Linux; esce la versione 2.6 del kernel;
- 2004: Tecnologicamente la strada è chiara e le carte sono vincenti... Linux e tutto

il software OpenSource sono decisamente all'altezza sia sui sistemi di fascia alta che sui desktop, oltre ad essere presente nel cuore invisibile di innumerevoli dispositivi elettronici.

Un solo kernel, molte distribuzioni
(di Emmanuele Bello)

Ho pensato fosse utile fornire una panoramica essenziale delle principali distribuzioni di sistemi operativi di tipo GNU, basati su kernel Linux. Ma conoscendo io a mala pena la differenza fra il tasto destro e il tasto sinistro del mouse, era forse il caso che di questo aspetto se ne occupasse un informatico, che tra l'altro è calato direttamente nella realtà della comunità di sviluppo e di promozione di tecnologie open source.

Di ogni distribuzione qui segnalata si riportano le caratteristiche fondamentali, le peculiarità tecniche, alcune informazioni sui progetti che le hanno realizzate o promosse, nonchè i link per scaricarle in rete.

[S. Aliprandi]

Linux è "solo" il kernel di una distribuzione, il resto è composto dai pacchetti software e dalle utility messe a disposizione dal progetto GNU.

Potete trovare, quindi, distribuzioni a pagamento o gratuite, che usano i pacchetti .rpm oppure archivi .deb, che si possono installare oppure si eseguono in "live" e molto altro.

Spesso sentirete parlare di dibattiti su quale "distro" sia migliore ma questo dipende solo da voi e dalle vostre esigenze, potete anche farvi aiutare nella scelta guardando la pagina seguente:

http://www.zegeniestudios.net/ldc/.

Debian – http://www.debian.org

E' una delle distribuzione cult del mondo Linux, creata nel 1993 da Ian Murdock, conserva ancora le idee del progetto iniziale, cioè che tutto deve essere distribuito come software libero. Vanta un parco software di circa 15.000 pacchetti e li gestisce con APT, uno strumento potentissimo. Grazie al suo stile purista vi rende completi padroni del sistema, ovviamente è indicata per persone con un grado di competenza abbastanza elevato.

E' scaricabile gratuitamente, e per un numero elevatissimo di piattaforme, dal link: http://www.debian.org/CD/

Fedora Core – http://fedora.redhat.com

Nata come Red Hat, dalla versione 10 ha cambiato nome in Fedora Core, rappresenta la giusta via di mezzo per chi è alle prime armi e per chi è desideroso di imparare a lavorare con Linux. L'azienda che sta alle sue spalle, Red Hat appunto, fornisce molti prodotti a pagamento e molte soluzioni in collaborazione con multinazionali informatiche. Viene spesso usata sui server web, utilizza pacchetti in formato .rpm. Il supporto software o manualistico offerto dagli utenti in rete, anche per le versioni gratuite, è davvero illimitato.

La potete trovare in versione CD o DVD al seguente link web:
http://fedora.redhat.com/Download/

SuSE – http://www.novell.com/linux/suse/

E' sicuramente una delle distribuzioni più famose ed utilizzate in Europa ed è rilasciata dall'azienda Novell. Viene utilizzata spesso nella pubblica amministrazione, anche grazie al supporto tecnico ed alla documentazione veramente ricca. E' sempre al corrente con i nuovi rilasci di software e tiene tutto in ordine ed aggiornato con il suo tool YaST.

E' distribuita in varie versioni; quella gratuita, OpenSuSE, è disponibile per il download in versione CD e DVD al seguente link:
http://www.opensuse.org/Download

Mandriva – http://wwwnew.mandriva.com/

Al secolo Mandrake, oggi frutto della collaborazione con Conectiva, rappresenta forse la distribuzione Linux più semplice per i neofiti. La gestione è immediata cosi come le configurazioni che non impegnano troppo tempo, grazie al centro di controllo. Sicuramente non è adatta a chi cerca completa autorità sul sistema.

Viene rilasciata in due versioni, una a pagamento e una scaricabile gra-

tuitamente dal seguente link:
http://wwwnew.mandriva.com/en/downloads

Ubuntu – http://www.ubuntulinux.org/

Una delle ultime release di successo basate su Debian, giunta di recente alla terza sua release ufficiale. Nasce in Sud Africa ed oggi vanta una comunità sempre più numerosa. Utilizza l'ambiente grafico Gnome e si pone come obiettivo quello di fornire un sistema operativo per l'umanità. Esiste anche la versione live, che non necessita di installazione; entrambe le potete scaricare gratuitamente dal sito di riferimento.

http://www.ubuntulinux.org/download

Parallelamente ad Ubuntu sono nati due progetti noti come Kubuntu, per chi preferisce l'ambiente KDE, ed Edubuntu che presenta un insieme di strumenti didattici per chi vuole usare Linux nelle scuole come sistema di insegnamento. Anche queste release sono disponibili gratuitamente.

Si veda www.kubuntu.org e www.edubuntu.org.

Slackware – http://www.slackware.com/

Rappresenta la storia delle distribuzioni GNU/Linux. Non semplice da installare per i meno esperti, è invece molto gradita dai puristi, che possono analizzarla a fondo per capirne i funzionamenti. Non esistono versioni commerciali, è stata tra le prime a comparire nel panorama Open Source. Purtroppo per l'assenza di un gestore software potente spesso la si considera per applicazioni server piuttosto che per computer desktop.

E' possibile fare il download delle ultime immagini dal sito:
http://www.slackware.com/getslack/

Gentoo - http://www.gentoo.org/

Gentoo Linux è una distribuzione che può essere ottimizzata e personalizzata per quasi ogni applicazione di cui possiate avere bisogno. Il sistema su cui si basa Gentoo è chiamato Portage ed è interamente scritto in Python. E' sicuramente una distribuzione impegnativa ma vi darà la soddisfazione di aver costruito il vostro sistema Linux quasi da zero (from scratch). Potrebbe non piacere a molti il fatto di dover compilare ogni pacchetto software anche per parecchie ore.

Trovate il sorgente per l'installazione, alla pagina web:
http://www.gentoo.org/main/en/where.xml

QiLinux - http://www.qilinux.org/

QiLinux è una distribuzione Linux italiana realizzata completamente da zero il cui obiettivo è quello di offrire un buon prodotto sia per ambienti Desktop che per ambienti server. Ha un buon supporto software. Viene distribuita anche una versione con il nome di "QiLinux docet" che si propone di portare il software libero nelle scuole italiane.

Trovate le versioni di questa distribuzione gratuitamente seguendo il link download su: http://www.qilinux.org/

Linspire - http://www.linspire.com/

Nata con il nome di Lindows, ha dovuto cambiarlo in Linspire per problemi di somiglianza con un altro sistema. Il primo obiettivo di questa distribuzione è quello di fornire ai suoi utenti un sistema quanto più simile a quelli commerciali, non basati sul kernel Linux. Presenta ottimi strumenti di portabilità tra piattaforme. E' una distribuzione di tipo "commerciale" con prezzi molto accessibili e offre un buon supporto.

Knoppix - http://www.knoppix.org/

Questa distribuzione è la "regina" delle versioni live di GNU/Linux, cioè quelle versioni direttamente utilizzabili dal CD-ROM senza la necessità d'installazione sull'hard-disk. E' disponibile su CD o DVD gratuitamente e vi permette di iniziare ad utilizzare il sistema operativo del pinguino senza problemi. Oggi grazie alle continue evoluzioni è anche possibile interagire maggiormente con il proprio disco fisso ed avere una distribuzione che si comporta a tutti gli effetti come una di quelle classiche.

Da questo progetto ne sono nati innumerevoli versioni e personalizzazioni, come ad esempio l'italiana EduKnoppix arricchita con utili applicativi per la didattica.

Download alla pagina web: http://www.knopper.net/knoppix-mirrors/.

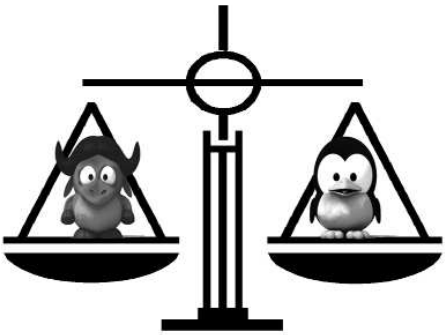
Slax - http://slax.linux-live.org/

Nel panorama delle distribuzioni live, Slax è diventata molto interessante perché molto versatile e prodotta in più versioni con diversi pacchetti software. Vi permette di eseguire un ambiente GNU/Linux senza dover installare nulla sul vostro disco, è basata su Slackware. Può essere usata come utile strumento di ripristino nel caso in cui il vostro sistema operativo via abbia momentaneamente abbandonato.

La trovate gratuitamente alla pagina download del sito:
http://slax.linux-live.org/download.php

Emanuele Bello è Presidente del TorLUG - Tor Vergata Linux User Group
emmanuele.bello@torlug.org - www.torlug.org

Parte quinta
ALCUNI ASPETTI GIURIDICI



Nella pagina precedente:
Un baby-gnu e un baby-pinguino (tratti dal sito www.gnu.org) sulla bilancia della giustizia (elaborazione grafica di Simone Aliprandi).

Il software libero in riferimento alle recenti disposizioni legislative sul diritto d'autore

(di Donato Molino)

Quello di Donato Molino è uno dei più sintetici e allo stesso tempo chiari testi sui rapporti fra il diritto d'autore italiano e il fenomeno dell'informatica libera. Dobbiamo sempre tenere presente che il diritto d'autore europeo-continentale ha alcuni tratti caratterizzanti che lo rendono non sempre coincidente con il concetto anglo-americano di copyright. Inoltre è una peculiarità tutta italiana quella della presenza - a volte ingombrante - di un organo di gestione dei diritti d'autore come la SIAE, ente di diritto pubblico che detiene una specie di monopolio legalizzato su quest'attività.

Su tutti questi aspetti ha cercato di fare luce l'articolo che qui si riporta e che è circolato in vari siti di documentazione già dal 2001. E' importante tenere presente che dal 2001 ad oggi sono sopraggiunte alcune modifiche legislative e regolamentari che hanno in un certo senso migliorato la situazione stigmatizzata dall'autore. Si consiglia infatti (anzi, si raccomanda vivamente) di integrare tale lettura con l'utilissimo articolo - firmato dallo stesso Molino in collaborazione con Leandro Noferini - dal titolo "Bollino HOW-TO", in cui si spiega in modo schematico e concreto come ottenere l'esenzione dall'applicazione del bollino SIAE su supporti contenenti software interamente libero. [S. Aliprandi]

Attraverso il presente documento cercheremo di comprendere come la normativa italiana sul diritto d'autore incida sul software in generale e sul software libero in particolarePremesso che per una completa conoscenza della materia in esame occorre destreggiarsi con termini e concetti di natura giuridica e informatica, vediamo subito di cosa stiamo parlando.Per tutte le opere dell'ingegno umano, vale il principio per cui il creatore originario dell'opera stessa viene definito dalla legge come l'autore, al quale, sotto il profilo giuridico, spettano diritti di natura "personale" e "patrimoniale".Si noti come la legge tuteli queste manifestazioni dell'attività umana come tali e non per una loro qualche utilità. Al contrario, nelle ipotesi di "invenzioni industriali" l'ordinamento giuridico tutela la scoperta solo in funzione di una sua utilità.Questa differenza ci e` agevole al fine di comprendere il motivo per cui la costituzione del diritto d'autore si acquista nel momento stesso della creazione dell'opera e non in forza di una "registrazione" o altre formalita' costitutive.Torniamo al diritto d'autore ed alla distinzione tra diritti personali e patrimoniali. I diritti di natura personale comprendono il cosidetto "diritto morale", secondo cui l'autore ha il potere di esercitare alcuni diritti, quali:

Diritto di paternità;

Diritto di anonimo;

Diritto di inedito (cioè la facoltà di pubblicare l'opera);

Diritto di opposizione (contro modifiche pregiudizievoli all'onore e alla reputazione dell'autore apportate da terzi).

I diritti patrimoniali danno all'autore la facoltà esclusiva di esercitare i seguenti diritti:

Diritto di pubblicazione (spesso e` il presupposto dello sfruttamento economico ed e` trasmissibile, a differenza del diritto di inedito);

Diritto di riproduzione;

Diritto di modifica;

Diritto di traduzione;

Diritto di distribuzione o di smercio.

Nella pratica l'esercizio di questi diritti e` sintetizzato dall'espressione "tutti i diritti riservati". Le regole appena esposte valgono per il diritto d'autore in generale, per i diritti sui libri come per i diritti sul software.Le norme nazionali ed internazionali regolano la complessa materia al fine di tutelare l'autore e i suoi diritti.In Italia la materia è regolata da alcuni articoli del Codice Civile (2575 - 2583), dove si stabiliscono i principi di massima della materia, e da una Legge ad Hoc (Legge n. 633 del 1941, cosiddetta «Legge sul diritto d'autore»). Nel corpo più generale di questa Legge, che

in buona sostanza e' una riproduzione della precedente Legge del 1865, sono state successivamente inserite nuove opere degne di tutela, tra le quali il Software (Decr. Legislativo n. 518/92).Successivamente si e' intervenuti altre volte sul testo di legge, per esempio determinando delle modalita` di tenuta dello speciale Registro pubblico per programmi di elaboratore, (D.P.C.M n.244 del 3/1/94).A tal uopo occorre ricordare che l'instituzione di questo registro non contrasta con quanto detto prima circa l'assenza di qualsiasi formalita' affinche' si costituisca il diritto d'autore, anzi, per il software la registrazione e' "facoltativa", mentre per l'altro registro speciale istituito dalla stessa legge (opere cinematografiche), essa e' obbligatoria. Non e' questa la sede la sede per l'analisi di una simile disparita' di trattamento, ma non si puo' fare a meno di rilevarla.La registrazione del programma si chiede presentando domanda alla SIAE e depositando contestualmente una copia del programma riprodotto su supporto magnetico. Affinche` il programma possa essere registrato e` necessario che sia stato pubblicato.In conclusione: l'iscrizione di un programma per elaboratore nello speciale registro comporta come vero effetto quello di presumere, fino a prova contraria, l'iscritto come autore dell'opera registrata.Di recente il legislatore e` ancora intervenuto con l'emanazione della già citata Legge 248 del 2000, la quale, come vedremo, ha suscitato non poche perplessità in alcuni suoi articoli perché considerati troppo rappresentativi di interessi di imprese che già di fatto beneficiano di situazioni di monopolio.Ma cerchiamo di approfondire i concetti che stiamo sinteticamente premettendo. Il Software attualmente in circolazione, come espressione dell'ingegno umano, si può dividere in:

SOFTWARE LIBERO;

SOFTWARE PROPRIETARIO.

Attraverso questa distinzione intendiamo dire che appartengono alla prima categoria quei programmi che vengono pubblicati senza che l'autore eserciti l'esclusiva sui diritti di sfruttamento economico. Il Software Libero si definisce tale se presenta le seguenti caratteristiche:

0. Il software e` liberamente eseguibile, per qualsiasi scopo;
1. Il software e` liberamente studiabile ed adattabile alle proprie esigenze;
2. Il software e` liberamente copiabile;
3. Il software, modificato o meno, e` liberamente redistribuibile.

Per esercitare le libertà 0 e 2 e` necessario il libero accesso al codice sorgente.Si noti come la libertà 1 si traduca in concreto nella realizzazione massima del principio, ribadito dal legislatore al comma 3 dell'art. 64 ter D. L. 518/92, "..osservare, studiare o sottoporre a prova il funzionamento del

programma..."Appartengono alla seconda categoria (software proprietario) quei programmi che vengono pubblicati insieme all'esercizio in esclusiva da parte dell'autore o del distributore di uno o più dei diritti di sfruttamento economico dell'opera (diritti evidenziati in precedenza). E' comunque prassi comune per il software proprietario l'esercizio da parte dell'autore di tutti i poteri previsti dalla legge al fine di restringere la sfera d'azione dell'utente. A cio' si sommano anche i poteri esercitati da chi distribuisce il prodotto che a sua volta si riserva l'esclusiva di altri e ulteriori diritti.Nel nostro ordinamento giuridico, i casi contemplati dalle varie norme che regolano la materia concernono esclusivamente le ipotesi rientranti nella seconda categoria (software proprietario).E` bene comunque sottolineare che entrambe le categorie illustrate si avvalgono delle regole poste a tutela del diritto d'autore anche se gli obiettivi sono radicalmente diversi. Infatti il software proprietario si avvale delle suddette regole al fine di sfruttare economicamente i programmi per elaboratore e per definirlo come proprietario; Il Software libero invece persegue finalità diverse, infatti il ricorso al diritto d'autore viene effettuato allo scopo di rendere effettivamente libero il programma, in alcuni casi anche al fine di evitare che terze parti possano deprivare gli utenti delle libertà concesse dall'autore originario del software.In entrambe le categorie definite, la distribuzione-circolazione del software avviene per mezzo delle cosidette "licenze d'uso".Le licenze d'uso nelle ipotesi di software proprietario si realizzano per mezzo di un contratto tra l'autore e colui che vuole utilizzare l'opera dell'ingegno. Questi contratti, spesso, conclusi a mezzo di moduli standard, prevedono molte restrizioni e diverse clausole vessatorie, ma per la loro esecuzione e` necessario che il beneficiario del software accetti le condizioni di licenza.Nelle ipotesi di software libero ci troviamo in presenza di un negozio atipico in quanto l'esecuzione del software e` comunque sempre permessa e non e` rinvenibile la bilateralità dei soggetti nel rapporto come nella ipotesi precedente. Diverso e` il caso in cui l'utente decida di modificare e ricompilare il codice sorgente; in questo caso l'utente deve accettare per intero le clausole imposte dalla licenza. La licenza d'uso più utilizzata nel campo del software libero e` senz'altro la GNU GPL (General Public License del progetto GNU); le clausole in essa previste possono essere ricomprese nei seguenti punti:

1.Ridistribuzione libera anche a fini di lucro;

2.Possibilità di modificare il software a patto che si rispettino le seguenti condizioni:

a.Indicazione espressa che la copia e` stata modificata con indicazione

della data di modifica;

b.Il programma utilizzato deve essere interamente utilizzato da terzi secondo le condizioni della licenza originaria;

c.Ad ogni avvio del programma deve esser visibile un messaggio con la nota di copyright, e con l'avvertenza di assenza di garanzia (anche se nulla impedisce che la garanzia possa esser fornita a titolo personale). Il messaggio dovrà inoltre contenere riferimenti precisi sulla locazione della licenza "originaria". Questo punto può essere escluso soltanto nel caso in cui il programma originario non stampi il messaggio.

3.Diritto di distribuzione del software originario o la sua successiva modifica cosi` come indicato nel punto 2, sotto forma di programma oggetto a condizione che si applichi almeno uno dei seguenti punti:

a.Al software sia accluso tutto il codice sorgente;

b.Al software sia acclusa una offerta scritta irrevocabile per almeno 3 anni, offerta con la quale ci si obbliga a fornire a chiunque ne faccia richiesta una copia completa del codice sorgente;

c.Nella ipotesi in cui il software sia stato ricevuto come descritto nel punto b, e soltanto per distribuzioni non commerciali, al software possono essere accluse le informazioni che a sua volta sono state ricevute.

4.Divieto di effettuare distribuzioni, copie o modifiche del software in modi diversi dai punti 1, 2, 3.

Per la versione completa si veda http://www.gnu.org/copyleft/gpl.html. Data la predominanza della licenza GPL e poiché normalmente i programmatori creano opere derivate attingendo dal programmi diversi, risulta importante suddividere le varie licenze usate nel software libero in queste due categorie:

1.Licenze compatibili con la GPL;

2.Licenze incompatibili con la GPL;

Sono compatibili con la GPL:

1.LGPL (Lesser General Public License), molto simile alla GPL, ma diversa da questa in quanto permette il collegamento (linking) con moduli proprietari.

2.Licenza MIT (Massachusetts Institute of Technology)

3.BSD (Berkeley System Distribution) modificata

Non vengono considerate compatibili con la GPL:

1.BSD originale (Berkeley System Distribution), questo tipo di licenza permette la incorporazione del software in un programma proprietario.

2.NPL (Netscape Public License) molto simile alla BSD, ma, rispetto a questa contiene dei privilegi speciali che si applicano soltanto all'autore ori-

ginario.

3.MPL (Mozilla Public License) molto simile alla NPL ma si differenzia da questa in quanto, in essa non viene inserita la clausola che permette all'autore originario di rilicenziare le modifiche (i privilegi che abbiamo visto nella NPL).

Per un elenco piu` dettagliato si veda http://www.gnu.org/philosophy/license-list.html. Quando l'autore di un programma decide di rilasciarlo secondo una delle suddette licenze, ogni utente potra` verificare la bonta` del prodotto (come pure verificare eventuali problemi di sicurezza), potra` modificare il programma per i propri scopi, potra` registribuire il programma ad altri. Va sottolineato come pochi utenti siano effettivamente interessati a studiare il programma nei suoi dettagli, ma e` importante che tale analisi sia lecita (in ogni caso anche un utente inesperto potra` rivolgersi ad un tecnico in caso di bisogno).

Molte nazioni, come la Cina, la Germania, l'Argentina, ad esempio, hanno già annunciato la decisione di voler abbandonare il software proprietario al fine di evitare una dipendenza dai produttori e il rischio di essere obiettivo di attacchi che sfruttano problemi relativi alla sicurezza dei programmi chiusi, problemi che vengono individuati con estrema difficoltà e non possono essere corretti.

E in Italia, qual'è la situazione? Emblematica e non al passo con i tempi a mio avviso.Come dimostrato nel recente dibattito sulla nuova Legge sul diritto d'autore organizzato dal L.U.G. Roma; dibattito che vedeva la partecipazione di Alessandro Rubini, Andrea Monti, e del relatore della legge e del responsabile relazioni istituzionali della SIAE, la situazione e` confusa. Infatti il legislatore italiano, legiferando in materia di software e di diritto d'autore, non dimostra una sufficiente conoscenza del software libero, egli non è in grado di cogliere le differenze che abbiamo evidenziato in premessa tra il software libero e quello proprietario dando per scontato che tutto il Software sia proprietario.

Ne conseguono naturalmente situazioni paradossali e la minaccia di un brusco arresto della diffussione del software libero, come è stato giustamente sottolineato da autorevoli fonti. A tale fenomeno ha contribuito molto la recente Legge n. 248/2000 recante disposizioni in materia di diritto d'autore.

Questa legge interviene specificamente in diverse aree di tutela dei beni di natura intellettuale, dalle fotocopie alla duplicazione di software. Riguardo al software, in forza del nuovo articolo 181 bis introdotto nella Legge sul Diritto d'Autore del 1941, diviene obbligatoria l'apposizione di

un contrassegno SIAE su ogni "supporto" contenente software.

A questo contrassegno sembra attribuirsi un particolare valore distintivo di riconoscimento legale del Software, tanto che la mancanza del contrassegno comporta gravi sanzioni penali. Fintanto che la Legge sarà scritta in questo modo per "supporto" si può intendere di tutto, dalla smart-card al CDROM al disco fisso, provocando non poche difficoltà di collocazione fisica del contrassegno.

Al fine di eliminare tali problemi il legislatore introduce il concetto di custodia, sopra della quale e` lecito apporre il contrassegno, ma lo smarrimento o la perdita accidentale di questa condurrebbero inevitabilmente ad una situazione di illegalita` punibile con sanzioni che prevedono severi periodi di reclusione.Infatti tra le novità introdotte dalla Legge 248/2000 vi è quella relativa alla mutazione delle modalità del "dolo" nella configurazione del reato, aumentando in termini reali le condotte sanzionabili. Si è infatti passati dalla forma "chiunque per fine di lucro" alla forma "chiunque per trarne profitto".

Cosa significa tutto questo?Significa che ogni comportamento che provoca vantaggio patrimoniale diretto o indiretto ("profitto") e` ora sanzionabile come reato, mentre prima solo il vantaggio pecuniario diretto ("lucro") era sanzionabile. Cosi`, per esempio, l'eventuale risparmio di costi è considerato profitto per l'utente, come pure l'attività di studiare un programma da parte di chi lavora nel settore. Inoltre, la norma in esame e` rivolta a "chiunque", pertanto non vi e` nella lettera della Legge alcuna distizione tra organizzazioni imprenditoriali e utenti privati, cosi anche il privato che fà una copia di riserva di troppo del proprio sistema rischia di essere incriminato e punito penalmente.

Va ricordato come la BSA (Business Software Alliance, organizzazione che raccoglie i piu` grandi produttori di software proprietario) ha accolto con un sonoro "finalmente" tale novita`.

Senza entrare nel merito della giustificatezza o meno di tale norma, non possiamo esimerci dal considerare che nel caso di software libero il problema dell'autenticità (garantita dal contrassegno) non si pone, in quanto sono gli stessi produttori e distributori che autorizzano la duplicazione del prodotto. Pertanto non si puo` parlare di pirateria nel caso di software libero, perché non ha alcun senso parlare di originali e di copie, ovvero di quante macchine riproducano il programma in questione. Sarebbe quindi stato piu` giusto esimere dall'obbligo di apposizione del bollino SIAE tutti i prodotti il cui autore non si avvale del diritto esclusivo di riproduzione.

Purtroppo il dado e` tratto e finché il legislatore non interverrà nuova-

mente l'obbligo di apporre il bollino persisterà anche per le ipotesi di software libero. Il legislatore avrà l'occasione per riparare ad una simile ingiustizia con l'emanazione del Regolamento d'attuazione previsto dalla stessa Legge 248, ma il regolamento, ancora in fase di lavorazione, è stato secretato e nulla ci è dato di conoscere sul suo contenuto. Inoltre, nessun rappresentante del mondo del software libero e` stato ascoltato al riguardo da parte delle autorità competenti.

I problemi finora emersi riguardo alla nuova normativa sono tanti. Si pensi per esempio alle ipotesi dei CDROM che le riviste regalano ai loro clienti, ipotesi queste che dovranno per forza realizzarsi in ossequio al nuovo dettato normativo e quindi tramite apposizione di "bollino" SIAE su tali CDROM, pur se il contenuto e` legalmente ridistribuibile. Inoltre bisogna ricordare come i programmi liberi sono oggetto di sviluppo continuo, pertanto e` normale per gli utenti di software libero aggiornare spesso il proprio parco programmi. Ma secondo la normativa ogni versione andrà contrassegnata, arrivando cosi` al paradosso che l'autore-distributore si dovrà accollare un onere economico che potrà anche trasferire sull'utente ma che comunque finirà nella casse della SIAE di fatto impedendo l'applicazione delle norme del diritto d'autore (secondo cui, ricordiamo, l'autore ha la facoltà di avvalersi dei diritti esclusivi, senza per questo essere tenuto a farlo). E` evidente che una simile situazione appare insostenibile oltreché ingiusta.

Ci troviamo davanti ad un sistema che tutela soltanto una parte dei distributori-autori, riconoscendo, e rafforzando di fatto, situazioni monopolistiche, senza tutelare affatto gli utenti che dal Software ottengono dei benefici e gli autori che scelgono di offrire tali benefici. Si puo` pertanto legittimamente affermare che l'obbligo di apporre il bollino SIAE sui supporti contenenti Software libero deve considerarsi illegittimo.In attesa della emanazione dell'annunciato regolamento d'attuazione sono gia` riscontrabili diverse interpretazioni della succitata Legge. Oltre alla posizione della dottrina, cominciano gia` ad affermarsi interpretazioni della SIAE che impongono anche al software libero la tariffa di 100 lire + IVA per l'apposizione del contrassegno (anche se va` ricordato che la SIAE e` organizzata sul territorio per uffici periferici, pertanto non sempre ci troviamo in presenza di un'interpretazione univoca). Ma il vero problema non sono le 120 lire, infatti quello che "pesa" maggiormente su autori e distributori e` il costo amministrativo che occorre sostenere per l'ottenimento dei bollini (contatti con gli uffici preposti per l'ottenimento di informazioni, ore di coda agli sportelli, ecc.). Tali costi amministrativi sono ammortizzati su un

grande numero di esemplari per la grande distribuzione ma risultano inaccetabili per la produzione di software in poche copie.

Molte persone hanno agito nel seguente modo: per non fare una montagna di contrassegni diversificati (ognuno con il costo amministrativo descritto), hanno dichiarato che il supporto contiene, per esempio, un "Sistema operativo GNU/linux", usando poi lo stesso contrassegno per prodotti diversi (seppur simili). Non e` chiaro se tale comportamento sia lecito, in quanto la Legge richiede che il contrassegno identifichi univocamente l'opera.

Un altro problema sta nel fatto che non tutti gli uffici SIAE sono abilitati per operare a pieno regime. Molti uffici periferici accettano la domanda ma in seguito recapitano i bollini richiesti a mezzo poste (con l'aggravio dei relativi costi), oppure presso un altro ufficio SIAE (aumentando quindi i costi amministrativi per chi deve ottenere i contrassegni).

Come ordine di grandezza, su un lotto di circa 500 CDROM il costo effettivo dell'apposizione del bollino puo` arrivare tranquillamente al triplo rispetto al costo del contrassegno stesso. Una simile situazione è inaccettabile per il software libero. Le interpretazioni che sta dando la Siae (in attesa di quelle della giurisprudenza) non risolvono, a mio avviso, il problema. Non risolvono il problema in quanto, per le ipotesi in discussione, sarebbe più rispondente a canoni di equità e di giustizia una espressa esclusione/esenzione delle fattispecie illustrate.

L'esclusione/esenzione del bollino SIAE su supporti contenenti software liberamente riproducibile, realizzerebbe soltanto una giusta valutazione delle fattispecie rispondente a canoni di equità. Ma questa non sarebbe comunque l'unica valutazione, in quanto ritengo che sarebbe altrettanto giusto ed equo che il software libero abbia anche un "riconoscimento" sul piano legislativo almeno pari al riconoscimento ufficiale e solenne finora realizzato per i programmi chiusi e proprietari in senso stretto, dando vita ad un sistema dove di fatto vengono create situazioni monopolistiche ponendo forti barriere al principio della libera concorrenza e di conseguenza al progresso tecnologico

Riepilogando, e` auspicabile una revisione, a breve, del dettato normativo che preveda:

1.Riconoscimento del software libero sul piano legislativo;

2.Esclusione/esenzione del software libero da inappropriati oneri amministrativi ed economici.*

In questo senso dovranno muoversi tutti i poli di aggregazione che si riconoscono negli ideali della comunità del software libero.

* A tal proposito si veda - come già detto nell'introduzione - l'articolo "Bollino HOW-TO" disponibile alla pagina web www.lugroma.org/contenuti/doc/legale.

Il problema dei brevetti sulle idee
(di Alessandro Rubini)

Il brevetto è uno strumento previsto dal diritto industriale per tutelare invenzioni e procedimenti tecnico-industriali. Fin dalla nascita del software come nuova e peculiare tipolgia di opera dell'ingegno, ci si è interrogati sull'opportunità di applicarvi una tutela di diritto d'autore oppure una tutela brevettuale. La scelta a suo tempo cadde (per motivazioni giuridico-dottrinali che qui non è il caso di approfondire) sul diritto d'autore, ma negli ultimi anni si è ripresentata l'ipotesi di cumulare a questa tutela anche quella brevettuale. Ciò - com'è intuibile - si scontra con i principi di libera condivisione sostenuti dal movimento free software, dunque è nata una fitta e attivissima rete di sensibilizzazione su queste tematiche.

L'articolo che segue è l'unico di questa sezione scritto da un 'non giurista' (un informatico, per la precisione); si tratta tuttavia di un 'non giurista' che ha seguito fin dagli albori tutte queste vicende e ha saputo cogliere in modo chiaro e sintetico le problematiche principali. La prima versione dell'opera risale al 2001, ma negli ultimi anni alcuni nuovi risvolti hanno cambiato la situazione (quanto meno quella europea) ed è stato necessario un aggiornamento, fornitomi gentilmente dall'autore in anteprima per questa antologia. [S. Aliprandi]

---

0. Introduzione

In questo documento cerchiamo di presentare in maniera sintetica ma precisa la annosa questione dei brevetti sulle idee astratte, spesso detti «bre-

vetti software» e ultimamente «invenzioni implementate al calcolatore». Le tre denominazioni sono assolutamente equivalenti, nonostante l'ultima dia come assunto che il brevetto sia appropriato in quanto si tratta di «invenzioni».

Il software, in realtà, è una pura elaborazione logica, in nulla dissimile dalla matematica; non a caso, «The Art of Computer Programming», uno dei più completi testi sulle metodologie di soluzione dei problemi tramite calcolatore, è opera di Donald Knuth, un matematico. In questo contesto ci troviamo in difficoltà a definire «invenzioni» i programmi per elaboratore, che sono molto più assimilabili alle dissertazioni che ai manufatti meccanici o di altra natura concreta.

Questo non preclude che una invenzione, un manufatto, possano includere una parte software al loro interno, l'argomento di discussione e` se il software senza il manufatto possa costituire invenzione e sia quindi brevettabile.

1. Il concetto di brevetto

Il brevetto è uno strumento nato per stimolare lo sviluppo della scienza e delle arti utili, come sancito da diverse costituzioni nazionali [1] [2] [3]. Perciò, ogni modifica alla normativa in vigore deve essere giustificata in tal senso, verificando a priori se la modifica proposta aiuti lo sviluppo del settore cui si applica la modifica.

Al fine di stimolare lo sviluppo, un punto cardine della normativa brevettuale in tutte le legislazioni nazionali sta nella rivelazione dell'«insegnamento inventivo» [4], cioè della realizzazione per cui si chiede l'esclusiva, perché lo stato dell'arte possa progredire.

Non a caso, la convenzione europea dei brevetti (Monaco, 1973) vieta la brevettabilità dei metodi commerciali, delle teorie matematiche, dei programmi per elaboratore e altre categorie di invenzioni astratte [5], divieto presente anche nella normativa italiana [6]. Interessante notare come anche Thomas Jefferson [7] si sia espresso contro la privatizzazione delle idee.

È importante ricordare come i programmi per elaboratore ricadano già sotto la normativa del diritto d'autore, come ratificato da tutti i maggiori trattati internazionali: convenzione di Berna [8], TRIPS (trade-related aspects of intellectual property rights) [9], trattato sul Copyright del WIPO [10]. Su queste basi legali, ogni argomentazione sulla necessità di «proteggere» il software è infondata. Nessun settore dell'attività umana giova di entrambe le normative, quella brevettuale e quella autorale.

1.1. La tutela dell'inventore

Il concetto di tutela dell'inventore, spesso usato per giustificare un allargamento del campo di applicazione dei brevetti, ha la sua ragion d'essere nel momento in cui lo sviluppo di un'invenzione richiede costosi investimenti, anche considerando che non tutte le invenzioni si riescono a convertire in un prodotto commercialmente interessante. Il monopolio ventennale (il brevetto) sull'utilizzo dell'invenzione di successo serve anche a coprire le spese di ricerca che non hanno uno sbocco produttivo.

Questa situazione non ha alcun riscontro nel campo delle idee astratte; non esistono costi di ricerca concreti a fronte dello sviluppo di idee, per cui non è necessario concedere l'esclusiva sull'utilizzo della presunta invenzione, perche` l'idea viene realizzata in ogni caso. Ma come diceva Thomas Edison, «il lavoro dell'inventore è 1% ispirazione e 99% essudazione», e quando all'idea astratta viene aggiunto il vero lavoro, il 99%, otteniamo uno specifico programma, il cui autore è tutelato dalla normativa sul diritto d'autore.

Chiunque lavori in un campo informatico produce in continuazione nuove idee o nuovi programmi per elaboratore e spesso la stessa procedura viene realizzata indipentemente da vari attori. Permettere l'utilizzo esclusivo di tali realizzazioni ad uno solo degli autori significa quindi impedire l'attività indipendente di tutti gli altri. In un regime di brevettabilità delle idee, chi lavora usando gli elaboratori troverà quindi il suo cammino pieno di intoppi. Nemmeno chi detiene un brevetto risulterà tutelato, dovendosi scontrare con innumerevoli altri brevetti non appena svolgerà attività produttiva.

É anche importante ricordare che ottenere un brevetto non è una pratica semplice, per cui molte piccole imprese semplicemente non potranno usufruire di questa possibilità, ma dovranno lavorare in un campo minato dai brevetti realizzati dai loro concorrenti.

1.2. La tutela del patrimonio culturale

Un'altra caratteristica fondamentale dei sistema brevettuale, è la limitazione temporale del monopolio garantito all'inventore. Tale limitazione è stabilita al fine di non bloccare lo sviluppo tecnologico del sistema produttivo, pur garantendo all'inventore un arco di tempo in cui godere in modo esclusivo dell'invenzione e recuperare gli investimenti di ricerca. In questo arco di tempo l'insegnamento inventivo è comunque gia` stato pubblicato e arricchisce il patrimonio culturale complessivo.

Mentre garantire un monopolio di venti anni può essere sensato nel campo delle realizzazioni meccaniche o idrauliche, tale arco di tempo non ha la minima correlazione con il ciclo di vita di un pacchetto software, che

si misura in due o tre anni al massimo. Una copertura brevettuale largamente superiore al ciclo di vita di un prodotto non può che bloccare la crescita culturale e limitare lo sviluppo complessivo di un settore produttivo, nuocendo quindi agli operatori del settore, con la sola esclusione dei pochi che si sono assicurati una copertura brevettuale sufficiente a non venire schiacciati da portafogli più nutriti.

2. Il brevetto sulle idee oggi

Nonostante il brevetto cosiddetto «software» non sia consentito dalla legge, almeno fino ad ora, sono state già concesse, anche in Europa, diverse migliaia di brevetti di tale tipo, anche perché la pratica è molto diffusa negli Stati Uniti.

2.1. La situazione statunitense

Il brevetto «software» sarebbe proibito negli Stati Uniti come lo è stato finora in Europa, ma la pratica legale, che in quel sistema ha un peso normativo rilevante, ha rivoltato questa norma.

I brevetti astratti americani vengono in genere acquisiti da grosse società (IBM, Apple, Microsoft, ...) che li usano come merce di scambio con altre società, oppure da persone giuridiche create appositamente, le cosiddette «litigation companies», la cui unica attività è riscuotere licenze d'uso sui brevetti che detengono, senza svolgere alcuna attività produttiva né inventiva.

Ovviamente, la situazione è tutt'altro che rosea per chi produce nel campo tecnologico e non è stato ancora assorbito in una grossa azienda. Non a caso la piccola e media impresa nel campo tecnologico è quasi inesistente negli Stati Uniti.

Numerose organizzazioni e persone autorevoli hanno espresso questo tipo di problemi, ma il sistema legale non ha alcun interesse ad affrontare il problema. Si vedano per esempio gli interventi della «League for Programming Freedom» del 1991 [11], o la lettera di Donald Knuth del 1995 [12]; interventi più recenti sono citati più avanti.

La galleria dell'orrore americana è ricca di esempi di brevetti dannosi per il mercato e la società [13]. Alcuni esempi sono «l'acquisto via rete con un singolo click» di Amazon (banale applicazione degli strumenti web esistenti), un algoritmo geometrico per linearizzare immagini panoramiche (realizzato in due ore da uno di noi, ignorando la questione del brevetto), il brevetto sul «link» di British Telecom (il «link» delle pagine web, un banale riferimento ad un documento esterno).

Il brevetto sul singolo click è a tutti gli effetti relativo ad un metodo

commerciale, che viene precluso ad ogni altro venditore. Ma oggi sono brevettabili negli stati uniti i metodi commerciali in gernere, anche quando svincolati da un'implementazione tramite calcolatore.

2.2. La situazione europea

In Europa, come sottolineato all'inizio, i brevetti «software» partivano da una situazione di divieto. Nel '97, però, la Commissione ha proposto di valutare l'introduzione legale dei brevetti astratti. Tale suggerimento era motivato dal «bisogno di uniformare il mercato europeo a quello americano», pensando con ciò di aiutare il mercato europeo. Cio` ha portato nel 2002 alla stesura di una proposta di Direttiva Europea in tal senso da parte della Commissione.

La Commissione Europea ha anche finanziato uno studio sugli effetti di una modifica della normativa, ma invece di commissionarlo ad un gruppo di studiosi di macroeconomia lo ha affidato all'«Intellectual Property Institute» di Londra, che evidentemente non può avere un atteggiamento scientifico e imparziale sul tema. Le conclusioni dello studio [14] dicono che «lo sviluppo dell'economia statunitense ha beneficiato dalla brevettabilità del software» e che «le nostre piccole e medie imprese non reputano interessante usufruire dei brevetti, ma potrebbero benissimo cambiare idea». L'unica cosa dimostrata, insomma, è la posizione delle nostre imprese, contrarie all'estensione della brevettabilità.

In realtà, la proposta di direttiva in favore dei brevetti software costituisce una minaccia alla libera concorrenza e alle piccole e medie imprese. Se, al contrario, i brevetti software non verranno introdotti, la U.E. godrà di condizioni più favorevoli all'economia ed alla concorrenza tra gli attori, condizioni meno adatte a pratiche di monopolio camuffate da azioni legali di protezione delle invenzioni.

La Commissione Europea ha sollecitato pareri sul problema [15]. I risultati [16] della consultazione sono largamente contro i brevetti astratti da parte del mondo tecnico e produttivo, mentre i pareri favorevoli sono limitati principalmente ad organismi legali e grandi aziende già detentrici di brevetti negli Stati Uniti. In particolare, è interessante notare come il 90% delle piccole e medie imprese si sia espressa contro. Si noti come l'analisi delle risposte sia nascostamente ma decisamente di parte, in quanto si parla di «peso economico» delle risposte, e i dati vengono interpretati in maniera a dir poco bizzarra [17].

Nel frattempo, nonostante la norma vigente vieti tuttora la concessione di brevetti su concetti astratti, l'Ufficio Brevetti Europeo ha già approvato

più di 30.000 di tali brevetti, arrivando al punto di piegare le normative, diramando direttive per gli esaminatori in diretto contrasto con la legislazione vigente [18].

La «galleria degli orrori» dei brevetti europei [19] offre già un'idea di quello che ci aspetta. Andiamo dal brevetto sul formato grafico JPEG alla diagnosi automatica (qualunque diagnosi), dal confronto della pronuncia dell'allievo con la pronuncia dell'insegnante, al ridimensionamento di una finestra grafica quando è oscurata da un'altra finestra, includendo la conversione di nomi da una convenzione ad un altra per rappresentarli.

2.3. L'iter della Direttiva Europea

La soria della Direttiva sulle «Invenzioni implementate al calcolatore», dopo la sua stesura inziale, e` stata abbastanza travagliata. In data 24 Settembre 2003, il Parlamento Europeo ne ha pesantemente emendato il testo, rendendendolo praticamente innocuo, facendo salvo il fine originale di armonizzazione della normativa comunitaria.

Ignorando questo voto, il 18 Maggio 2004 il Consiglio dei Ministri della UE ha approvato un nuovo testo, definito «di compromesso», che avrebbe permesso la brevettabilità praticamente illimitata dei programmi per elaboratore.

Dopo varie peripezie e il rischio di un'approvazione senza discussione (evitato in extremis da alcuni Ministri dei paesi membri), la Direttiva e` approdata in Parlamento per la seconda lettura, dove e` stata bocciata dalla maggioranza assoluta degli aventi diritto, il 6 Luglio 2005. Questo risultato e` dovuto al lavoro di tante realta` industriali e non, che sono riuscite a far presente ai Parlamentari un punto di vista alternativo rispetto a quello delle potenti lobby presenti a Bruxelles.

La questione pero` non e` chiusa e dobbiamo aspettarci nuove mosse verso la brevettazione indiscriminata. Per esempio,la bozza di Direttiva sul «Brevetto Europeo», presentata il 20 Gennaio 2006, il cui scopo e` semplificare la burocrazia associata al rilascio dei brevetti, contiene una norma che da` mano libero all'Ufficio Europeo dei Brevetti nel definire le proprie regole per l'approvazione dei Brevetti, indipendentemente dalla Convenzione Europea cui ora dovrebbe essere soggetto.

3. Problemi pratici

La brevettabilità delle idee astratte non solleva solo problemi di principio, ma anche problemi pratici non indifferenti. Tali problemi vengono riconosciuti anche dai sostenitori dei brevetti software, anche se non ne viene riconosciuta la strutturale irrisolvibilità.

3.1. Valutare lo stato dell'arte

Non è pensabile che un ufficio brevetti possa valutare lo stato dell'arte, quando l'arte in questione copre tutto lo scibile umano (in quanto ogni concetto astratto può essere messo nella forma brevettabile di «programma per elaboratore»).

Il risultato è che la maggior parte dei brevetti rilasciati coprono realizzazioni che sono obsolete al momento stesso della richiesta. Ovviamente, una volta concesso il brevetto, nessuno può muoversi in quella parte dello scibile umano senza pagare o essere portato in tribunale.

Lo stesso Gregory Aharonian, strenuo sostenitore dei brevetti sul software sottolineava già nel 1994 come non sia possibile evitare di concedere un enorme numero di brevetti su idee banali [20]. Ora Aharonian sostiene anche la brevettablita` delle creazioni artistiche e ricreative [21], lavorando come consulente nell'invalidare i brevetti che non meritavano di essere assegnati.

3.2. Valutare il passo inventivo

Un altro problema insolubile è come valutare il «passo inventivo» necessario per l'ottenimento di un brevetto. La maggior parte dei brevetti «software» in effetti non contengono alcun passo inventivo, come esemplificato da Richard Stallman [22].

Inoltre, come citato all'inizio, l'«insegnamento inventivo» relativo al brevetto deve essere rivelato. Invece questo requisito viene spesso aggirato nel caso dei brevetti sulle idee astratte, in quanto la rivelazione dell'invenzione consiste semplicemente nella descrizione a grandi linee del problema (l'1% nella suddivizione di Edison) più che della soluzione allo stesso; questo nonostante il passaggio dall'idea astratta alla realizzazione pratica sia la parte più impegnativa del lavoro inventivo.

Negli altri campi tecnologici il brevetto si riferisce al prodotto finito, cioe` l'idea e l'arduo lavoro di realizzarla al fine di produrre un «unsegnamento inventivo sull'uso delle forze naturali controllabili». Le forze della natura non entrano nel software, che rimane una creazione logica, e l'opera completa e`, come gia` notato, ambito del diritto d'autore.

In effetti, quasi tutti gli esempi nelle varie gallerie degli orrori si riferiscono proprio a brevetti sul problema piuttosto che sulla soluzione del problema stesso.

4. Perché brevettare il software?

In un quadro come quello descritto, perché c'è così tanta spinta a brevettare il software? Forse il software non è già «protetto» dal diritto d'autore,

e nessun altro ambito di produzione umana rientra contemporaneamente nel dominio del diritto d'autore e del brevetto?

4.1. Le argomentazioni a favore

Chi argomenta a favore della brevettablità del software in genere usa due argomentazioni: «la tutela del povero inventore» e «l'uniformità del mercato internazionale».

Purtroppo nessuna delle due argomentazioni è sostenibile. Il povero inventore dopo aver investito in spese legali per avere il suo brevetto non potrà far altro che sostenere ulteriori spese legali per difendersi dalle cause legali per violazione di altri brevetti. Se ciascuno recintasse il suo metro quadrato di terreno il risultato sarebbe che nessuno potrebbe muoversi e gli unici a guadagnarci sarebbero i venditori di recinzioni.

L'uniformità del mercato sicuramente non è un argomento sostenibile non appena si verifica la situazione negli Stati Uniti, dove le aziende minori vengono spesso soffocate o acquisite a causa di supposte violazioni di brevetto (gli esempi fanno parte della galleria degli orrori già citata).

É interessante notare, poi, come chi spinga per la brevettabilità delle idee sia sempre qualcuno che ha interessi personali nella questione: o si tratta di dirigenti degli uffici brevetti, o si tratta di studi legali specializzati nella questione, o si tratta di aziende dotate già di un consistente portafoglio brevettuale.

4.2. Gli studi economici in proposito

Naturalmente non mancano gli studi macroeconomici del problema, e tutti invariabilmente dimostrano come l'estensione del regime di brevettabilità sia nocivo per lo sviluppo del mercato e della tecnologia [23].

Fritz Machlup, già nel 1958 sosteneva che il sistema brevettuale non porta vantaggi nel mercato in cui viene inserito [24]. Da allora sono stati realizzati numerosi altri studi, fino a quello di Bessen e Maskin, del dipartimeto di economia del MIT, che nel 2000 hanno dimostrato la nocività di un sistema brevettuale negli ambiti economici dinamici [25].

Personalmente evitiamo di considerare le analisi realizzate da studi legali o «istituzioni per la protezione della proprietà intellettuale» di qualunque tipo, per la evidente parzialità delle parti. Ma anche questi studi spesso non riescono a giustificare le conclusioni cui arrivano.

5. Conclusioni

L'istituzione della brevettabilità del software è nociva per la piccola e

media impresa. L'attuale spinta verso questa direzione viene da soggetti con un diretto interesse personale nella questione (uffici brevetti, studi legali, grandi aziende).

Qualsiasi programma per elaboratore di una qualche utilità infrange dozzine di brevetti software già validi in USA e che verrebbero riconosciuti anche qui da noi, perciò il mito della «tutela del piccolo inventore» risulta, appunto, soltanto un mito.

Non mancano gli studi economici indipendenti a sostegno di questa tesi, mentre gli studi in direzione opposta vengono invariabilmente da parti con specifici interessi.

La piccola e media impresa si è finora schierata contro la brevettabilità delle idee astratte, ma occorre prestare molta attenzione agli eventi «informativi» sul problema, valutando con attenzione la posizione di chi prende la parola.

L'attuale impostazione rappresenta un vantaggio competitivo dell'Europa rispetto a USA e Giappone. Non abbiamo motivo di cambiarla, per non esporci a pratiche di monopolio e di eliminazione sleale della concorrenza, proprio nel campo strategico delle nuove tecnologie informatiche.

6. Per saperne di più

Questo documento, nella versione aggiornata, è disponibile come http://www.linux.it/GNU/nemici/brevetti.shtml.

EuroLinux gestisce una petizione per l'abolizione dei brevetti software: http://petition.eurolinux.org.

FFII, organismo europeo, cura un sito molto ricco sul problema: http://swpat.ffii.org/indexen.html.

La LPF e la EFF seguono il problema negli Stati Uniti: http://lpf.ai.mit.edu e http://www.eff.org.

- -

NOTE

1: «The Congress shall have power to [...] promote the progress of science and useful arts, by securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries» («Il parlamento ha il potere di [...] promuovere il progresso della scienza e delle arti utili garantendo ad autori ed inventori diritti esclusivi sui propri scritti e sulle proprie scoperte»). In: La costituzione degli Stati Uniti:

2: L'articolo 14 della costituzione tedesca, secondo il Bundestag è incompati-

bile con la brevettazione delle idee.

3: La costituzione italiana non tocca l'argomento e la normativa sui brevetti (CC 2584-2594 decreto 1127/39, decreto 244/40, decreto 360/1994) si limita a descrivere le modalità di attuazione.

4: «The European patent application must disclose the invention in a manner sufficiently clear and complete for it to be carried out by a person skilled in the art» («La domanda di brevetto europeo deve rivelare l'invenzione in una modo sufficientemente chiaro e completo perché possa essere realizzata da una persona esperta nel settore»). European Patent Convention, Art. 83

5: «2. The following in particular shall not be regarded as inventions within the meaning of paragraph 1: [...] (c) schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers» («In particolare, non sono considerate invenzioni nel significato del paragrafo 1: [...] (c) schemi, regole e metodi per svolgere atti mentali, disputare giochi o fare commercio, e i programmi per elaboratore»). European Patent Convention, Art. 52.2c

6: Decreto 1127/1939 e successive modificazioni, articolo 12, comma 2: «Non sono considerate come invenzioni ai sensi del precedente comma in particolare: [...] b) i piani, i principi ed i metodi per attività intellettuali, per gioco o per attività commerciali e i programmi per elaboratori»

7: «If nature has made any one thing less susceptible than all others of exclusive property, it is the action of the thinking power called an idea» («Se la natura ha reso una specifica cosa meno passibile di proprietà esclusiva rispetto alle altre, questa è l'azione della facoltà intellettuale chiamata idea»). Citato per esempio in «The Economy of Ideas: A Framework for Patents and Copyrights in the Digital Age ((Everything you know about intellectual property is wrong)», di John Perry Barlow (http://www.eff.org/cafe/barlow.html).

8:Convention de Berne pour la protection des oeuvres litteraires et artistiques http://www.law.cornell.edu/treaties/berne/overview.html

9: Art. 10 - Computer Programs and Compilations of Data1. Computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention (1971). Il testo originale si può trovare all'indirizzo http://www.wto.org/english/tratop_e/trips_e/t_agm3_e.htm

10: Article 4 Computer Programs Computer programs are protected as literary works within the meaning of Article 2 of the Berne Convention. Such protection applies to computer programs, whatever may be the mode or form of their expression.http://www.wipo.org/eng/diplconf/distrib/94dc.htm

11: «Against Software Patents», della League for Programming Freedom (http://lpf.ai.mit.edu/Patents/against-software-patents.html) e «Why Patents Are Bad for Software», di Garfinkel, Stallman e Kapor

(http://lpf.ai.mit.edu/Links/prep.ai.mit.edu/issues.article).

12: Donald Knuth è uno dei maggiori studiosi mondiali di informatica. La sua lettera all'ufficio brevetti, del Febbraio 1994, denuncia come il brevetto software sia dannoso per la maggior parte delle persone che lavorano con gli elaboratori, a beneficio solo del sistema legale e di un piccolo gruppo di inventori. La lettera è riportata su http://lpf.ai.mit.edu/Patents/knuth-to-pto.txt, ed è disponibile in traduzione italiana su http://no-patents.prosa.it/brevetti/docs/knuth_letter_it.html.

13: La galleria degli orrori dei brevetti statunitensi è su http://lpf.ai.mit.edu/Patents/.

14: Lo studio è disponibile come PDF
(http://europa.eu.int/comm/internal_market/en/indprop/study.pdf)
e una sintesi è disponibile sulla pagina
http://europa.eu.int/comm/internal_market/en/indprop/studyintro.htm).

15: La consultazione è stata aperta il 19 Ottobre 2000, è disponibile il documento in PDF (http://europa.eu.int/comm/internal_market/en/indprop/soften.pdf) e un riassunto sulla pagina web
http://europa.eu.int/comm/internal_market/en/indprop/softpaten.htm.

16: Le risposte, pubblicate nel Luglio 2001, sono disponibili su http://europa.eu.int/comm/internal_market/en/indprop/softreplies.htm, mentre l'analisi è un file PDF:
http://europa.eu.int/comm/internal_market/en/indprop/softanalyse.pdf

17: Per esempio, a pagina 14 dell'analisi si sottolinea come la percentuale di piccole e medie imprese a favore e contro i brevetti software sia equivalente (15%), trascurando che il dato reale è nel 90% delle piccole e medie imprese schierate contro i brevetti software. Il fatto che tali imprese siano il 15% dei pareri raccolti è irrilevante.

18: Si veda il comunicato stampa di EuroLinux: http://petition.eurolinux.org/pr/pr14.html,
in italiano su http://www.softwarelibero.it/news/news011109_01.shtml.

19: La galleria degli orrori faccolta di FFII include sia esempi di brevetti malfatti (http://swpat.ffii.org/vreji/pikta/mupli/index.en.html) sia esempi di impedimenti allo sviluppo software a causa di brevetti (http://swpat.ffii.org/vreji/pikta/index.en.html).

20: Nota Aharonian come gli Stati Uniti hanno visto una crescita esponenziale dei brevetti software (e delle cause legali associate), mentre la crescita dell'innovazione non può essere più che lineare («In short, we have strong exponential growth in the number of software patents, and everything that derives from these patents, including lawsuits. Now it can probably be shown that the number of unobvious, novel, unpublished news ideas in the software world grows linearly at best.»). In

http://www.eff.org/Intellectual_property/crisis_softpatent.article.

21: http://patenting-art.com.

22: «The Anatomy of a Trivial Patent», pubblicato da LPF: http://lpf.ai.mit.edu/Patents/anatomy-trivial-patent.txt.

23: La raccolta di FFII è molto interessante, e si trova presso http://swpat.ffii.org/vreji/minra/siskuen.html.

24: Fritz Machlup, «The Economic Foundations of Patent Law», 1958, http://www.ipmall.fplc.edu/hosted_resources/jepson/unit1/aneconom.htm: «If we did not have a patent system, it would be irresponsible... to recommend instituting one».

25: James Bessen and Eric Maskin, «Sequential Innovation, Patents, and Imitation», 2000, http://www.researchoninnovation.org/patent.pdf.
Interssante anche la pagina di FFII, che contiene anche una critica al modello, fatta da un anonimo avvocato del campo dei brevetti: http://swpat.ffii.org/vreji/papri/bessenmaskin00/indexen.html.

L'effettività giuridica delle licenze copyleft

(di Simone Aliprandi)

In questo articolo - già apparso in due diverse occasioni editoriali - ho cercato di far luce su alcuni aspetti problematici del cosiddetto "enforcement" dei diritti derivanti da licenze copyleft. Di primo acchito, possono sembrare argomenti di interesse riservato unicamente ad avvocati e giuristi; ma le domande che spesso mi sento rivolgere via e-mail o durante conferenze e lezioni, dimostrano che c'è in effetti un certo bisogno d'informazione in materia, anche per chi si affaccia a questo mondo da semplice curioso.

[S. Aliprandi]

Da quando mi occupo degli aspetti giuridici legati alle licenze copyleft in ambito sia software che non software, la domanda che più spesso mi sento porre è: "ma queste licenze hanno validità legale?".

Tale interrogativo, posto in questi termini, agli orecchi di chiunque abbia un'infarinatura di concetti giuridici, suona come un'assurdità e una goffaggine, ma cela una legittima serie di dubbi relativi alla reale efficacia di questi strumenti di tutela delle opere.

Innanzitutto la parvenza di goffaggine deriva dal fatto che, essendo tali licenze attinenti all'ambito contrattuale, non si può parlare di una loro invalidità giuridica a priori. Mi spiego in parole accessibili anche a chi non ha alcuna nozione di base: il contratto è uno strumento di diritto privato con

cui le parti regolano una certa situazione di rilevanza giuridica e da cui si generano fra esse degli obblighi reciproci. Il diritto pubblico (cioè quello attinente alla sfera legislativa) non può far altro che regolare le forme con cui questa potestà contrattuale privata dev'essere esercitata e al massimo indicare quali siano i vizi che determinano l'invalidità del contratto (o di alcune sue parti). Questo per dire che le licenze copyleft, in quanto appunto rientranti nell'ambito del diritto privato (come d'altronde tutta la gestione dei diritti di proprietà intellettuale), non possono essere considerate "legali o illegali"; è piuttosto da verificare se le principali licenze (freesoftware, opensource o opencontent) nate ad esempio in un contesto statunitense non contengano alcune clausole che, applicate in un contesto europeo, risultino invalide.

E' questo un aspetto decisamente delicato, poichè i sistemi giuridici anglo-americano e europeo-continentale denotano alcune rilevanti differenze proprio nel trattamento dei diritti d'autore e in un certo senso anche nella definizione del tipo contrattuale in questione: mi riferisco alla disputa sul fatto che tali licenze siano da considerare atti unilaterali oppure atti bilaterali. Tale disputa è però più di natura giurico-dottrinale che pratica e per questo compete più agli accademici che ai consulenti o agli avvocati. A questi soggetti si pone più che altro il problema dell'interpretazione (in senso giuridico) delle disposizioni contenute in queste licenze; problema che si pone però solamente nel momento in cui si instauri una controversia civile nata dall'applicazione della licenza. Questo vale a dire - in termini molto essenziali – che ogni licenza è giuridicamente perfetta finchè qualcuno che abbia avuto a che fare con essa non intenda contestarla di fronte a un giudice o in sede stragiudiziale. E' a questo punto che gli avvocati delle varie parti coinvolte e poi il giudice cercheranno di entrare nel merito del significato giuridico delle clausole della licenza, verificando quali di esse siano da considerare eventualmente invalide oppure a quali di esse attribuire una certa interpretazione piuttosto che un'altra.

Ecco chiarito perchè da molte parti si sente parlare di una necessità di test giudiziali sulle licenze, ovvero di reali pronunce di organi giurisdizionali da cui poter trarre i primi principi guida per l'enforcement di questi strumenti di tutela. Considerate che – nel sistema delle fonti del diritto –, in mancanza di precise disposizioni di legge, un rilevate ruolo è ricoperto dalle sentenze delle supreme corti che con la loro giurisprudenza possono sgombrare il campo dai più pregnanti dubbi interpretativi.

Attualmente, in europa attualmente l'unico punto di riferimento è una sentenza (risalente all'aprile 2004) di un tribunale tedesco il quale ha con-

dannato al risarcimento del danno una società che aveva reso proprietarie (cioè "chiudendone" il codice sorgente) parti di codice rilasciate sotto licenza GPL. E' ancora troppo poco per poter avere dei riferimenti giurisprudenziali rilevanti, soprattutto perchè trattasi di pronuncia di primo grado suscettibile di revisione in appello; ma è comunque un inizio.

Alcuni malignamente (e miopemente) sostengono che la scarsità di pronunce giurisdizionali sulla GPL (o su licenze simili) derivi da una sorta di paura delle associazioni come Free Software Foundation a far valere in giudizio le proprie ragioni o da una pretesa mancanza di organizzazione delle stesse. Ma Eben Moglen (principale consulente legale di FSF) nel suo articolo "Enforcing the GNU GPL" (disponibile alla pagina http://www.gnu.org/philosophy/enforcing-gpl.html) ha tenuto a precisare con grande sicurezza: "La verità è esattamente il contrario: non ci siamo mai trovati a portare la GPL in tribunale perché mai nessuno ha voluto correre il rischio di contestarcela in quella sede."

Al di là di posizioni non del tutto neutrali e non scevre da componenti propagandistiche, non c'è dubbio che ancora molti problemi legati alla tutela giuridica delle licenze copyleft rimangono aperti. Ne cito principalmente due, entrambi con grandi ripercussioni nella sfera pratica: uno è relativo all'individuazione dei soggetti che hanno titolo per difendere in giudizio la licenza. Il software sviluppato sotto il modello free/open è per definizione un'opera con un numero indefinito di autori, dunque è difficile individuare chi tra essi (tutti? alcuni? quali?) abbia titolarità giuridica per attivarsi in sede legale. E l'altro è relativo alla gestione pratica dell'attività legale: cause di questo tipo infatti posso risultare piuttosto onerose e le fondazioni/associazioni (a volte piuttosto piccole) che raccolgono gli sviluppatori di questi software potrebbero incontrare la resistenza opposta da imprese (a volte piuttosto grandi) dotate di mezzi e di visibilità incomparabili. E' proprio per questo motivo che a New York è stato costituito da qualche mese il Software Freedom Law Center, un ente di consulenza e assistenza legale specializzato e partecipato da giuristi di spicco come lo stesso Moglen (FSF) e Lawrence Lessig (Creative Commons): l'esistenza di un centro catalizzatore come questo può certamente favorire il coordinamento dell'attività legale in ambito copyleft.

Il Software Freedom Law Center

Il Software Freedom Law Center è uno studio di cosulenza e assistenza legale in materia di free e open source software, operante negli Stati Uniti e con sede a New York (Manhattan). Il team è composto da Eben Moglen, Diane M. Peters, Lawrence Lessig, Daniel J. Weitzner, Daniel B. Ravicher, Bradley M. Kuhn. Lo studio offre anche la possibilità per giovani giuristi di svolgere un bimestre di praticantato a fianco dei suoi professionisti. Maggiori informazioni sul sito: www.softwarefreedom.org.

Parte sesta
DALL’OPENSOURCE ALL’OPENCONTENT

Nella pagina precedente:
Il simbolo di Creative Commons

La licenza GNU FDL
(Free documentation license)

E' una prassi abbastanza comune fra gli informatici scrivere della documentazione relativa allo sviluppo del loro software: testi di presentazione, commenti, istruzioni tecniche, veri e propri manuali... Come lo stesso Stallman ha fatto notare in molti dei suoi saggi di sensibilizzazione, "il software libero ha bisogno di documentazione libera". Proviamo a pensare al paradosso che si creerebbe se distribuissimo un software sotto GPL con un manuale sotto copyright tradizionale: ogni volta che qualcuno realizzerà delle modifiche al software, sarà costretto a scrivere un nuovo manuale o comunque non potrà integrare quello già esistente.

Inizialmente gli sviluppatori del progetto GNU, per far fronte a questa esigenza, pensarono di applicare la GPL anche alla documentazione: ciò è tecnicamente fattibile, ma dobbiamo tener presente che si tratta di una licenza pensata originariamente per il software e in cui si usano concetti e definizioni rivolte al mondo informatico. Finalmente nel 2000 la Free Software Foundation pensò di redigere il testo di una licenza pensata appositamente per la documentazione e per le opere testuali in generale. Nasce così la GNU Free documentation license (FDL): una licenza chiara ed efficace, forse la più comprensibile delle tre licenze GNU proprio per il fatto che non usa concetti tecnico-informatici. Attualmente, rappresenta una delle licenze più utilizzate nel mondo (assieme al set di licenze Creative Commons) per opere letterarie: basti pensare a gran parte della documen-

tazione relativa alle varie distribuzioni GNU/Linux e alla mastodontica enciclopedia virtuale Wikipedia.org.

Vale lo stesso discorso fatto per la GPL: la traduzione in italiano ha solo uno scopo informativo, mentre la versione ufficiale resta quella in inglese.

[S. Aliprandi]

Licenza per Documentazione Libera GNU

Versione 1.1, Marzo 2000
[traduzione italiana disponibile al sito www.softwarelibero.it/gnudoc/fdl.it.html]

0. PREAMBOLO
Lo scopo di questa licenza è di rendere un manuale, un testo o altri documenti scritti "liberi" nel senso di assicurare a tutti la libertà effettiva di copiarli e redistribuirli, con o senza modifiche, a fini di lucro o no. 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 che contenga una nota messa dal detentore del copyright che dica che si può distribuire nei termini di questa licenza. Con "Documento", in seguito ci si riferisce a qualsiasi manuale o opera. Ogni fruitore è un destinatario della licenza e viene indicato con "voi".
Una "versione modificata" di un documento è ogni opera contenente il documento stesso o parte di esso, sia riprodotto alla lettera che con modifiche, oppure traduzio-

ni 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 alcune sezioni secondarie i cui titoli sono esplicitamente dichiarati essere sezioni non modificabili, nella nota che indica che il documento è realizzato sotto questa licenza.
I "testi copertina" sono dei brevi brani di testo che sono elencati nella nota che indica che il documento è realizzato sotto questa licenza.
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 è stato progettato per intralciare o scoraggiare modifiche future da parte dei lettori non è trasparente. 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 ad una DTD pubblica e disponibile, e semplice HTML conforme agli standard e progettato per essere modificato manualmente. Formati opachi sono PostScript, PDF, 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 generato 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.

2. COPIE LETTERALI
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 distribuisco-

no. 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 più di 100 copie del documento, e la nota della licenza indica che esistono uno o più testi copertina, si devono includere nelle copie, in modo chiaro e leggibile, tutti i testi copertina indicati: il testo della prima di copertina in prima di copertina e il testo di quarta di copertina in quarta di copertina. Ambedue 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 il documento 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.

E' 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:

A. 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.

B. 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).
C. Dichiarare nella pagina del titolo il nome dell'editore della versione modificata in qualità di editore.
D. Conservare tutte le note sul copyright del documento originale.
E. Aggiungere un'appropriata licenza per le modifiche di seguito alle altre licenze sui copyright.
F. 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.
G. Preservare in questo avviso di licenza l'intera lista di sezioni non modificabili e testi copertina richieste come previsto dalla licenza del documento.
H. Includere una copia non modificata di questa licenza.
I. Conservare la sezione intitolata "Storia", e il suo titolo, e aggiungere a questa un elemento che riporti al minimo 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, createne una che riporti il titolo, gli autori, gli editori del documento come figurano nella pagina del titolo, quindi aggiungete un elemento che descriva la versione modificata come detto in precedenza.
J. 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'originario editore della versione cui ci si riferisce ne dà il permesso.
K. In ogni sezione di "Ringraziamenti" o "Dediche", si conservino il titolo, il senso, il tono della sezione stessa.
L. 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.
M. Si cancelli ogni sezione intitolata "Riconoscimenti". Solo questa sezione può non essere inclusa nella versione modificata.
N. Non si modifichi il titolo di sezioni esistenti come "miglioria" o per creare confusione con i titoli di sezioni non modificabili.
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 immutabili 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 defi-

nizione normativa di uno standard.
Si può aggiungere un brano fino a cinque parole come Testo Copertina, e un brano fino a 25 parole come Testo di Retro Copertina, alla fine dell'elenco dei Testi Copertina nella versione modificata. Solamente un brano del Testo Copertina e uno del Testo di Retro Copertina possono essere aggiunti (anche con adattamenti) da ciascuna persona o organizzazione. Se il documento include già un testo 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 rimpiazzare 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 Invarianti di tutti i documenti originali, senza modifiche, e si elenchino tutte come Sezioni Invarianti della sintesi di documenti nella licenza della stessa.
Nella sintesi è necessaria una sola copia di questa licenza, e multiple sezioni invarianti possono essere rimpiazzate da una singola copia se identiche. Se ci sono multiple Sezioni Invarianti 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 gli stessi aggiustamenti ai titoli delle sezioni nell'elenco delle Sezioni Invarianti nella nota di copiright della sintesi.
Nella sintesi 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 rimpiazzare 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 ASSIEME A LAVORI INDIPENDENTI
Una raccolta del documento o sue derivazioni con altri documenti o lavori separati o indi-

pendenti, all'interno di o a formare un archivio o un supporto per la distribuzione, non è una "versione modificata" del documento nella sua interezza, se non ci sono copiright per l'intera raccolta. Ciascuna raccolta si chiama allora "aggregato" e questa licenza non si applica agli altri lavori contenuti in essa che ne sono parte, per il solo fatto di essere raccolti insieme, qualora non siano però loro stessi lavori derivati dal documento. Se le esigenze del Testo Copertina della sezione 3 sono applicabili a queste copie del documento allora, se il documento è inferiore ad un quarto dell'intero aggregato i Testi Copertina del documento possono essere piazzati in copertine che delimitano solo il documento all'interno dell'aggregato. Altrimenti devono apparire nella copertina dell'intero aggregato.

8. TRADUZIONI
La traduzione è considerata un tipo di modifica, e di conseguenza si possono distribuire traduzioni del documento seguendo i termini della sezione 4. Rimpiazzare sezioni non modificabili con traduzioni richiede un particolare permesso 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 immutabili. Si può fornire una traduzione della presente licenza a patto che si includa anche l'originale versione inglese di questa licenza. In caso di discordanza fra la traduzione e l'originale inglese di questa licenza la versione originale inglese prevale sempre.

9. TERMINI
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 ne 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.

[FINE DEI TERMINI E DELLE CONDIZIONI]

Appendice: 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 subito dopo la pagina del titolo:

> Copyright (c) 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 1.1 o ogni versione
> successiva pubblicata dalla Free Software Foundation; con le
> Sezioni Non Modificabili ELENCARNE I TITOLI, con i Testi
> Copertina ELENCO, e con i Testi di Retro Copertina ELENCO.
> Una copia della licenza è acclusa nella sezione intitolata
> "Licenza per Documentazione Libera GNU".

Se non ci sono Sezioni non Modificabili, si scriva "senza Sezioni non Modificabili" invece di dire quali sono non modificabili. Se non c'è Testo Copertina, si scriva "nessun Testo Copertina" invece di "il testo Copertina è ELENCO"; e allo stesso modo si operi per il Testo di Retro Copertina.

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.

L'opencontent:
intervista a Simone Aliprandi
per il sito www.scarichiamoli.org

Questa è un'intervista che ho rilasciato nel luglio 2005 alla redazione di www.scarichiamoli.org, sito web che si propone di sostenere "l'accesso pubblico al sapere e la libera fruizione delle opere dell'ingegno" come comune denominatore fra i movimenti free software, open source, open access, open content, web accessibility. In questo testo ho avuto l'occasione di chiarire alcuni concetti per me fondamentali relativi alla condivisione dei contenuti. [S. Aliprandi]

Magnus Cedergren, autore del libro "Open content and value creation", ha così definito l'open content: «Un contenuto non prodotto per fini di profitto, spesso collettivamente, con lo scopo di renderlo disponibile a ulteriori distribuzioni e miglioramenti da parte di altri, a costo zero». Le sembra una definizione corretta?

E' una bella definizione, che sicuramente rende giustizia all'idea originaria di open content, ovvero quella riferita ai contenuti liberi da vincoli di copyright che accompagnano lo sviluppo di software libero (documentazione, manualistica). Sappiamo però che con il tempo l'espressione "open" ha assunto il ruolo di identificare una certa cultura, derivante sì dalla cultura hacker del software libero e condiviso, ma successivamente ampliatasi a vari campi della creatività e dello sviluppo. L'aspetto del "non profitto",

inoltre, appare a mio avviso non sempre aderente alla realtà del mondo della creazione sia artistico-culturale che informatica: molti autori creano opere per poterne ricavare un profitto, e questo non è un male. L'importante è che sia un profitto equilibrato e commisurato alle nuove esigenze del mercato culturale, mercato che sta subendo mastodontiche rivoluzioni in questi anni di digitalizzazione e interconnessione telematica. Stallman stesso nei suoi saggi divulgativi sul software libero sostiene che vendere software libero è bene.

Il termine copyleft ha un solo significato o può assumere significati diversi?

Anche in questo caso si può dire che esistano due significati: uno originario (e più ristretto) e uno successivo (e più ampio). Il termine nasce da un gioco di parole usato goliardicamente fra i primi sviluppatori di software libero, coloro che hanno dato vita al progetto GNU. E' infatti in seno a questo progetto che si parla di "copyleft" in senso più stretto: in questo caso una licenza è intesa come copyleft solo se richiede che chiunque utilizzi e modifichi l'opera rilasci, a sua volta, le modifiche sotto la stessa licenza; in questo modo le libertà attribuite dalla licenza si trasmettono ad oltranza per tutti gli utenti. Tale clausola nel linguaggio Creative Commons viene appunto chiamata "share alike" ovvero "condividi allo stesso modo", e non è assolutamente da confondere con il concetto di "viralità" del software libero, che è riferito ad un fenomeno diverso e prettamente informatico. Recentemente, però, si sente più spesso parlare anche di "copyleft" come fenomeno culturale cui molti autori aderiscono spontaneamente; un movimento culturale che vuole innovare i modelli tradizionali di diritto d'autore che, come molti autorevoli studiosi fanno notare, vanno ormai stretti al mondo contemporaneo, in cui i tempi e le modalità di diffusione delle opere sono notevolmente cambiati. In questo secondo significato, dunque, "copyleft" vuole indicare appunto "una forma alternativa di copyright", grazie alla quale si possono riequilibrare i rapporti di forza fra autore, editore ed utente dell'opera, senza però intaccare i principi giuridici acquisiti a livello internazionale.

Nel Suo libro “Copyleft & opencontent - l'altra faccia del copyright” Lei sostiene che la traduzione di "copyleft" in "permesso d'autore" è forzata: perché?

La risposta risulta ovvia a chiunque abbia un po' di dimestichezza con la lingua inglese. L'espressione "copyleft" ha insito in sè un DUPLICE gioco

di parole, che con la traduzione "permesso d'autore" si perde necessariamente. In inglese "left" significa "permesso" ma anche "sinistra"; e nella traduzione italiana non si coglie l'idea fondamentale di ribaltamento dei modelli che appunto si cela dietro lo scambio di "right" (che vuol dire sia "diritto" che "destra") con "left".

La filosofia che sta alla base delle licenze libere (licenze per il software e per la documentazione per il software) è identica a quella che sta alla base delle licenze open content?

Direi laconicamente che la filosofia è identica, le esigenze sono diverse. Come sarà già filtrato dalle mie precedenti risposte, sono fermamente convinto che tutto è partito dalla cultura della condivisione tipica del mondo dell'informatica indipendente. Internet stessa nasce così, quindi, tutto ciò che ha a che fare con le libertà di informazione e condivisione delle conoscenze non può rinnegare tale origine. E' anche vero, però, che il software e le opere artistico-espressive hanno caratteristiche ontologiche piuttosto distanti. Il software sottostà alla normativa sul diritto d'autore in quanto artificiosamente si è pensato di equiparare il software in versione di codice sorgente ad un'opera lettararia. Ma questo è solo un "espediente giuridico dottrinale" per poter meglio inquadrare la tutela giuridica di quel tipo di opera. Mantenendo la divisione presente nel mio libro, Copyleft & opencontent, la peculiarità che distingue un'opera software da un'opera più genericamentre artistico-espressiva sta nella sua "vocazione" tecnico-funzionale, che ovviamente non si riscontra nel caso di poesia, prosa, fotografia, musica: in questi tipi di opere prevale la "vocazione" a trasmettere lo spirito creativo dell'autore, a comunicare delle emozioni. Di conseguenza, nel software rimane fondamentale la possibilità di modificare ed intervenire sull'opera, mentre negli altri casi ciò che conta è che l'opera sia il più possibile fruibile dal pubblico. Se poi si parla di testi di espressione del pensiero, in cui più che delle emozioni si vogliono trasmettere delle opinioni (come i saggi divulgativi di Stallman, ma come nel suo piccolo può essere anche questa intervista), è evidente come sia legittimo proibire in toto le modifiche, onde evitare che il senso dell'opera venga snaturato o addirittura sovvertito. Poi, ovviamente, esistono molteplici situazioni intermedie in cui è importante concedere la possibilità di modifica nonostante la natura artistico-espressiva dell'opera: pensiamo, ad esempio, a molte forme di musica contemporanea, che hanno stretta connessione con il digitale e con l'idea di remix/sampling; o alla fotografia digitale, che si presta al ritocco e alla modifica con appositi programmi.

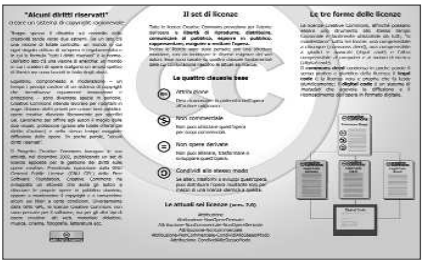
Alcuni esponenti del mondo del software libero sostengono che le licenze open content che non prevedono la rinuncia all'esercizio esclusivo di tutti i diritti di utilizzazione economica sono "not free". Ma allora anche il verbatim copying (copia letterale), assai diffuso nel mondo del software libero, è "not free"?

Sono due problemi diversi ma che nascono entrambi dalla tendenza a voler applicare integralmente i principi validi per il free software al mondo delle opere non software. Come ho già fatto rilevare, penso che, nonostante le radici culturali, etiche e filosofiche siano identiche, i due mondi abbiano esigenze concrete non sempre coincidenti. Un conto è un testo di documentazione relativo ad un software, un altro conto è una raccolta di poesie. Il verbatim copying (ovvero la formula "è concessa la copia letterale ed integrale del testo") era la formula usata dai massimi esponenti del movimento free software nei loro scritti divulgativi ed era giustificata dal fatto che in quegli articoli si condensavano opinioni personali ben precise, mirate a sensibilizzare l'opinione pubblica su certi temi: in casi come questo la libertà di modifica non avrebbe senso, anzi sarebbe addirittura controproducente. Dunque, se lo scopo principale di un'opera è la sensibilizzazione e la divulgazione di un messaggio, il fatto che chiunque possa stampare e diffondere tali opere può solo giovare all'autore. Nel caso invece in cui l'autore produca l'opera per meri scopi artistici, la situazione è diversa: a meno che si tratti di un hobbista (o di un emergente), è comprensibile che voglia riservarsi la possibilità di un profitto sulla distribuzione commerciale dell'opera. D'altronde il falegname vive grazie alla sua abilità nel trattare il legno, l'avvocato grazie alle sue competenze giuridiche... e l'artista grazie alle sue doti creative: se ci riesce, beato lui; non ci vedo nulla di anti-etico. Molta gente (a volte anche personaggi attivi nel settore) tende grossolanamente ad equiparare il concetto di copyleft con il concetto di gratuità dell'opera. Questa generalizzazione, a parere mio e di altri teorici illustri, genera un equivoco concettuale apodittico e molto pericoloso per lo sviluppo di questa filosofia.

Creative Commons

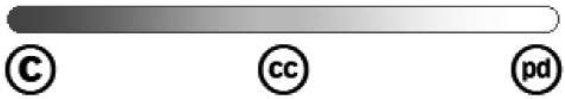
brochure informativa con i concetti base

Si riporta l'impaginato del pieghevole informativo che ho realizzato nell'autunno 2005 con lo scopo di distribuirlo in tutte le occasioni di informazione e sensibilizzazione relative a Creative Commons e all'opencontent in generale. Per approfondire i vari aspetti trattati si consiglia di navigare sui siti ufficiali di Creative Commons. [S. Aliprandi]



creative commons

un copyright flessibile
per opere creative



Creative Commons offre un insieme flessibile di protezioni e libertà per autori e artisti. Basandoci sul diritto d'autore tradizionale - "tutti i diritti riservati" - abbiamo creato un diritto d'autore su base volontaria fondato sul principio "alcuni diritti riservati". Creative Commons è un'organizzazione non-profit. Tutti i nostri strumenti sono utilizzabili gratuitamente.

"Alcuni diritti riservati"

creare un sistema di copyright ragionevole

Troppo spesso il dibattito sul controllo della creatività tende verso due estremi. Da un lato c'è una visione di totale controllo: un mondo in cui ogni singolo utilizzo di un'opera è regolamentato e in cui la formula "tutti i diritti riservati" è la norma. Dall'altro lato c'è una visione di anarchia: un mondo in cui i creatori di opere scelgono un ampio spettro di libertà ma sono lasciati in balia degli abusi.

Equilibrio, compromesso e moderazione – un tempo i principi cardine di un sistema di copyright che incentivasse equamente innovazione e protezione – sono diventate specie in pericolo. Creative Commons intende lavorare per riportarli in auge. Usiamo diritti privati per creare beni pubblici: opere creative rilasciate liberamente per specifici usi. Lavoriamo per offrire agli autori il meglio delle due visuali: protezione (grazie alle tutele offerte dal diritto d'autore) e nello stesso tempo maggiore diffusione delle opere. In poche parole, "alcuni diritti riservati".

Il Progetto Creative Commons inaugura le sua attività, nel dicembre 2002, pubblicando un set di licenze apposite per la gestione dei diritti sulle opere creative. Prendendo ispirazione dalla GNU General Public License (GNU GPL) della Free Software Foundation, Creative Commons ha sviluppato un sitoweb che aiuta gli autori a rilasciare le proprie opere in pubblico dominio, oppure a mantenerne il copyright e a consentirne alcuni usi liberi a certe condizioni. Diversamente dalla GNU GPL, le licenze Creative Commons non sono pensate per il software, ma per gli altri tipi di opere creative: siti web, materiale didattico, musica, cinema, fotografia, letteratura ecc.

Il set di licenze

Tutte le licenze Creative Commons prevedono per l'utente dell'opera la **libertà di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire o recitare l'opera.**
Inoltre le licenze sono state pensate con una struttura modulare, così da incontrare le diverse esigenze dei vari autori. Esse sono basate su quattro clausole fondamentali dalla cui combinazione nascono le attuali sei licenze.

Le quattro clausole base

Attribuzione

Devi riconoscere la paternità dell'opera all'autore originario.

Non commerciale

Non puoi utilizzare quest'opera per scopi commerciali.

Non opere derivate

Non puoi alterare, trasformare o sviluppare quest'opera.

Condividi allo stesso modo

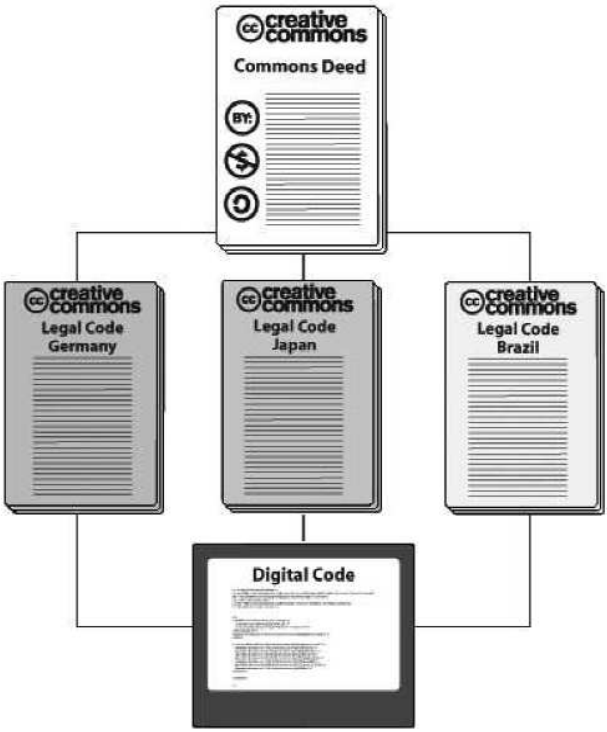
Se alteri, trasformi o sviluppi quest'opera, puoi distribuire l'opera risultante solo per mezzo di una licenza identica a questa.

Le attuali sei licenze (vers. 2.0)

Attribuzione
Attribuzione-NonOpereDerivate
Attribuzione-NonCommericale-NonOpereDerivate
Attribuzione-NonCommerciale
Attribuzione-NonCommerciale-CondividiAlloStessoModo
Attribuzione-CondividiAlloStessoModo

Le tre forme delle licenze

Le licenze Creative Commons, affinché possano essere uno strumento allo stesso tempo funzionale e facilmente utilizzabile da tutti, "si manifestano" sotto tre forme: una comprensibile a chiunque (*commons deed*), una comprensibile a giudici e avvocati (*legal code*) e l'altra comprensibile al computer e ai motori di ricerca (*digital code*).
Il **commons deed** condensa in poche parole il senso pratico e giuridico della licenza; il **legal code** è la licenza vera e propria che fa testo giuridicamente; il **digital code** è un sistema di *metadati* che agevola la diffusione e il riconoscimento dell'opera in formato digitale.



Il Progetto iCommons

Dal 2002 (data della fondazione del progetto Creative Commons negli Stati Uniti) ad oggi, è andata sempre crescendo l'attenzione per questo nuovo modo di affrontare il diritto d'autore. Sono nati così nei vari paesi del mondo movimenti e progetti ispirati agli stessi principi. Il progetto International Commons (*iCommons*) intende dunque creare una rete internazionale di divulgazione delle licenze e delle relative opere; e soprattutto intende coordinare un network di giuristi che si occupi della traduzione e dell'adattamento delle licenze ai vari contesti giuridici nazionali (il cosiddetto *porting*) nonché dello studio delle loro possibili applicazioni ed evoluzioni.

Creative Commons Italia

Dal novembre 2003 ufficialmente anche in Italia viene attivato un progetto iCommons, che fa capo a due Affiliate-Institutions locali: l'Istituto di Elettronica e di Ingegneria dell'Informazione e delle Telecomunicazioni del CNR di Torino (per gli aspetti tecnico-informatici) e il Dipartimento di Scienze Giuridiche dell'Università di Torino (per gli aspetti legali). Dopo un lavoro di community e di traduzione durato circa un anno, nel dicembre 2004 vengono presentate ufficialmente le licenze italiane. Fin da subito i "commoners" italiani dimostrano interesse e curiosità per questi nuovi strumenti e il gruppo di utenti o di semplici osservatori è in continua crescita.

Brochure a scopo divulgativo realizzata da Simone Aliprandi in collaborazione con Maurizio Borghi per Creative Commons Italia nel settembre 2005.
Parte del materiale qui riportato è tratto dai siti ufficiali Creative Commons.

Approfondimenti e contatti,
sul sito ufficiale del progetto
Creative Commons
www.creativecommons.org

oppure sul sito di Creative Commons Italia
www.creativecommons.it

NOTA
La brochure è disponibile in diversi formati alla sezione Press Kit del sito Creative Commons italiano: http://www.creativecommons.it/Stampa/PressKit

Open Access

Dichiarazione di Berlino per l'accesso aperto alla letteratura scientifica (22 ottobre 2003)

La sottoscrizione della dichiarazione di Berlino rappresenta un'importante tappa della rivoluzione che sta investendo il mondo della produzione culturale con l'avvento del digitale e dell'interconnessione telematica. E' un punto d'arrivo, un grande traguardo, perchè dimostra che tale rivoluzione è riuscita a toccare anche le alte sfere della cultura scientifica; ma è anche e soprattutto un punto di partenza con il quale intraprendere una seria e sistematica campagna di sensibilizzazione e informazione su queste tematiche.

La redazione e la sottoscrizione di questo importante testo programmatico sono state promosse dalla Max Planck Society nell'ottobre del 2003, di concerto con importanti istituzioni accademiche e scientifiche internazionali; a Messina, un anno più tardi (il 4 novembre 2004), tale testo è stato "ratificato" anche dai principali atenei italiani, nell'ambito della Conferenza dei Rettori delle Università italiane. [S. Aliprandi]

Premessa

Internet ha radicalmente modificato le realtà pratiche ed economiche della distribuzione del sapere scientifico e del patrimonio culturale. Per la prima volta nella storia, Internet offre oggi l'occasione di costituire un'istanza globale ed interattiva della conoscenza umana e dell'eredità culturale e di offrire la garanzia di un accesso universale.
Noi, i firmatari, ci impegniamo ad affrontare le sfide di Internet come mezzo funzionale emergente per la diffusione della conoscenza. Siamo certi che questi sviluppi saranno in grado di incidere significativamente tanto sulla natura delle pubblicazioni scientifiche quanto sul sistema esistente di valutazione della qualità scientifica.
In accordo con lo spirito della Dichiarazione della Budapest Open Access Initiative, la Carta di ECHO e il Bethesda Statement sull'Open Access Publishing, abbiamo redatto la Dichiarazione di Berlino per promuovere Internet quale strumento funzionale alla conoscenza scientifica generale di base e alla speculazione umana e per indicare le misure che le figure dominanti nelle politiche di ricerca, le istituzioni scientifiche, i finanziatori, le biblioteche, gli archivi ed i musei devono tenere in considerazione.

Obiettivi

La nostra missione di disseminazione della conoscenza è incompleta se l'informazione non è resa largamente e prontamente disponibile alla società. Occorre sostenere nuove possibilità di disseminazione della conoscenza, non solo attraverso le modalità tradizionali ma anche e sempre più attraverso il paradigma dell'accesso aperto via Internet. Definiamo l'accesso aperto come una fonte estesa del sapere umano e del patrimonio culturale che siano stati validati dalla comunità scientifica.
Per mettere in pratica la visione di un'istanza globale ed accessibile del sapere, il Web del futuro dovrà essere sostenibile, interattivo e trasparente. I contenuti ed i mezzi di fruizione dovranno essere compatibili e ad accesso aperto.

Definizione di contributi ad accesso aperto

Accreditare l'accesso aperto quale procedura meritevole richiede idealmente l'impegno attivo di ogni e ciascun produttore individuale di conoscenza scientifica e di ciascun depositario del patrimonio culturale. I contributi ad accesso aperto includono le pubblicazioni di risultati originali della ricerca scientifica, i dati grezzi e i metadati, le fonti, le rappresentazioni digitali grafiche e di immagini e i materiali multimediali scientifici.
Ciascun contributo ad accesso aperto deve soddisfare due requisiti:
1. L'autore(i) ed il detentore(i) dei diritti relativi a tale contributo garantiscono a tutti gli utilizzatori il diritto d'accesso gratuito, irrevocabile ed universale e l'autorizzazione a riprodurlo, utilizzarlo, distribuirlo, trasmetterlo e mostrarlo pubblicamente e a produrre e distribuire lavori da esso derivati in ogni formato digitale per ogni scopo responsabile, soggetto all'attribuzione autentica della paternità intellettuale (le pratiche della comunità scientifica manterranno i meccanismi in uso per imporre una corretta attribuzione ed un uso responsabile dei contributi resi pubblici come avviene attualmente), nonché il diritto di riprodurne una quantità limitata di copie stampate per il proprio uso personale.

2. Una versione completa del contributo e di tutti i materiali che lo corredano, inclusa una copia della autorizzazione come sopra indicato, in un formato elettronico secondo uno standard appropriato, è depositata (e dunque pubblicata) in almeno un archivio in linea che impieghi standard tecnici adeguati (come le definizioni degli Open Archives) e che sia supportato e mantenuto da un'istituzione accademica, una società scientifica, un'agenzia governativa o ogni altra organizzazione riconosciuta che persegua gli obiettivi dell'accesso aperto, della distribuzione illimitata, dell'interoperabilità e dell'archiviazione a lungo termine.

Sostenere la transizione verso il paradigma dell'accesso aperto elettronico
Le nostre organizzazioni sono interessate all'ulteriore promozione del nuovo paradigma dell'accesso aperto per offrire il massimo beneficio alla scienza e alla società. Perciò intendiamo favorirne il progresso:
· incoraggiando i nostri ricercatori e beneficiari di finanziamenti per la ricerca a pubblicare i risultati del loro lavoro secondo i principi dell'accesso aperto
· incoraggiando i detentori del patrimonio culturale a supportare l'accesso aperto mettendo a disposizione le proprie risorse su Internet
· sviluppando i mezzi e i modi per valutare i contributi ad accesso aperto e le pubblicazioni in linea, così da preservare gli standard qualitativi della validazione e della buona pratica scientifica
· difendendo il riconoscimento delle pubblicazioni ad accesso aperto ai fini delle valutazioni per le promozioni e l'avanzamento delle carriere
· difendendo il merito intrinseco dei contributi ad un'infrastruttura ad accesso aperto attraverso lo sviluppo di strumenti di fruizione, la fornitura di contenuti, la creazione di metadati o la pubblicazione di articoli individuali.
Noi riconosciamo che il passaggio all'accesso aperto modifica la disseminazione della conoscenza nei suoi aspetti legali e finanziari. Le nostre organizzazioni mirano a trovare soluzioni che sostengano futuri sviluppi degli attuali inquadramenti legali e finanziarie al fine di facilitare l'accesso e l'uso ottimale.

Traduzione di Susanna Mornati (CILEA, Segrate) e Paola Gargiulo (CASPUR, Roma), Italy. - http://www.aepic.it

Documento tratto dalla pagina web
http://www.zim.mpg.de/openaccess-berlin/BerlinDeclaration_it.pdf

APPENDICE

Bibliografia commentata della cultura open

(di Simone Aliprandi)

Questo intende essere, più che una semplice bibliografia, un testo sinottico di riferimento per avere una visuale completa (o quasi) della letteratura in lingua italiana attinente al mondo dell'informtica libera e della cultura open. I testi sono stati suddivisi in tre categorie: testi di tipo scientifico, testi informativi e divulgativi, tesi di laurea; tale suddivisione dipende esclusivamente dall'impostazione editoriale delle opere in questione e non vuole assolutamente dare a priori un giudizio di valore. Alla fine si aggiunge una quarta sezione di opere che non ho avuto modo di conoscere approfonditamente e su cui dunque non mi sento di esprimere un giudizio, ma che tuttavia meritano di essere citate. Alcuni di questi testi (purtroppo meno di quanto sia auspicabile per questo tipo di letteratura) sono distribuiti in un regime di copyleft e ciò verrà segnalato, riportando anche - dove possibile - un link da cui scaricare le versioni digitali. La disposizione all'interno delle varie categorie rispetta un ordine indicativamente cronologico.

Testi di approfondimento scientifico

- BERRA M., MEO A.R., *Informatica solidale. Storia e prospettive del software libero*, Bollati Boringhieri, Torino, 2001.

Una pietra miliare per la letteratura italiana di settore. Edito nel 2001, rappresenta il primo studio monografico sul fenomeno del software libero e sui modelli aperti di tecnologia informatica in generale. Azzeccata la scelta editoriale di unire il contributo di una sociologa (Mariella Berra) e di un informatico (Angelo Raffaele Meo): un connubio da cui nascono interessanti spunti di riflessione che tengono conto opportunamente sia delle problematiche tecnico-pratiche che di quelle culturali.

- SPAZIANTE (a cura di), *La conoscenza come bene pubblico*. CSI-Piemonte, Torino, 2004.

Questa pubblicazione contiene gli atti dell'omonimo convegno tenutosi a Torino nel novembre 2003 che è stata una delle prime occasioni italiane di tipo scientifico in cui ci si è confrontati con tali tematiche, alla luce delle nuove istanze culturali derivate da Internet e dalla cultura della condivisione; fra i nomi coinvolti: Marco Ricolfi, Lawrence Lessig, Angelo Raffaele Meo. Purtroppo tale pubblicazione è fuori commercio e il copyright dei contributi in essa contenuti

appartiene ad ogni rispettivo autore, senza alcuna nota di copyleft. Perciò resta di difficile reperibilità e si auspica una sua riedizione o una sua distribuzione in formato digitale.

- SISSA (a cura di), *Scuole in rete - Soluzioni open source e modelli UML*, Franco Angeli, Milano, 2004.

Si tratta di una fotografia sullo stato dell'arte dell'informatica libera nella scuola pubblica italiana. La curatrice è infatti responsabile dell'Osservatorio Tecnologico del Ministero dell'Istruzione, dell'Università, della Ricerca (servizio di monitoraggio e ricerca con sede a Genova) ed è un'attenta osservatrice dei rapporti fra cultura opensource e didattica; l'ente promotore delle ricerche che stanno dietro a questo libro è AICA, importante associazione di promozione della cultura informatica. Con prefazione di Angelo Raffaele Meo e introduzione di Giulio Occhini.

- LESSIG, *Cultura Libera*, Apogeo, Milano, 2005; versione italiana di *Free culture: how big media uses technology and the law to lock down culture and control creativity*, The Penguin Press, New York, 2004; disponibile anche alla pagina web www.copyleft-italia.it/pubblicazioni. [sotto licenza Creative Commons].

Un libro cult da quello che forse attualmente è il più autorevole teorico internazionale in fatto di rivisitazione dei tradizionali modelli di tutela della proprietà intellettuale. Questo libro prosegue il discorso iniziato qualche anno prima con il libro "The future of ideas", in cui Lawrence Lessig già si interrogava sul senso di alcuni paradigmi classici nel nuovo (e sempre in evoluzione) mondo del digitale e dell'interconnessione telematica. Con "Free Culture" l'autore fa un passo in più e si concentra alcuni concetti che impregnano l'analisi del mondo della comunicazione e dell'informazione contemporaneo: pirateria, creatività, proprietà (intellettuale), nuovi rapporti ed equilibri fra autori ed editori... E soprattutto pone le basi per una riflessione e una teorizzazione del concetto di beni comuni creativi, ovvero di Creative Commons e del movimento di cui lo stesso Lessig è fondatore e portavoce. Un libro scritto da un giurista ma che non risulta mai nel linguaggio troppo tecnico, tenendo conto anche di aspetti sociologici ed economici; e soprattutto che riesce ad unire ad un certo rigore scientifico anche un approccio divulgativo.

- ALIPRANDI, *Copyleft & opencontent - l'altra faccia del copyright*, PrimaOra, Lodi, 2005.

Non posso dire molto per evitare patetici “conflitti d'interesse”, se non che nemmeno io mi aspettavo tanto interesse per la mia prima opera di questo tipo. In quel libro ho cercato di compiere una panoramica completa del fenomeno culturale in questione, con un occhio di riguardo per gli aspetti giuridici, come d'altronde la mia estrazione culturale mi porta a fare. E' rilasciato sotto licenza Creative Commons ed è disponibile in versione digitale completa anche su www.copyleft-italia.it/libro.

- AA.VV., *Open Source - Atti del covegno*, Quaderni di AIDA n.13 (a cura di

M. BERTANI), Giuffrè, Milano, 2005.
E' la raccolta delle relazioni del convegno "Open Source" tenutosi a Foggia nel luglio 2004 e a cui ho avuto il piacere di assistere. Scopo del convegno era quello di effettuare un primo approccio scientifico al fenomeno opensource in vista del più articolato e specifico convegno "Open source, software proprietario e concorrenza" (Pavia, settembre 2004). La pubblicazione contiene alcuni interessanti saggi che si occupano degli aspetti filosofici e culturali del fenomeno e inizia ad aprire la strada ad un'analisi più tecnica (giuridica ed economica), compiuta nel convegno successivo.

- AA.VV., *AIDA - Annali italiani del diritto d'autore e dello spettacolo* (rivista giuridica a cura di L.C. Ubertazzi), anno 2004, Giuffrè, Milano.
E' l'edizione annuale della rivista AIDA, uno dei massimi punti di riferimento per la dottrina del diritto industriale; è curata e diretta dal Prof. Luigi Carlo Ubertazzi, docente universitario di grande prestigio che unisce un approccio classico e tradizionale ad una costante curiosità per le istanze innovative del diritto industriale e più specificamente del diritto d'autore. In questa pubblicazione sono raccolte le relazioni del convegno "Open source, software proprietario e concorrenza" tenutosi a Pavia nel settembre 2004 e che ha riunito i maggiori nomi della dottrina industrialistica: Ricolfi, Ghidini, Ammendola, Guglielmetti, Spada. Nelle 530 pagine della prima parte della rivista, monograficamente dedicate all'opensource, vengono indagate con la massima profondità tutte le problematiche giuridico-economiche del fenomeno, anche se solo in ambito informatico. Attualmente in Italia non esiste altra pubblicazione così scientificamente elevata e dettagliata in quest'ambito.

- GRUPPO LASER (a cura di), *Il sapere liberato*, Feltrinelli, Bologna, 2005.
Dietro il nome collettivo "Gruppo Laser" si celano alcuni ricercatori scientifici indipendenti che hanno voluto indagare alcuni aspetti interessanti quanto controversi relativi alla libertà di accesso al sapere scientifico. Si compie un'introduzione chiara e rigorosa al concetto di proprietà intellettuale con le relative critiche dottrinali e si parla per la prima volta di una sorta di copyleft anche in campo brevettuale. In appendice si riportano i testi delle dichiarazioni d'intenti promosse e sottoscritte da quello che va definendosi come "movimento Open Access".

Testi informativi e divulgativi

- AA.VV. (a cura di DIBONA, OCKAM E STONE), *Open Sources - Voices from The Open Source Revolution*, O'Reilly, U.S.A., 1999. Edizione italiana: *Open Sources - Voci dalla rivoluzione open source*, Apogeo, Milano, 1999; disponibile anche in versione telematica gratuita (non integrale) su www.apogeonline.com/ebook/90016/scheda.
E' la prima opera editoriale in assoluto in cui si parla di Open Source ed è infatti stata pubblicata pochi mesi dopo l'inaugurazione della Open Source Initiative. Contiene alcuni dei saggi divulgativi più importanti del settore e rappresenta

ancora - nonostante siano passati alcuni anni dalla pubblicazione - uno dei testi più completi per affacciarsi e capire il fenomeno dell'informatica libera. Vi partecipano quasi tutti i grossi nomi del movimento: Richard Stallman (fondatore del Progetto GNU), Bruce Perens (autore della Open Source Definition), Eric Raymond (fondatore della Open Source Initiative), Linus Torvalds (creatore del kernel Linux). E' anche uno dei primi esempi di editoria opencontent dato che alcuni dei saggi contenuti (purtroppo solo la minoranza) sono rilasciati sotto licenza GPL o con permesso di copia letterale.

- TORVALDS & DIAMOND, *Rivoluzionario per caso. Come ho creato Linux (solo per divertirmi)* [seconda ed.], Garzanti, 2005 (prima edizione: 2001).

Un testo un po' biografico, un po' autobiografico in cui si narrano gli aspetti anche più intimi e personali della nascita del kernel Linux per opera di uno studente di informatica finlandese, che ha dato il nome a uno dei fenomeni informatici più famosi della nostra era. Si riportano curiosi scambi di e-mail fra Torvalds e i suoi compagni di avventura informatica. Peccato per la mancata distribuzione sotto licenza opencontent.

- STALLMAN, *Free Software, Free Society: The Selected Essays of Richard M. Stallman*, Free Software Foundation, U.S.A., 2002; ed. italiana *Software libero, pensiero libero: saggi scelti di Richard Stallman* (vol. 1 e vol. 2), a cura di B. Parrella e Ass. Software Libero, Stampa Alternativa, 2003; disponibile anche alla pagina web www.copyleft-italia.it/pubblicazioni. [con permesso di copia letterale].

Una raccolta antologica dei più significativi saggi divulgativi di quello che è considerato unanimemente il guru della cultura dell'informatica libera e della libera condivisione delle conoscenze. Tutti i testi hanno un'efficacia dialettica elevatissima e l'autore riesce a trattare tematiche disparate (dall'informatica, al diritto, dalla sociologia, all'economia) con un buon rigore terminologico; anche se - come prevedibile - è difficile epurare i testi da un certa componente ideologica che traspare costantemente. L'opera nell'edizione americana si presenta in un unico volume, impreziosito dalla prefazione di Lawrence Lessig (altro grandissimo teorico del movimento) che purtroppo non è stata mantenuta e tradotta nell'edizione italiana.

- WILLIAMS, *Free as in freedom*, O'Reilly, U.S.A., 2002; edizione italiana *Codice libero: Richard Stallman e la crociata per il software libero* (traduzione di B. Parrella), Apogeo, Milano, 2003; disponibile anche su www.copyleft-italia.it/pubblicazioni. [sotto licenza GNU FDL]

Nasce con l'intento di essere una sorta di biografia di Richard Stallman, ma fa qualcosa di più: cioè traccia un'interessante dinamica storica del free software e dell'open source, senza entrare mai eccessivamente in particolari privati della vita del guru (il quale - tra l'altro - pare non aver mai dato il suo benestare alle pagine biografiche del libro). Inoltre riporta numerosi brani di articoli e interviste ai principali esponenti del movimento, cogliendone sem-

pre in modo obbiettivo il pensiero.

- SCIABARRÀ, *Il software Open Source e gli standard aperti*, McGraw-Hill Companies, 2004.

Un testo di tipo tecnico-informatico che cerca però di arrivare anche ad un pubblico non necessariamente elitario. Dopo due utilissimi capitoli di introduzione concettuale al fenomeno opensource e di storia del movimento, l'autore ci presenta con chiarezza le particolarità e i vantaggi dei principali software Open Source (OpenOffice, KDE, Linux, MySQL, Mozilla, Apache), addentrandosi infine nei segreti per lo sviluppo e la manutenzione di questi strumenti.

- NMI CLUB (a cura di), *NoSCOpyright*, Stampa Alternativa, 2004; disponibile anche alla pagina web www.copyleft-italia.it/pubblicazioni. [sotto licenza Creative Commons]

Il libro ricostruisce nei particolari la dinamica storica del sistema UNIX e di come da una sua costola sia nato il progetto GNU e successivamente il kernel Linux. Si pone particolare attenzione su un caso giuridico ancora poco chiaro che vede un'impresa di sviluppo software (la SCO appunto) minacciare azioni legali a destra e a manca, fondandosi su una non ben specificata violazione di copyright da parte degli sviluppatori del sistema GNU/Linux. Il problema è particolarmente delicato e richiede la massima attenzione perchè potrebbe rivelare una pericolosa falla nel modello di distribuzione copyleft. Molto utile la cronologia che viene riportata in appendice.

- IPPOLITA, *Open non è free. Comunità digitali tra etica hacker e mercato globale*, Eleuthera, 2005; disponibile anche alla pagina web www.copyleft-italia.it/pubblicazioni. [sotto licenza Creative Commons]

E' un libro ben scritto e ben pensato, anche se a mio avviso si perde un po' troppo in concettualizzazioni e astrazioni, aiutando a riflettere su vari aspetti ma dando poche risposte; probabilmente gli autori si sono posti espressamente un intento più che altro filosofico e sociologico. E' scritto da un autore virtuale e collettivo che si cela dietro il nome di Ippolita.

- VALVOLA SCELSI, *No copyright - nuovi diritti nel 2000*, Shake Underground, Milano, 1994.

Una pubblicazione pionieristica in ambito italiano uscita in tempi non sospetti, quando non si parlava ancora di software libero, di Internet, di condivisione di file digitali. Eppure apre la strada a questa cultura, raccogliendo e commentando alcuni interessanti saggi sia di tipo giuridico sia di tipo sociologico, che aiutano a riflettere sull'opportunità di un regime rigido e tradizionale di diritto d'autore.

Altri testi (solo citazione)

- BASSI, *Open Source - analisi di un movimento*, Apogeo, Milano, 2000; disponibile anche alla pagina web *www.apogeonline.com/ebook/90026/scheda.xhtml* oppure alla pagina web www.copyleft-italia.it/pubblicazioni.

- LESSIG, *The Future of Ideas: the fate of the commons in a connected world*,

Random House, U.S.A., 2001.

- MEO, *Software libero e open source*, Mondo Digitale, 2002; disponibile anche su www.aiscris.it/open_source.php oppure su www.dvara.net/HK/open.asp.

- AA.VV. (a cura di Mari e Romagnolo), *Revolution OS*, Apogeo, Milano, 2003.

- GRANDI, *Introduzione al mondo del software libero e dell'opensource*, 2004; pubblicazione solo digitale, disponibile alla pagina web http://it.tldp.org/doc-it/intro-swlibero [sotto duplice licenza GNU FDL e Creative Commons].

- MUFFATTO & FALDANI, *Open Source: strategie, organizzazione, prospettive*, Il mulino, 2004.

- ST. LAWRENT, *Understanding open source and free software licensing*, O'Reilly, 2004. Disponibile anche su www.copyleft-italia.it/pubblicazioni [sotto licenza Creative Commons].

- MARANDOLA, *Il nuovo diritto d'autore*, DEC, 2005.

- BALKIN, DAVIS, LEMLEY, LESSIG, SAMUELSON, *I diritti nell'era digitale: libertà d'espresisone e proprietà intellettuale*, Diabasis, Reggio Emilia, 2005.

- DI CORINTO, *Revolutions OS II - Software libero, proprietà intellettuale, cultura e politica*, Apogeo, Gennaio 2006.

Tesi di laurea (solo citazione)

[tutte le tesi qui citate sono accessibili dalla pagina web www.copyleft-italia.it/pubblicazioni]

- RAGUSA, *Sviluppo e ordinamento istituzionale nel mercato del software: il modello del software libero*, anno academico 2001-2002, Facoltà di Scienze Politiche. [sotto licenza Creative Commons].

- DIDONÈ, *Modelli di business per il software libero*, 2001, Facoltà di Economia. [sotto licenza GNU FDL]

- MASSARA, *Linux*, anno accademico 2001-2002, Facoltà di Giurisprudenza.

- SANTO, *Le licenze pubbliche GNU*, anno accademico 2001-2002, Facoltà di Giurisprudenza.

- ZANOTTI, *Open Source Initiative*, anno accademico 2002-2003, Facoltà di Giurisprudenza.

- GAMBARDELLA, *La filosofia Open Source: Benefici e modelli di sviluppo aziendali. Analisi di casi di interfacciamento tra mondo Microsoft e mondo Linux/OpenSource*, anno accademico 2004-2005, Facoltà di Economia.

- ZOPPELLETTO, *Il modello opensource vs il modello proprietario nel mercato del software*, anno accademico 2004-2005, Facoltà di Economia. [sotto licenza GNU FDL]

- TODON, *Il software libero/open source: una dimensione sociale*, aprile 2005, Facoltà di Scienze della Comunicazione. [sotto licenza Creative Commons]

Siti web di maggiore rilevanza

I siti web che trattano questi argomenti - al di là dei siti ufficiali delle grandi realtà che si fanno portavoce e promotrici del movimento - sono innumerevoli e ogni giorno spuntano nella rete come funghi in un bosco a settembre. Questo per dire che un lavoro di classificazione e recensione come quello fatto per le pubblicazioni cartacee è pressochè impossibile (anche se volessimo limitarci all'area italiana) e sarebbe fatica sprecata dato che nel giro di poco risulterebbe obsoleto.

Come avrete notato ogni articolo e documento riportato in questo libro è corredato dei link necessari per l'approfondimento delle tematiche di volta in volta trattate; quindi qui non mi preoccuperò di riportare tutti i riferimenti già citati. Quello che posso fare è - più che altro - segnalare gli indirizzi web a mio avviso più rilevanti e invitarvi a seguire i percorsi interattivi che questi siti vi forniranno; molti di questi infatti sono dei veri e propri portali, aggiornati assiduamente e dai quali potrete monitorare la nascita in rete di nuove realtà rilevanti. Inoltre, alla pagina www.copyleft-italia.it/links trovarete una versione il più possibile aggiornata di questa 'sitografia'.

ENTI, ASSOCIAZIONI E GRUPPI SPONTANEI (internazionali)

www.creativecommons.org - Creative Commons

www.fsf.org - Free software foundation

www.softwarefreedom.org - Software Freedom Law Center

www.opensource.org - Open source initiative

www.eff.org - Electronic Frontier Foundation

www.hipatia.info - Hipatia, Conoscenza libera in azione per i popoli del mondo

ENTI, ASSOCIAZIONI E GRUPPI SPONTANEI (Italia)

www.linux.it - Italian Linux Society

www.softwarelibero.it - Associazione Software Libero (Firenze)

www.creativecommons.it - Creative Commons Italia (Torino)

www.osservatoriotecnologico.net - Osservatorio tecnologico MIUR (Genova)

www.consorziocirs.it - Consorzio CIRS (aziende attive in ambito open source)

www.pluto.it - Progetto di sviluppo e promozione di tecnologie libere

www.openlabs.it - Openlabs (associazione culturale - Milano)

evangelion.polito.it - Netstudent (associazione culturale - Torino)

www.lugroma.org - Linux Users Group di Roma

www.torlug.org - Tor Vergata Linux Users Group

www.lugcr.it - Linux Users Group di Cremona

filibusta.crema.unimi.it - Linux Users Group di Crema

www.linux-club.org - Associazione culturale Linux Club

www.diago.it/creattiva - CreAttiva, gruppo di attivismo creativo

PROGETTI

www.wikipedia.org - enciclopedia libera multilingue (interamente sotto copyleft)

www.wiktionary.org - dizionario libero

www.gnu.org - Gnu Not Unix (progetto capostipite del software libero)

www.sciencecommons.org - Progetto Science Commons

cyber.law.harvard.edu/openlaw - Openlaw Project (studi giuridici sul mondo copyleft)

www.gnutemberg.org - raccolta, distribuzione e stampa di documentazione libera

ildp.pluto.it - Italian Linux Documentation Project (del progetto PLUTO)

www.pluto.it/journal - PLUTO Journal, periodico d'informazione on line sul mondo del software libero

www.dschola.it - Le scuole per le scuole (open source in ambito didattico)

edu.os3.it - Progetto EDU (materiale didattico libero)

www.scarichiamoli.org - promozione dell'accesso pubblico al sapere e la libera fruizione delle opere dell'ingegno

www.costozero.org - movimento per la gratuità del diritto alla comunicazione

www.aepic.it - Soluzioni avanzate per l'editoria elettronica

www.openarchives.it - sito della Open Archive Initiative

www.ippolita.net - Tempo di web semantico

SITI D'INFORMAZIONE E CULTURA LIBERA

www.copyleft-italia.it - Il primo sito italiano dedicato al fenomeno copyleft

www.liberacultura.it - Libera cultura, libera conoscenza.

www.kronstadt.it - Rivista elettronica e cartacea di approfondimento culturale

www.fioriblu.it - Rivista elettronica di giardinaggio socioculturale

www.annozero.org - informazione libera sul software libero

www.ilsecolodellarete.it - Il secolo della Rete. For a free knowledge society

www.autistici.it - Socializzare saperi, senza fondare poteri

www.e-laser.org - sito del Gruppo LASER

www.punto-informatico.it - Il quotidiano della rete dal 1996 (articoli interamente rilasciati sotto licenza Creative Commons)

www.radiodigitale.info - Radio on line e sito rilasciato sotto licenza Creative Commons

SITI PERSONALI e BLOGS

www.stallman.org - Richard M. Stallman, ideatore progetto GNU e scrittore

www.lessig.org - Lawrence Lessig, giurista e scrittore

www.catb.org/~esr/ - Eric. S. Raymond, fondatore Open Source Initiative

www.perens.com - Bruce Perens, autore della Open Source Definition

www.cs.helsinki.fi/u/torvalds - Linus Torvalds, creatore del kernel Linux

www.beppegrillo.it - Beppe Grillo, comico e opinionista

www.caravita.biz - Beppe Caravita, giornalista

www.attivissimo.net - Paolo Attivissimo, divulgatore informatico e cacciatore di bufale

www.dicorinto.it - Arturo Di Corinto, giornalista e docente universitario

www.copyleft-italia.it/ali - Simone Aliprandi, giurista osservatore della cultura open

ARCHIVI DI OPERE IN COPYLEFT

www.ibiblio.org/eldritch - Eldritch Press (interi libri di letteratura in pubblico dominio)

www.oreilly.com/openbook - Open books Project (archivio dell'editore statunitense O'Reilly)

www.plos.org - PLOS, Public Library Of Sciences (articoli scientifici in copyleft)

www.tldp.org - The Linux Documentation Project (documentazione copyleft relativa a GNU/Linux e al software libero in generale)

cdrom.gnutemberg.org - GNUtemberg CD-ROM (raccolta di testi copyleft)

www.liberliber.it - Progetto Liber Liber (biblioteca telematica ad accesso gratuito)

www.openphoto.net - Openphoto (archivio di immagini royalty free)

www.flickr.com/creativecommons - sezione del sito Flickr con immagini opencontent

openmusic.linuxtag.org/modules/freecontent/content/openmusic - Openmusic, music for a free world (compilation di brani musica in copyleft)

www.negativland.com - archivio di musica no-copyright e copyleft

www.anomolo.com - archivio italiano di musica no-copyright

creativecommons.org/wired - The Wired CD (compilation sotto licenze opencontent)

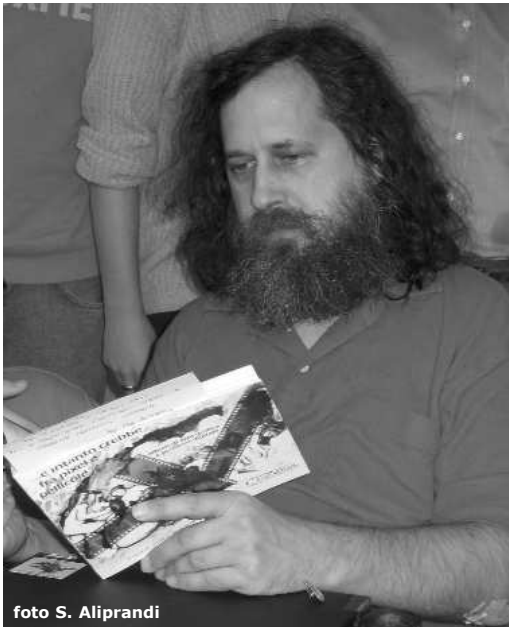
www.musique-libre.com - archivio francese di musica libera

www.archive.org - Universal access tu human knowledge

Ma che faccia avrà...?

Un'altra curiosità che secondo me vale la pena di colmare è quella di vedere che facce hanno i personaggi di cui tanto si sente parlare. Qui riporto un paio di foto che ho scattato io personalmente e un paio trovate altrove ma rilasciate sotto copyleft.

Richard Stallman,
fotografato in occasione del convegno "Free software e libertà nella ricerca scientifica" tenutosi a Milano, Università Statale, il 20 aprile 2004 (v. www.copyleft-italia.it/eventi). In questa foto Stallman sta sfogliando il libro "*...e intanto crebbe fra pixel e pellicole*" (a cura di S. Aliprandi), prima opera editoriale in Italia rilasciata sotto licenza Creative Commons (v. www.copyleft-italia.it/libri).



Linus Torvalds,
foto tratta dal numero 2 (ottobre '03) della rivista Open Source, edita da Systems e interamente rilasciata sotto licenza GNU Free Documentaion License.

Eric Raymond,
foto tratta dal numero 6 (febbraio '04) della rivista Open Source, edita da Systems e interamente rilasciata sotto licenza GNU Free Documentaion License.

Lawrence Lessig,
fotografato in occasione della presentazione ufficiale delle licenze Creative Commons Italiane, tenutasi a Torino, Fondazione Giovanni Agnelli, il 16 dicembre 2004 (v. www.copyleft-italia.it/eventi).

L'autore

Simone Aliprandi nasce nel 1979. Frequenta la facoltà di Giurisprudenza a Pavia dove nel 2003 ottiene la laurea a pieni voti con una tesi in Diritto industriale e specificamente sul diritto d'autore e il mondo opensource. Attualmente è iscritto al registro dei Praticanti avvocati di Lodi ed è laureando in Scienze della pubblica amministrazione sempre presso l'Università di Pavia.

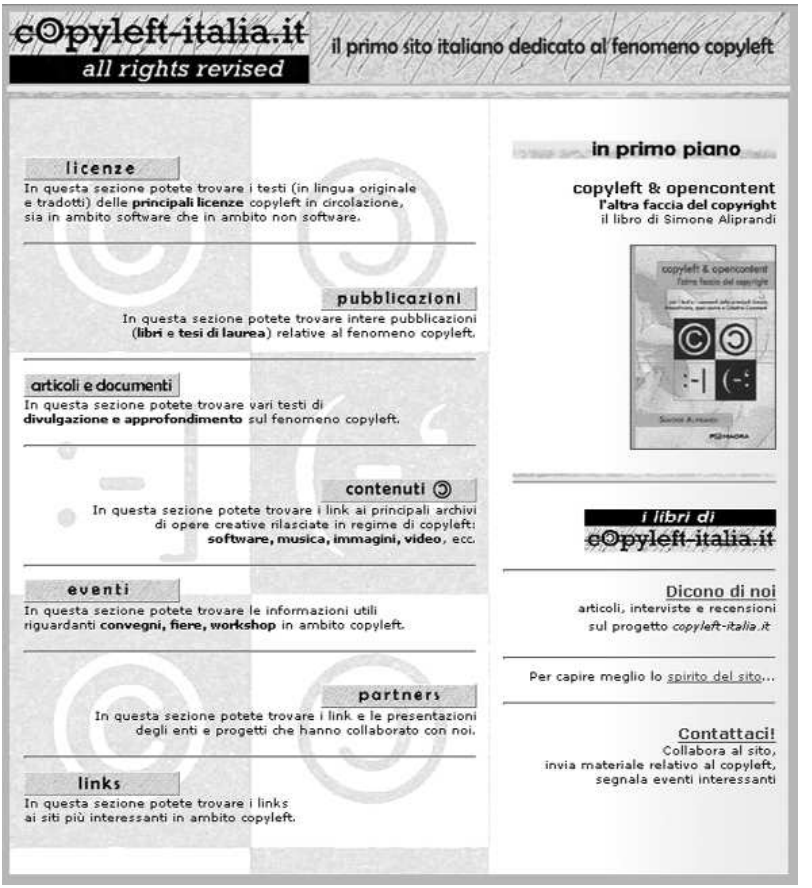
Fin dai tempi del liceo si interessa di materie giornalistiche e comunicazione, collaborando in vari progetti editoriali e organizzando eventi d'intrattenimento e approfondimento culturale. Nel 2002 fonda l'associazione culturale "CreAttiva - gruppo di attivismo creativo" di cui è tuttora presidente e con la quale pubblica lavori di vario tipo ed organizza corsi di formazione.

Si tiene costantemente in contatto con le principali realtà del mondo opensource ed opencontent, svolgendo attività di consulenza per associazioni e imprese del settore. Collabora con alcune riviste specializzate ed è l'ideatore responsabile del sito www.copyleft-italia.it, interamente dedicato al fenomeno copyleft come nuovo modello di gestione dei diritti d'autore e in cui sono pubblicati numerosi suoi testi d'approfondimento.

Collabora stabilmente con l'editore PrimaOra, per il quale - oltre a coordinare la promozione e la distribuzione - ha pubblicato come autore altri due titoli: "Copyleft & opencontent - L'altra faccia del copyright" e "Risvegli d'inverno - Allucinazioni poetiche dalla stanza della Leda".

Maggiori informazioni e contatti sul suo sito personale:
www.copyleft-italia.it/ali

www.copyleft-italia.it



Copyleft-italia.it è una sorta di osservatorio sui nuovi modelli di gestione dei diritti d'autore che derivano dalla cultura dell'informatica libera e della condivisione dei contenuti.
E' nato dall'idea di Simone Aliprandi e funge sia da raccolta di materiale utile, sia da portale per conoscere meglio le realtà (associazioni, eventi, progetti) che giorno per giorno arricchiscono questo panorama.
E' possibile collaborare al sito inviando articoli, tesi di laurea, interi libri, oppure suggerendo links e proponendo progetti.

Da anni il Pinguino, nato e rimasto senza proprietari, è il simbolo del sistema operativo che prende il nome dal suo giovane inventore, Linus Torvalds. Un sistema operativo che chiunque può implementare, sviluppare e accrescere gratuitamente adattandolo alle proprie esigenze, alla sola condizione che i contributi restino a disposizione di tutti. Sicuramente chi decide di passare all'Open Source compie la scelta etica di fondo di contrastare il monopolio del diritto proprietario che, proprio come il brevetto sui farmaci, nega l'accessibilità a milioni di persone, riconoscendo grande valore sociale all'accesso aperto dei saperi.

Le grandi questioni di fondo sono tutte contenute nelle leggi che regolano lo sviluppo dell'innovazione tecnologica e in quelle più recenti in materia di antiterrorismo. Come la legge sul copyright, che protegge e favorisce gli interessi delle grandi major ostacolando le pratiche di condivisione collettiva, o le misure a tutela della privacy, che obbligano alla schedatura degli utenti di internet, o le politiche della SIAE, che continua a mantenere il monopolio della gestione dei diritti d'autore e delle opere. Risposte che devono venire dalla politica, dai cittadini che dovrebbero indirizzarla, impegnandosi a non fare prevalere forme di limitazione della democrazia e a non permettere che le idee e i valori diventino merci assoggettate al mercato economico.

Una scelta democratica fondamentale che debbono compiere in primo luogo le Pubbliche Amministrazioni, le Università, la libera ricerca, perché non ci può essere vera autonomia di governo, per uno Stato e i suoi Enti, senza il pieno controllo dei mezzi con i quali operano. Passare all'Open Source significa migliorare i propri servizi e abbattere i costi elevatissimi delle licenze, ma anche

e soprattutto iniziare a percorrere la strada di una vera innovazione, formando nuove professionalità e nuove competenze tra i giovani, che devono potersi riappropriare del sapere tecnologico.

Con la pubblicazione di questo libro e con l'apertura del Net Open Source, vogliamo dare il nostro piccolo contributo affinché si cominci a ragionare diffusamente e concretamente su questo cambiamento possibile.

Elisa Romagnoli
Assessore alle Politiche Giovanili
del Comune di Modena

I progetti del Comune di Modena sulla Rete:
www.stradanove.net
www.comune.modena.it/netgarage
www.comune.modena.it/biblioteche/holden
www.comune.modena.it/biblioteche/holden/scritmet.htm

www.prosa.it

PROSA Progettazione Sviluppo Aperto (questo il nome per esteso) è stata creata nel 1998, come azienda specializzata nel software libero, per promuovere l'uso di quest'ultimo negli ambienti professionali.

E' stata fondata da attivisti GNU e Linux di vecchia data, che hanno imposto uno statuto che obbliga l'azienda a usare, sviluppare e vendere solo software libero, come definito dalla Free Software Foundation. Questo la rende unica al mondo.

Da quando è nata, Prosa fornisce servizi di supporto tecnico di alto livello per il sistema operativo GNU/Linux e il software libero in generale. Prosa con le sue persone e le sue idee è parte integrante della comunità del software libero: lavorando con Prosa si ha direttamente accesso alle migliori risorse umane che la comunità è in grado di fornire.

Sono disponibili servizi di supporto (Support Services) per il monitoring e la gestione completamente automatizzata di sistemi Linux e servizi di consulenza (Professional Services), per ottenere personalizzazioni complete di software libero, ivi compreso il porting di applicazioni, device driver, tecnologie clustering.

Prosa ha coltivato al suo interno forti competenze riguardo sistemi embedded, kernel linux, realtime linux, device drivers. Ha creato e supporta lo sviluppo di una distribuzione embedded di Linux (www.etlinux.org).

Prosa ha fornito soluzioni a clienti GLOBAL 1000 e molto del lavoro fatto da Prosa sia per i propri clienti che per la comunità si può reperire attraverso le risorse che la società mette a disposizione.

Maggiori informazioni sono disponibili
sul sito web di Prosa www.prosa.it.

Soli e squallidamente dispersi
in questo inverno di ombre sottili
che persiste deciso a colpirci i neuroni.
Immensa la via di fuga
che appare di fronte al poeta isolato:
l'unica possibile prospettiva
disegnata da un dio ribelle.
Intanto sento la tua voce chiamarmi,
ragazza metropolitana dagli occhi taglienti,
e vedo i miei timori muoversi,
schifosamente curiosi e incoscienti,
verso una nuova frontiera di emozioni:
effimere ma irrinunciabili.

brano tratto da
Risvegli d'inverno
allucinazioni poetiche dalla stanza della Leda
www.diago.it/risvegli

il primo libro
di Simone Aliprandi

un testo chiaro e completo
sulla cultura opensource
e sulle nuove forme di
copyright che ne derivano

con i testi e i commenti delle
principali licenze
freesoftware e opencontent

rilasciato sotto licenza
Creative Commons

edito da PrimaOra (Lodi)
nel marzo 2005
prezzo di copertina: €12
pagine: 176
ISBN: 88-901724-0-1

Con questo saggio si può disporre finalmente di un testo completo e onnicomprensivo riguardante la nuova cultura opensource e opencontent, derivata dall'avvento rivoluzionario di Internet e della multimedialità.
Si raccolgono gli elementi storici che hanno portato la scienza informatica sulla strada di GNU/Linux e del software libero in generale e si commentano nel dettaglio i principi giuridici nati o modificati da tale filosofia; inoltre - ecco l'aspetto più originale dell'opera - si esaminano e ipotizzano i riflessi che essa può apportare nell'ambito della creatività in senso più ampio (manualistica informatica, editoria multimediale, musica...).
E' questo il panorama in cui si realizza la dicotomia fra il copyright tradizionale e il suo più interessante "figlio degenere": il copyleft, fenomeno che ormai si propone come innovativo modello di gestione dei diritti di proprietà intellettuale nell'attuale mondo multimediale e interconnesso.

informazioni su come avere una copia
di questo libro e di altri libri simili
al sito www.copyleft-italia.it/libri

Presto disponibile nelle librerie, il nuovo libro di Simone Aliprandi

Copyleft - dalla teoria alla pratica
guida all'uso delle licenze opencontent

informazioni in anteprima sul sito www.ndanet.it

Un'utilissima e completa antologia di articoli divulgativi e documenti ufficiali (esplicati e commentati) per affacciarsi a questa interessante cultura emergente.

Contiene materiale informativo su:

- *il copyleft e i nuovi modelli per il diritto d'autore*
- *i brevetti software e i relativi problemi*
- *il progetto GNU e la Free Software Foundation*
- *il software Open Source*
- *i sistemi GNU/Linux e le varie distribuzioni*
- *la documentazione libera*
- *Creative Commons e l'opencontent in generale*

e le traduzioni italiane di documenti come:

- *la licenza GNU GPL*
- *la Open Source Definition*
- *la Licenza GNU FDL*
- *la dichiarazione di Berlino sull'OpenAccess*

ISBN 88-901724-3-6
9 788890 172434 >

€ 8 (i.v.a. inclusa)

---

questa pubblicazione è stata realizzata grazie al contributo di

NET OPEN SOURCE

Comune di Modena
Assessorato alle Politiche Giovanili
www.comune.modena.it/netgarage

PROSA
PROgettazione Sviluppo Aperto
www.prosa.it