Indice generale

Perché la scuola deve usare esclusivamente software libero

di Richard Stallman

Le istituzioni didattiche in genere, dalla scuola materna all'università, hanno il dovere morale di insegnare solo il software libero.

Tutti gli utenti informatici devono insistere con il software libero: esso offre agli utenti la libertà di poter controllare il proprio computer; con il software proprietario il programma fa quanto stabilito dal suo proprietario o sviluppatore, non quel che vuole l'utente. Il software libero offre inoltre agli utenti la libertà di poter collaborare tra loro. Queste caratteristiche si applicano alla scuola come a qualsiasi altro soggetto. Lo scopo di questo articolo è di evidenziare le caratteristiche che si applicano in modo specifico al settore dell'istruzione.

Il software libero consente alle scuole di risparmiare, anche se questo è un beneficio secondario: il risparmio è dovuto al fatto che il software libero offre agli istituti scolastici, come ad ogni altro utente, la libertà di copiare e ridistribuire il software; di conseguenza il sistema didattico può farne copie per tutte le scuole, e ogni scuola può installare il programma in tutti i suoi computer, senza obbligo di pagare per farlo.

Questo beneficio è importante, ma assolutamente non prioritario, perché le questioni etiche sono molto più importanti. Convertire le scuole al software libero è più che un semplice "miglioramento" della didattica: è la differenza tra una didattica buona e una didattica sbagliata. Quindi occupiamoci delle questioni più profonde.

La scuola ha una missione sociale: insegnare a chi studia a diventare cittadino di una società forte, capace, indipendente, collaborativa e libera. Dovrebbe promuovere l'uso del software libero così come promuove la protezione dell'ambiente, o il diritto di voto. Se la scuola insegna l'uso del software libero, potrà sfornare cittadini pronti a vivere in una società digitale libera. Ciò aiuterà la società nel suo insieme ad evitare di essere dominata dalle multinazionali.

Al contrario, insegnare un programma non libero crea dipendenza, e questo va contro la missione sociale delle scuole: le scuole non dovrebbero mai farlo.

Perché alcuni produttori di software proprietario offrono copie gratuite(1) alle scuole? Perché vogliono *usare* le scuole per creare dipendenza nei confronti dei loro prodotti, come i produttori di tabacco che distribuiscono sigarette gratis ai ragazzini(2). Una volta che gli studenti saranno diventati adulti, queste aziende non offriranno più alcuna copia gratuita a loro o ai loro datori di lavoro. Chi sviluppa dipendenza dovrà pagare.

Il software libero consente a chi studia di poter imparare il funzionamento di un programma. Alcuni studenti, che hanno un talento innato per la programmazione, quando diventano adolescenti vogliono imparare tutto quanto c'è da sapere riguardo al computer e al software. Sono animati dalla fervida curiosità di leggere il codice sorgente dei programmi che usano ogni giorno.

Il software proprietario respinge la loro sete di conoscenza; dice loro: "La conoscenza che stai cercando è un segreto: vietato imparare!". Il software proprietario è nemico dello spirito didattico, quindi non lo si può tollerare nelle scuole, se non utilizzato al solo scopo di studiarne il funzionamento interno.

Il software libero incoraggia tutti ad imparare. La comunità del software libero rifiuta "il sacerdozio della tecnologia", secondo cui il grande pubblico va tenuto nell'ignoranza sul funzionamento della tecnologia; noi incoraggiamo gli studenti di ogni età e situazione a leggere il codice sorgente e ad imparare tutto quello che vogliono sapere.

La scuole che usano software libero devono permettere ai bravi studenti di programmazione di progredire. Per imparare a scrivere del buon software, gli studenti devono potere leggere e scrivere una grande quantità di programmi reali, di uso concreto. Si impara a scrivere codice buono e chiaro solo leggendo e scrivendo molto codice. Solo il software libero permette questo.

Come si impara a scrivere codice per programmi complessi? Scrivendo tante piccole modifiche per programmi complessi esistenti. Il software libero lo permette, il software proprietario lo vieta. Qualsiasi scuola può dare ai propri studenti la possibilità di diventare esperti di programmazione, purché usi software libero.

La più profonda motivazione in sostegno all'utilizzo del software libero nella scuola è per la formazione morale. Dalla scuola ci si aspetta l'insegnamento di fatti fondamentali e di capacità utili, ma ciò non ne esaurisce il compito. Missione fondamentale della scuola è quella di insegnare a essere cittadini coscienziosi e buoni vicini, e quindi anche ad aiutare gli altri. In campo informatico ciò significa insegnare la condivisione del software. Le scuole, a cominciare dalle elementari, dovrebbero dire agli studenti: “Se porti a scuola del software devi dividerlo con gli altri bambini. Devi mostrare il codice sorgente ai compagni, se qualcuno vuole imparare. Quindi è vietato portare a scuola software proprietario se non per studiare come funziona ai fini di poterlo riprodurre”.

Naturalmente la scuola deve praticare quanto predica: deve utilizzare in aula solo software libero (tranne file binari usati per studiarne il funzionamento interno) e condividere copie, incluso il codice sorgente, con gli studenti in modo che questi possano copiare, portare a casa e ridistribuire ulteriormente tale software.

Insegnare a chi studia l'uso del software libero, e a far parte della comunità del software libero, è una lezione di educazione civica sul campo. Ciò insegna inoltre il modello del servizio pubblico anziché quello dei potentati. Il software libero dovrebbe essere usato in scuole di ogni ordine e grado.

Chi ha una relazione con una scuola (studente, insegnante, impiegato, dirigente, donatore, genitore) deve fare pressione affinché la scuola passi al software libero. Se le richieste personali non funzionano, occorre dare pubblicità alla cosa nelle varie comunità, per riuscire in questo modo a sensibilizzare più persone e trovare alleati.

Perché le istituzioni educative devono usare ed insegnare il software libero

"Le scuole devono insegnare ai propri studenti ad essere cittadini di una società forte, capace, indipendente e libera."

Queste sono le ragioni principali per cui le università e le scuole di tutti i livelli devono utilizzare esclusivamente software libero.

Condivisione

Le scuole devono insegnare il valore della condivisione ponendosi come un esempio. Il software libero favorisce l'istruzione perché permette la condivisione della conoscenza e degli strumenti:

- **Conoscenza**. Molti giovani studenti hanno un talento per la programmazione; essi sono affascinati dai computer e ansiosi di imparare come funzionano. Con il software proprietario queste informazioni sono un segreto, così gli insegnanti non hanno la possibilità di renderle accessibili agli studenti. Con il software libero invece, l'insegnante può spiegare i concetti di base e poi fornire il codice sorgente in modo che possa essere letto e studiato dallo studente.
- **Strumenti**. Gli insegnanti possono dare ai loro studenti copie dei programmi che usano in classe da utilizzare anche a casa. Con il software libero, la copia non solo è autorizzata, è anche caldamente incoraggiata.

Responsabilità sociale

L'informatica è diventata parte essenziale della vita di ogni giorno. La tecnologia sta trasformando la società molto velocemente, e le scuole hanno un impatto decisivo sul futuro della società. La loro missione è di preparare gli studenti a partecipare ad una società digitale libera, insegnando loro le capacità di cui avranno bisogno per assumere il controllo delle loro vite senza difficoltà. Il software non deve essere sotto il potere di uno sviluppatore che compie unilateralmente decisioni che nessun altro può cambiare. Le istituzioni educative non devono permettere alle aziende di software proprietario di imporre il loro potere sul resto della società e sul suo futuro.

Indipendenza

Le scuole hanno la responsabilità etica di insegnare ad essere forti, non dipendenti da un singolo prodotto o da una potente azienda specifica. Inoltre, scegliendo di utilizzare software libero, la scuola stessa guadagna indipendenza da ogni interesse commerciale ed evita di legarsi mani e piedi ad un singolo venditore.

- Le aziende di software proprietario sfruttano le scuole e le università come un trampolino per raggiungere gli utenti in modo da imporre il loro software all'intera società. Offrono degli sconti o addirittura delle copie gratuite dei loro programmi proprietari alle istituzioni educative, così gli studenti imparano ad usarli e finiscono per dipenderne. Dopo la laurea, né loro né le aziende dove andranno a lavorare riceveranno alcuno sconto. In poche parole, ciò che queste aziende fanno è reclutare scuole e università per usarle come rappresentanti commerciali dei loro prodotti in modo da portare le persone verso una dipendenza

permanente per tutta la vita.

- Le licenze del software libero non scadono; questo vuol dire che una volta che il software libero è stato adottato, le istituzioni rimangono indipendenti dal venditore. In più, le licenze del software libero concedono agli utenti i diritti non solo di usare il software come desiderano, copiarlo e distribuirlo, ma anche di modificarlo per adattarlo alle proprie necessità. Pertanto, se le istituzioni eventualmente volessero implementare una particolare funzione in un programma, possono usufruire delle prestazioni di un qualsiasi programmatore per fare il lavoro, indipendentemente dal venditore originale.

Formazione

Gli studenti che si apprestano ad iscriversi ad un'università prendono sempre di più in considerazione quelle che includono il software libero nei corsi d'informatica e di sviluppo di software. Questo perché il software libero permette di studiare il funzionamento dei programmi e di modificarli all'occorrenza. Con il software libero s'impara la pratica professionale e l'etica nello sviluppo di software.

Risparmio

Questo è un vantaggio ovvio che attrarrà subito molti dirigenti scolastici, ma si tratta di un beneficio marginale. Il punto fondamentale di questo aspetto è che, essendo autorizzate a distribuire copie del programma ad un prezzo basso o gratuitamente, le scuole possono andare incontro alle famiglie in difficoltà economica, promuovendo di conseguenza l'equità e le pari opportunità di apprendimento tra gli studenti.

Qualità

Stabile, sicuro e facilmente installabile, il software libero offre una vasta varietà di soluzioni già disponibili per l'educazione. In ogni caso, l'eccellenza delle prestazioni è un beneficio secondario; lo scopo finale è la libertà degli utenti di computer.

Perché l'“Open Source” manca l'obiettivo del Software Libero

di **Richard Stallman**

Quando definiamo “libero” il software, intendiamo che rispetta le libertà essenziali degli utenti: la libertà di eseguire il programma, di studiare il programma e di ridistribuire delle copie con o senza modifiche. Questa è una questione di libertà, non di prezzo. Per capire il concetto, bisognerebbe pensare alla “libertà di parola” e non alla “birra gratis” [NdT: il termine free in inglese significa sia gratuito che libero, in italiano il problema non esiste].

Queste libertà sono d'importanza vitale. Sono delle libertà essenziali, non soltanto per quanto riguarda l'utente in sé, ma perché queste libertà promuovono la solidarietà sociale, cioè lo scambio e la cooperazione. Diventano sempre più importanti man mano che la nostra cultura e le attività delle nostre vite sono sempre più legate al mondo digitale. In un mondo di suoni, immagini e parole digitali, il software libero diventa sempre più una cosa simile alla libertà in generale.

Decine di milioni di persone in tutto il mondo usano oggi del software libero; nelle scuole di alcune regioni dell'India e della Spagna viene insegnato agli studenti l'uso del sistema operativo GNU/Linux, che è un sistema operativo libero. Tuttavia la maggior parte di questi utenti non sono mai venuti a conoscenza delle ragioni etiche per cui abbiamo sviluppato questo sistema e abbiamo creato la comunità del software libero, perché oggi questo sistema e questa comunità sono molto spesso descritte con il termine “open source”, e queste cose vengono riferite a una diversa filosofia, in cui, di solito, non si fa neppure riferimento a queste libertà.

Il movimento per il software libero sta facendo una campagna per le libertà degli utenti del computer dal 1983. Nel 1984 abbiamo fatto partire lo sviluppo del sistema operativo libero GNU, in modo da non dover utilizzare i sistemi operativi non liberi che negano la libertà ai propri utenti. Nel corso degli anni Ottanta abbiamo sviluppato la maggior parte dei componenti essenziali di questo sistema, così come abbiamo sviluppato la Licenza Pubblica Generica GNU (GNU GPL), una licenza studiata appositamente per proteggere la libertà di tutti gli utenti di un programma.

Tuttavia non tutti gli utenti e sviluppatori del software libero sono in accordo sugli obiettivi del movimento del software libero. Nel 1998 una parte della comunità del software libero si è staccata ed ha dato origine ad una campagna nel nome dell'“open source”. Questo era stato originariamente proposto per evitare la possibile confusione del termine inglese “free software”, ma ben presto questo termine è diventato associato a delle visioni filosofiche piuttosto diverse da quelle del movimento del software libero.

Alcune delle persone che promuovono l'“open source” lo considerano come una “campagna di marketing per il software libero”, che vorrebbe attrarre i dirigenti delle aziende mostrando i benefici pratici ed evitando di citare gli ideali di cosa sia giusto o sbagliato, cose il cui ascolto potrebbe non essere gradito. Altre persone che promuovono l'“open source” rigettano completamente i valori etici e sociali del movimento del software libero. Qualunque sia la loro visione, quando fanno una campagna per l'"open source"non fanno riferimento né prendono le difese di questi valori. Il termine “open source” è stato ben presto associato con la pratica di citare soltanto i valori pratici, come fornire software più potente e più affidabile. Molti dei sostenitori dell'“open source” sono giunti a questo da allora e questa pratica è quello che ha fatto sì che sia diventato questo il

significato di questo termine.

I due termini descrivono all'incirca la stessa categoria di software. Ma si basano su valori fondamentalmente diversi. L'open source è una metodologia di sviluppo; il software libero è un movimento sociale. Per il movimento per il software libero, il software libero è un imperativo etico, il rispetto essenziale della libertà degli utenti. Al contrario la filosofia dell'open source pensa a come "migliorare" il software soltanto da un punto di vista pratico. Dice che il software non libero è una soluzione non ottimale. Spesso le discussioni sull'“open source” non considerano quel che è giusto o sbagliato, ma solo il successo e la popolarità; ecco un esempio tipico (in inglese).

Per il movimento per il software libero, tuttavia, il software non libero è un problema sociale e la soluzione è passare al software libero.

Software libero. Open source. Se si tratta dello stesso software (o quasi), ha importanza quale nome venga utilizzato? Sì, perché parole differenti portano con sé idee diverse. Benché un programma libero, in qualunque modo venga chiamato, vi dia oggi la stessa libertà, stabilire la libertà in modo che perduri nel tempo dipende soprattutto dall'insegnare alla gente il valore della libertà. Se volete aiutarci in questo è essenziale che parliate di “software libero”.

Noi del movimento del software libero non pensiamo che il campo dell'open source sia il nemico; il nemico è il software proprietario (non libero). Vogliamo però che la gente sappia che noi ci battiamo per la libertà e che pertanto non vogliamo essere scorrettamente identificati come sostenitori dell'open source.[1]

Differenze pratiche tra software libero e open source

In pratica, open source indica criteri leggermente più deboli di quelli previsti per il software libero. Per quanto ne sappiamo, tutto il software libero esistente è anche open source. E anche quasi tutto il software open source è anche software libero, ma ci sono eccezioni. Innanzitutto, alcune licenze open source sono troppo restrittive (ad esempio: “Open Watcom” non è libero perché la sua licenza non permette di realizzare una versione modificata e usarla in privato) e non si possono considerare libere, ma fortunatamente tali licenze sono poco usate.

E poi, e questo è più importante in pratica, molti prodotti che contengono computer controllano l'integrità dei loro programmi eseguibili per impedire all'utente di installare eseguibili diversi; solo una specifica azienda può produrre eseguibili che funzionino sul dispositivo e che ne possano sfruttare tutte le capacità. Chiamiamo questi dispositivi "tiranni" e questa pratica "tivoization", dal nome del primo prodotto (Tivo) in cui l'abbiamo incontrata. Anche se questi eseguibili vengono da codice sorgente libero, gli utenti non possono eseguirne versioni modificate, quindi l'eseguibile è non libero.

I criteri per l'open source non riconoscono questo problema; guardano solamente la licenza del codice sorgente. Quindi questi eseguibili non modificati, quando prodotti a partire da codice sorgente come Linux che è open source e libero, sono open source ma non liberi. Molti prodotti basati su Android contengono eseguibili Linux non liberi e "tivoizzati".

1 Le evidenziazioni sono mie (Cesiano)

Errori comuni nell'attribuire il significato dei termini “software libero” e “open source”

Il termine “software libero” ha un problema di erronea interpretazione: un significato non intenzionale, il significato “il software che si può avere a costo zero” rientra nel significato del termine così quanto il significato che realmente si voleva dare a questo termine, “il software che dà agli utenti certe libertà”. Cerchiamo di puntualizzare questo problema con la pubblicazione della definizione di software libero e dicendo: pensate alla “libertà di parola” e non alla “birra gratis”. Questa non è una soluzione perfetta; non riesce a eliminare del tutto il problema. [NdT: il termine free in inglese significa sia gratuito che libero, in italiano il problema non esiste]. Un termine corretto e non ambiguo sarebbe meglio, se non avesse altri problemi.

Sfortunatamente tutte le alternative in inglese pongono alcuni problemi. Abbiamo preso in considerazione molti termini alternativi, che sono stati proposti, ma nessuno di questi è certamente la soluzione “giusta”, tale cioè che passare ad utilizzare quel termine sarebbe una buona idea. Ad esempio, usare la parola francese e spagnola "libre" può funzionare in certi contesti, ma non in altri, ad esempio in India. In inglese, ogni proposta per sostituire il termine “software libero” ha un certo tipo di problema a livello semantico. Ed anche il termine “software open source” rientra in questa categoria.

La definizione ufficiale di “software open source” (che è pubblicata dalla Open Source Initiative e che è troppo lunga per essere citata qui) venne creata derivandola in modo indiretto dai nostri criteri per il software libero. Tuttavia non è la stessa cosa: è più permissiva in alcuni aspetti. Ma nella pratica è una definizione abbastanza vicina alla nostra.

Tuttavia, il significato ovvio della espressione “software open source” è “Puoi guardare il codice sorgente” e la maggior parte delle persone sembra attribuire questo significato al termine. Questo è un criterio molto più debole che non quello di software libero e molto più debole della definizione ufficiale di open source. In questo criterio rientrano molti programmi che non sono liberi né open source.

Dal momento che il significato ovvio del termine “open source” non è quello che intendono i loro sostenitori, si è avuto come risultato che molte persone intendono in modo scorretto il significato del termine. Secondo lo scrittore Neal Stephenson, “Linux è “open source”, il che semplicemente significa che ognuno può avere una copia del codice sorgente”. Non penso che abbia deliberatamente cercato di rigettare o di porre in dubbio la definizione ufficiale. Penso che abbia semplicemente utilizzato il significato che, in inglese, viene convenzionalmente associato a questo termine. Lo stato del Kansas ha pubblicato una definizione simile: “Fate uso del software open-source software (OSS). OSS è quel software per il quale il codice sorgente è disponibile liberamente e pubblicamente anche se i termini delle specifiche licenze variano così come varia quello che è permesso fare con quel codice”.

Il *New York Times* ha pubblicato un articolo che estende il significato del termine fino a riferirlo al beta testing da parte degli utenti (lasciare che alcuni utenti provino una versione preliminare di un programma in forma confidenziale), pratica che gli sviluppatori di software proprietario adottano da decenni.

Le persone che sostengono l'open source cercano di affrontare questo problema indicando la loro

definizione ufficiale, ma questo approccio correttivo è molto meno efficace per loro di quanto lo sia, nel nostro caso, per noi. Il termine “software libero” ha due significati naturali, uno dei quali è il significato che intendiamo esprimere, pertanto una persona che abbia compreso la differenza tra libero e gratuito, come la differenza tra la “libertà di parola e non la birra gratis”, non prenderà in seguito il significato sbagliato. Invece il termine “open source” ha soltanto un significato naturale e questo significato è diverso da quello che intendono dire le persone che sono a favore dell'“open source”. Pertanto non vi è un modo veloce di spiegare e di giustificare la definizione ufficiale di “open source”. Questo crea la peggiore delle confusioni.

Un altro fraintendimento di “open source” è interpretarlo come “non distribuito con licenza GNU GPL”. Questo nasce dall'equivoco che “software libero” significhi “software coperto da GPL”. Entrambe queste idee sono errate: la GNU GPL è anche una licenza open source e, viceversa, la maggior parte delle licenze open source sono anche licenze di software libero. Ci sono molte licenze libere oltre alla GNU GPL.

Il termine “open source” è inoltre stato esteso per via della sua applicazione ad ambiti che non prevedono codice sorgente, come governo, istruzione e scienza; in questi campi i criteri delle licenze software sono fuori luogo, l'unica cosa che hanno in comune queste attività è l'invitare le persone in qualche modo a partecipare. Quindi a quel punto il termine assume il semplice significato di “partecipativo” o “trasparente”, o persino meno descrittivo. Nel caso peggiore è solo un termine di moda, senza significato preciso.

Differenti valori possono portare a conclusioni simili... ma non sempre

I gruppi radicali degli anni Sessanta hanno avuto la reputazione di essere faziosi ed interessati solo a se stessi: alcune organizzazioni si divisero perché vi era un disaccordo nei dettagli della strategia e i due nuovi gruppi che si formarono a seguito della scissione si trattavano l'un con l'altro come nemici, piuttosto che come organizzazioni con obiettivi e valori di fondo pressoché simili. La destra fece leva su questo aspetto per criticare l'intera sinistra.

Alcuni tentano di screditare il movimento per il software libero facendo un paragone tra il disaccordo tra noi e l'open source con il disaccordo di quei gruppi radicali. Hanno preso tutto alla rovescia. Noi non siamo in accordo con il gruppo dell'open source negli obiettivi e nei valori di base, ma le nostre visioni ci portano, in molti casi, verso lo stesso comportamento pratico, come sviluppare software libero.

Come risultato le persone del movimento del software libero e le persone della campagna per l'open source spesso lavorano assieme a progetti pratici come sviluppatori di software. È notevole come visioni tanto diverse dal punto di vista filosofico possano, così spesso, motivare persone che la pensano in modo differente a partecipare ad un progetto in comune. Ciò nonostante le visioni sono molto differenti e ci sono situazioni che ci portano ad compiere azioni molto differenti.

L'idea dell'open source è quella che permettere agli utenti di apportare modifiche al software e di ridistribuirlo renderà il software più potente e più affidabile. Ma questo non è garantito. Gli sviluppatori del software proprietario non sono necessariamente degli incompetenti e a volte producono dei programmi che sono potenti ed affidabili, anche se non rispettano le libertà degli

utenti. Come reagiranno a questo gli attivisti del software libero e gli entusiasti dell'open source?

Una persona che sia solamente un entusiasta dell'open source, e dunque una persona che non sia affatto influenzata dagli ideali del software libero, dirà: “Sono sorpreso che voi siate stati in grado di fare un programma che funzioni così bene senza utilizzare il nostro modello di sviluppo, ma ce l'avete fatta. Come posso avere una copia del vostro programma?” Questo favorirà atteggiamenti che allontanano dalle nostre libertà, che rischieranno di andare perse.

Una persona che sia un attivista del software libero dirà invece: “Il vostro programma è molto attraente, ma non può valere il prezzo della mia libertà, per cui devo farne a meno. Troverò un altro modo di fare quel che devo fare e darò il mio supporto ad un progetto che sviluppi un'alternativa libera.” Se noi consideriamo un valore la nostra libertà, allora agiremo in modo da mantenerla e da difenderla.

Anche il software potente ed affidabile non sempre è un bene

L'idea che vogliamo che il software sia potente ed affidabile nasce dall'ipotesi che il software è fatto per essere utile a chi lo usa. Se è potente ed affidabile sarà di maggiore utilità per gli utenti.

Ma si può dire che un software è utile agli utenti solo se rispetta le loro libertà. Cosa accade se il software è progettato in modo da mettere i suoi utenti in catene? In questo caso l'affidabilità vuol soltanto dire che le catene saranno più difficili da rimuovere. Funzionalità discutibili, come spiare gli utenti, restringerne la libertà, inserire back doors, imporre aggiornamenti sono comuni nel software proprietario, e qualche sostenitore dell'open source le vuole inserire nei programmi open source.

Sotto la pressione delle società di produzione cinematografica e discografica, il software che dovrebbe essere a disposizione degli individui è sempre più spesso specificatamente progettato per porli invece sotto restrizione. Questa minacciosa caratteristica è nota come DRM, cioè Digital Restrictions Management (vedi DefectiveByDesign.org) ed è l'antitesi dello spirito delle libertà che il software libero cerca di ottenere. E non è solo una questione ideologica: dal momento che lo scopo del DRM è quello di bloccare la vostra libertà, gli sviluppatori del DRM fanno sì che per voi sia difficile, impossibile e perfino illegale cambiare il software che ha il DRM.

Tuttavia alcun sostenitori dell'open source hanno fatto una proposta per un software “DRM open source”. La loro idea è che, con la pubblicazione del codice sorgente progettato per impedirvi l'accesso a supporti criptati e per dare la possibilità ad altri di cambiarli, si produrrà un software più potente e più affidabile nel porre le restrizioni a quelli che come voi siete gli utenti. E questo software vi verrà inviato in dispositivi che non vi è permesso cambiare.

Questo software potrebbe anche essere “open source”, ed utilizzare il modello di sviluppo dell'open source, ma non sarà software libero, perché non rispetterà le libertà degli utenti dove quel programma verrà eseguito. Se il modello di sviluppo dell'open source riuscirà a fare un software più potente e più affidabile nell'imporvi delle restrizioni, allora quello che si sarà ottenuto sarà di aver reso le cose ancora peggiori.

Paura della Libertà

Il più importante motivo che ha portato al termine “software open source” è che le idee etiche del

“software libero” non sono bene accettate da parte di alcune persone. Questo è vero: parlare di libertà, di questioni etiche, della responsabilità e della convenienza significa chiedere alle persone di pensare a delle cose che preferirebbero ignorare, come chiedersi se si stanno comportando in maniera eticamente corretta. Questo può causare un senso di imbarazzo e alcune persone possono decidere di risolvere la questione decidendo di non pensare a queste cose. Questo però non implica che dobbiamo smettere di parlare di queste cose.

Tuttavia questo è quello che i capi dell'“open source” hanno deciso di fare. Hanno ritenuto che evitando di parlare di questioni etiche e di libertà e parlando soltanto dei benefici pratici immediati di alcuni programmi liberi, sarebbero stati in grado di “vendere” più efficacemente il software a certi clienti, in particolar modo in ambito aziendale.

Questo approccio si è mostrato, a modo suo, efficace. La retorica dell'open source ha convinto molte aziende e persone private ad utilizzare, e persino sviluppare, software libero, cosa che ha fatto estendere la nostra comunità, ma soltanto alla superficie, ad un livello pratico. La filosofia dell'open source, con i suoi valori esclusivamente pratici, impedisce la comprensione delle idee più profonde del software libero; porta molte persone nella nostra comunità, ma non insegna loro a difenderla. Ciò è un bene finché le cose vanno bene, ma non è abbastanza per mettere al sicuro la libertà. Attirare gli utenti verso il software libero li porta soltanto verso il primo passo per diventare difensori delle loro libertà.

Prima o poi questi utenti saranno invitati a tornare indietro ad utilizzare software proprietari per qualche ragione di vantaggio pratico. Molte aziende cercano di offrire una tentazione di tal genere, alcune offrendo perfino delle copie gratis. Perché gli utenti dovrebbero rifiutare? Lo faranno soltanto se avranno imparato a dare un valore alla libertà che il software libero dà a loro, a dare un valore alla libertà in quanto tale piuttosto che alla convenienza tecnica e pratica di un particolare software libero. Per diffondere questa idea dobbiamo parlare di libertà. La tecnica dell'“evita di parlarne” può essere utile nei confronti delle aziende, ma fino ad un certo punto: se ne si abusa, al punto da diventare una cosa comune, si corre il rischio che l'amore per le libertà diventi quasi qualcosa di eccentrico.

Questa situazione pericolosa, ora descritta, è proprio ciò che abbiamo noi ora. Molte persone coinvolte nel software libero dicono ben poco sulla libertà (di solito perché cercano di essere “il più possibile accettabili dalle aziende”. In particolar modo i distributori di software mostrano questo atteggiamento. Quasi tutti i distributori di sistemi operativi GNU/Linux aggiungono pacchetti proprietari al sistema libero di base e invitano gli utenti a considerarli come un vantaggio, piuttosto che un passo indietro che li allontana dalla libertà.

Le aggiunte (add-on) software proprietarie e le distribuzioni GNU/Linux parzialmente non libere trovano un terreno fertile perché la maggior parte della nostra comunità non insiste sulla libertà del proprio software. Questa non è una coincidenza. La maggior parte degli utenti dei sistemi GNU/Linux hanno fatto la conoscenza di questi sistemi dalle discussioni sull'“open source”, nelle quali non veniva detto che l'obiettivo era la libertà. La pratica di non proteggere a tutti i costi la libertà e la decisione di non parlare della libertà vanno di pari passo e ognuna promuove l'altra. Per superare questa tendenza, dobbiamo parlare di più, e non di meno, di libertà.

“FLOSS” e “FOSS”

I termini “FLOSS” e “FOSS” sono usati quando si desidera essere neutrali tra software libero e open source. La scelta migliore per chi desidera essere neutrale è “FLOSS” che è davvero neutrale, ma chi vuole difendere la libertà dovrebbe evitare di utilizzare termini neutrali. Per difendere la libertà bisogna mostrare agli altri che si privilegia la libertà.

Rivalità nell'attenzione del pubblico

I termini "libero" e "open" competono per l'attenzione del pubblico: "software libero" e "software open source" sono idee diverse ma competono per il medesimo spazio per molti degli osservatori. Se una persona si abitua ad usare il termine "open source", questo la ostacolerà a capire la filosofia del movimento del software libero. Chi già è abituato ad associare noi e il nostro software con la parola "open" non capirà che noi siamo qualcosa di *diverso* se non con un enorme sforzo intellettuale. Tutte le attività che promuovono la parola "open" infittiscono la nebbia che nasconde gli ideali del movimento per il software libero.

Quindi gli attivisti del software libero devono riflettere prima di lavorare su un'attività che si definisce "open", perché anche se l'attività in sé è buona, ogni contributo che si fa ad essa ha dei piccoli effetti collaterali negativi. Meglio contribuire a una delle tante altre attività che si definiscono "libere", perché in questo caso ogni contributo ha piccoli effetti collaterali in senso positivo. Con tanti progetti utili a disposizione, perché non sceglierne uno che che ha effetti collaterali positivi?

Conclusioni

Dato che i sostenitori dell'open source portano nuovi utenti nella nostra comunità, noi attivisti del software libero dobbiamo lavorare ancor di più per portare all'attenzione dei nuovi utenti l'argomento della libertà. Dobbiamo dire: “Questo è software libero e ti dona la libertà!”, più spesso e con più forza che mai. Tutte le volte che direte “software libero” al posto di “open source”, aiuterete la nostra causa.

Note

Un articolo di Lakhani e Wolf sulle motivazioni degli sviluppatori di software libero dice che una parte considerevole sono motivati dall'idea che il software debba essere libero. Ciò nonostante il fatto che la ricerca si sia svolta su SourceForge, un sito che non dà supporto alla visione che questo sia un argomento etico.

-

Linux e il sistema GNU

di Richard Stallman

> Per ulteriori informazioni vedere anche le domande frequenti su GNU/Linux e Perché GNU/Linux?

Molti usano ogni giorno nei propri computer una versione modificata del sistema GNU senza rendersi conto. A causa di una serie di eventi particolari, la versione del sistema GNU attualmente più diffusa è spesso nota come "Linux", e sono tanti gli utenti che non sanno che si tratta fondamentalmente del sistema GNU, sviluppato dal Progetto GNU.

Esiste davvero un Linux, e queste persone lo usano, ma è solo una parte del sistema. Linux è il kernel, il programma del sistema che si occupa di assegnare le risorse della macchina agli altri programmi che vengono eseguiti. Il kernel è una parte essenziale del sistema operativo, ma non si può utilizzare da solo; un kernel è utile soltanto in quanto parte di un sistema operativo completo. Linux è normalmente utilizzato in combinazione con il sistema operativo GNU: il sistema è fondamentalmente GNU con l'aggiunta di Linux, vale a dire GNU/Linux. Tutte le distribuzioni cosiddette "Linux" sono in realtà distribuzioni GNU/Linux.

Molti utenti non capiscono la differenza tra il kernel, che è Linux, e il sistema completo, che loro chiamano sempre "Linux". L'uso ambiguo di questo nome non facilita la comprensione. Questi utenti spesso credono che Linus Torvalds, con un po' di aiuto, abbia sviluppato tutto il sistema operativo nel 1991.

I programmatori di solito sanno che Linux è un kernel; ma dato che anche loro sovente hanno sentito parlare di "Linux" riferito al sistema completo, il più delle volte immaginano una storia che possa giustificare il fatto di dare a tutto il sistema il nome del kernel. Per esempio, molti credono che quando Linus Torvalds ebbe finito di scrivere il kernel, gli utenti si guardarono intorno alla ricerca di altro software libero e trovarono che, come per caso, praticamente tutto il necessario per costruire un sistema simile a Unix era già disponibile.

Quello che trovarono non fu casuale: si trattava del sistema GNU già quasi completo. Il software libero disponibile si unì per formare un sistema completo perché il Progetto GNU stava già lavorando dal 1984 per creare un tale sistema. Nel Manifesto GNU avevamo stabilito l'obiettivo di sviluppare un sistema libero simile a Unix chiamato GNU. L'Annuncio iniziale del Progetto GNU delinea inoltre alcuni dei piani originali per il sistema GNU. Quando si cominciò a scrivere Linux, GNU era già quasi finito.

La maggior parte dei progetti di software libero ha lo scopo di sviluppare un programma che svolga un particolare compito. Per esempio, Linus Torvalds si propose di scrivere un kernel compatibile con Unix (Linux); Donald Knuth si propose di scrivere un impaginatore di testi (TeX); Bob Scheifler di sviluppare un sistema a finestre (X Windows). E' normale misurare il contributo di questo tipo di progetti in base ai programmi specifici che hanno prodotto.

Se provassimo a quantificare il contributo del Progetto GNU in base a questo criterio, a quale conclusione arriveremmo? Un distributore di CD-ROM rilevò che nella sua "distribuzione Linux", il software GNU costituiva la componente maggiore, circa il 28% del totale del codice sorgente,

inclusi alcuni dei principali componenti essenziali senza i quali non potrebbe esserci alcun sistema. Linux propriamente detto costituiva circa il 3%. (Nel 2008 le proporzioni sono state simili: nel repository principale di gNewSense, Linux rappresenta l'1.5% e i pacchetti GNU il 15%). Quindi se si dovesse scegliere un nome per il sistema in base a chi scrisse i programmi che ne fanno parte, la scelta più appropriata sarebbe "GNU".

Ma non è questo questo il modo più appropriato di considerare la questione. Il Progetto GNU non era, e non è, un progetto per lo sviluppo di pacchetti software specifici. L'obbiettivo non era lo sviluppo di un compilatore C, anche se lo abbiamo fatto. Non era un progetto per scrivere un editor di testo, anche se ne abbiamo sviluppato uno. Lo scopo del progetto GNU era sviluppare un sistema libero completo simile a Unix: GNU.

Molte persone hanno dato contributi importanti al software libero presente nel sistema, e tutti meritano riconoscimento per il lavoro svolto. Ma il motivo per cui si tratta di *un sistema integrato* e non soltanto di una collezione di programmi utili, è che il Progetto GNU si era costituito allo scopo di costruire un sistema unitario. Abbiamo fatto un elenco dei programmi necessari per ottenere un sistema libero *completo*, e sistematicamente abbiamo trovato, scritto o ottenuto che altri scrivessero tutti i programmi presenti nel nostro elenco. Abbiamo scritto parti importanti, essenziali ma a volte anche tediose (1) perché senza questi strumenti non si può avere un sistema. Alcuni dei nostri componenti, gli strumenti per la programmazione, furono molto apprezzati dai programmatori, ma abbiamo anche scritto molti altri componenti che non sono strumenti (2). Abbiamo persino scritto un programma per giocare a scacchi, perché un sistema completo ha bisogno anche di giochi.

All'inizio degli anni '90 avevamo messo insieme l'intero sistema ad eccezione del kernel. Avevamo anche iniziato a lavorare su un kernel, GNU Hurd, che si appoggia su Mach. Lo sviluppo di questo kernel è stato molto più difficile di quanto ci aspettassimo. Il kernel GNU Hurd fu pronto e funzionante in modo affidabile nel 2001, ma manca ancora molto per renderlo usabile da parte del pubblico.

Per fortuna, non è stato necessario aspettare che Hurd fosse finito. Quando Torvalds rilasciò Linux come software libero nel 1992, il suo kernel colmò l'ultimo vuoto principale del sistema GNU. Fu quindi possibile mettere insieme Linux e il sistema GNU per ottenere un sistema libero completo: una versione del sistema GNU che conteneva anche Linux; in altre parole, il sistema GNU/Linux.

Far funzionare bene i due insieme non fu compito semplice. Alcuni componenti GNU (3) furono modificati considerevolmente per farli funzionare con Linux. L'integrazione di un sistema completo per ottenere una distribuzione "pronta all'uso" fu un lavoro altrettanto impegnativo. Abbiamo dovuto affrontare i problemi dell'installazione e dell'avvio del sistema, questioni che non erano ancora state affrontate. Le persone che svilupparono le varie distribuzioni del sistema diedero un contributo sostanziale. Ma si trattò di un compito che, per natura di cose, era comunque destinato ad essere portato a termine da qualcuno.

Il Progetto GNU supporta sia i sistemi GNU/Linux sia *il* sistema GNU. La Free Software Foundation (FSF) ha finanziato la riscrittura delle estensioni relative a Linux della libreria GNU C, in modo che ora sono ben integrate, e i sistemi GNU/Linux più recenti usano la versione attuale della libreria senza modifiche. La FSF ha anche finanziato una prima fase di sviluppo di Debian GNU/Linux.

Oggi esistono numerose varianti del sistema GNU/Linux, di solito vengono chiamate “distribuzioni” o, più brevemente, “distro”. La maggior parte include software non libero, in quanto gli sviluppatori di tali versioni seguono la filosofia associata a Linux piuttosto che la filosofia GNU. Ma esistono anche distribuzioni GNU/Linux completamente libere. La FSF dà sostegno a gNewSense.

Per ottenere una distribuzione libera non basta eliminare i vari programmi non liberi. Al giorno d'oggi, anche la versione più diffusa del kernel Linux contiene programmi non liberi. Tali programmi vengono caricati nei dispositivi I/O all'accensione del sistema; si tratta di una serie di numeri inclusi nel “codice sorgente” di Linux. Di conseguenza, gestire una distribuzione libera di GNU/Linux oggi comporta anche la manutenzione di una versione libera di Linux.

Sia che usiate GNU/Linux o meno, vi preghiamo di non confondere il pubblico utilizzando la forma “Linux” in modo ambiguo. Linux è il kernel, uno dei componenti esenziali del sistema. Il sistema nella sua totalità è sostanzialmente il sistema GNU con l'aggiunta di Linux. Quindi se vi riferite a questa combinazione, vi preghiamo di usare la dicitura “GNU/Linux”.

Chi desiderasse dei collegamenti di riferimento per il termine “GNU/Linux”, può utilizzare questa pagina oppure http://www.gnu.org/gnu/the-gnu-project.html. Per il kernel Linux invece si può usare questo indirizzo di riferimento: http://foldoc.org/linux.

Nota: Oltre a GNU, esiste un altro progetto che ha prodotto indipendentemente un sistema operativo libero simile a Unix. Questo sistema è conosciuto con il nome di BSD, ed è stato sviluppato all'Università di Berkeley. Negli anni '80 il sistema non era libero, ma fu liberato agli inizi degli anni '90. Qualsiasi sistema operativo libero oggi esistente (4) è quasi certamente una variante di un sistema GNU o un tipo di BSD.

A volte le persone si chiedono se BSD sia una versione di GNU, come GNU/Linux. Gli sviluppatori di BSD si ispirarono all'esempio fornito dal progetto GNU per liberare il loro codice, e furono incoraggiati dagli espliciti appelli da parte degli attivisti di GNU, ma in realtà il loro codice aveva poco in comune con GNU. Gli attuali sistemi BSD usano dei programmi GNU, così come il sistema GNU e le sue varianti usano alcuni programmi BSD; ma considerati complessivamente, sono due sistemi differenti che si sono evoluti separatamente. I programmatori di BSD non svilupparono un kernel che sia poi stato aggiunto al sistema GNU, pertanto una dicitura del tipo GNU/BSD sarebbe inappropriata. (5)

Note:

1. Questi componenti tediosi ma essenziali sono l'assemblatore GNU (GAS) e il linker (GLD), che ora fanno parte del pacchetto GNU Binutils , GNU tar, e molto altro.
2. Per esempio la Bourne Again SHell (BASH), l'interprete PostScript Ghostscript e la libreria GNU C non sono strumenti di programmazione. Così come non lo sono GNUCash, GNOME e GNU Chess.
3. Per esempio la libreria GNU C.
4. E' stato poi sviluppato un sistema quasi libero simile a Windows, ma tecnicamente non è assolutamente come GNU o Unix, quindi non riguarda la questione. La maggior parte del kernel di Solaris è stato rilasciato come software libero, ma se uno volesse farne un sistema

libero non solo ci sarebbe da sostituire le parti mancanti del kernel, ma bisognerebbe anche inserirlo nel sistema GNU o BSD.

5. D'altra parte, col passare del tempo da quando è stato scritto questo articolo, la libreria GNU C è stata adattata alle diverse versioni del kernel BSD, il che ha semplificato la combinazione del sistema GNU con questo kernel. Così come succede con GNU/Linux, si tratta infatti di varianti del sistema GNU, di conseguenza vengono chiamate con nomi tali come GNU/kFreeBSD e GNU/kNetBSD, a seconda del kernel utilizzato. Per gli utenti comuni è spesso difficile distinguere tra GNU/Linux e GNU/*BSD.

Cos'è il Progetto GNU?

Cos'è il Progetto GNU?