# The Open Source Definition

**di Bruce Perens**

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 funziona 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 rimozione 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 precedenti 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.*

7. 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.*

*8.* 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.*

9. 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.

   *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 e una derivato 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 e 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 | | | | |
| LGPL | **V** | | | |
| BSD | **V** | **V** | | |
| NPL | **V** | **V** | | **V** |
| MPL | **V** | **V** | | |

| | | | | |
|---|---|---|---|---|
| Dominio Pubblico | **V** | **V** | **V** | |

**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.