

Software libero dal produttore al consumatore

Gianni Bianchini

Università di Siena - Firenze Linux User Group

`giannibi@dii.unisi.it` - `giannibi@firenze.linux.it`

Linux Day 2003

Siena, Facoltà di Ingegneria, 29 Dicembre 2003

Copyright ©2003 Gianni Bianchini

La copia letterale integrale e la redistribuzione di questo documento sono consentite a condizione che questa nota sia riprodotta

Che cos'è il software?

Il software è un insieme di istruzioni che permettono l'esecuzione di un dato compito da parte di un elaboratore

- Il software è informazione
- Il software è un'opera dell'ingegno
- Il software è un prodotto culturale
- Nella società moderna, il software permette la produzione e la diffusione della conoscenza, ed è esso stesso conoscenza
- Il software può essere replicato e diffuso con facilità ed a costo zero
- Il software è tutelato dalle norme sul diritto d'autore

Che cos'è il software?

Alcune definizioni

- **Codice sorgente**: programma scritto in un linguaggio comprensibile all'uomo ma non direttamente alla macchina
- **Codice binario**: insieme di istruzioni interpretabili ed eseguibili dalla macchina ma difficilmente intelligibili dall'uomo
- **Compilazione**: passaggio dalla prima alla seconda forma
- **Licenza**: l'insieme delle condizioni stabilite dal detentore dei diritti d'autore sul software riguardo all'uso ed alla distribuzione dello stesso

N.B.: per la semplice esecuzione del programma è sufficiente la disponibilità del solo codice binario, ed è questa la forma di distribuzione di larga parte del software

Software libero: le motivazioni

- Libero scambio dell'informazione
- Condivisione di idee e risultati
- Uso del patrimonio comune di conoscenze per facilitare lo sviluppo

Si tratta degli stessi principi a cui da sempre si rifà la comunità scientifica. Senza di essi non vi è progresso nella ricerca.

Il software libero è essenzialmente software le cui modalità di sviluppo e d'uso si ispirano ai suddetti principi.

Software libero: le origini

- Prima dell'avvento dell'informatica "di massa" (anni '60 e '70), la diffusione del software tra le comunità di ricercatori e sviluppatori avveniva in modo del tutto libero
- Nei primi anni '80 si affermò l'idea del software *proprietario*
 - ★ Uso soggetto ad accordi di non-diffusione
 - ★ Copia non consentita
 - ★ Ambito ed entità dell'utilizzo ristretti
 - ★ Nella maggior parte dei casi, non disponibilità del codice sorgente, con conseguente impossibilità di conoscere il funzionamento
 - ★ Impossibilità di effettuare e/o distribuire modifiche

Ma qualcuno non ci stava...

Il progetto GNU

- Nacque nel 1984 per iniziativa di Richard M. Stallman, ex membro della comunità di *hacker* del laboratorio di intelligenza artificiale del MIT (Massachusetts Institute of Technology)
- GNU is Not Unix!
- Obiettivo: la creazione di un sistema operativo completo e *libero*
 - ★ Nucleo (o *kernel*)
 - ★ Programmi di utilità
 - ★ Librerie e compilatori
 - ★ Applicativi ed interfacce grafiche
- Nacque la *Free Software Foundation*, ombrello legale per il progetto
- Attualmente il progetto conta numerosi applicativi software e migliaia di sviluppatori attivi

Software libero: l'idea fondamentale

- L'idea centrale del software libero fu sintetizzata in 4 libertà che debbono essere concesse all'utente dall'autore tramite la licenza:
 - ★ poter eseguire il programma per qualunque scopo
 - ★ poter studiare il funzionamento del programma ed adattarlo alle proprie esigenze
 - ★ poter distribuire copie del programma senza limitazioni
 - ★ poter creare modifiche e lavori derivati e distribuirli pubblicamente
- Affinché il software sia libero, è necessario che il codice sorgente sia disponibile
- “Libero” \neq “di dominio pubblico”. Possono essere imposte restrizioni nella licenza affinché le libertà concesse siano preservate
 - ★ Copyleft: all rights *reversed!* Licenza GPL
 - ★ Software libero senza copyleft. Licenza BSD

Il fenomeno GNU/Linux

- Fino al 1991, i programmi sviluppati nell'ambito del progetto GNU rimasero dipendenti da kernel Unix proprietari
- *Linux* nacque per opera di Linus Torvalds, allora studente all'università di Helsinki. Per il suo kernel *Unix-like* scritto per PC 386, Linus adottò la licenza libera GPL del progetto GNU
 - ★ Ciò che viene chiamato generalmente “il sistema Linux”, rappresenta l'integrazione di GNU con Linux
- La diffusione di massa della rete Internet e la licenza libera causarono un'esplosione della notorietà e guadagnarono a Linux un gran numero di sviluppatori e l'attenzione di realtà commerciali
- La fortuna nelle applicazioni server e di rete fu immediata grazie anche alla tradizione Unix
- Lo sviluppo di interfacce grafiche sofisticate (KDE/Gnome) e aprì la porta all'uso “Desktop”

Sfatiamo alcuni miti

- Il software libero è gratuito
 - ★ È falso: la libertà del software non ha nulla a che vedere con il prezzo. Benché gran parte del software libero più diffuso sia distribuito gratuitamente, ci sono programmatori che vivono della vendita e della manutenzione dei programmi liberi da loro creati, esiste il software libero commerciale.
- Il software gratuito è libero
 - ★ È falso: molti programmi proprietari vengono distribuiti gratuitamente (freeware, shareware)
- Il software libero è privo di copyright
 - ★ Al contrario, il copyright ed il diritto d'autore sono gli istituti in base ai quali è possibile definire una licenza d'uso, anche se libera (copyleft o meno)
- Software libero vs. Open Source

Dal produttore al consumatore

- Il software è un bene astratto, ma pur sempre un bene

Nel software **proprietario**...

- Ruoli molto ben definiti
- Modalità di fruizione dell'opera ristrette e limitate
- Assistenza e formazione vincolate al produttore/fornitore
- Dati “prigionieri” dell'applicazione (problema dei formati)
- Modello di sviluppo tipico: *a cattedrale*
 - ★ Gruppo di sviluppo ristretto
 - ★ Nel caso di tecnologie chiuse, spesso vincolato al segreto

Dal produttore al consumatore

Nel software **proprietario**...

- Limitate possibilità di "movimento" dell'utente
 - ★ Sorgente **chiuso**
 - * *Non funziona*
 - * Richiesta nuove caratteristiche al solo produttore
 - * Scoperta e segnalazione di *bug* su sola base esperimento
 - * In casi estremi (ed a volte non legali) *reverse engineering*
 - ★ Sorgente **aperto**
 - * Maggiore flessibilità data dalla possibilità di esame ed *auditing* del codice (fondamentale in applicazioni come l'autenticazione crittografica e la firma digitale)
 - * Eventuali contributi dell'utente sono soggetti all'approvazione (e spesso all'appropriazione) da parte del produttore
- Creazione di alibi distorti nel caso di software a sorgente chiuso
 - ★ *Security through obscurity* come alternativa ad auditing e correzione delle vulnerabilità (*brrr...*)

Dal produttore al consumatore

Nel software libero...

- I ruoli non sono affatto ben definiti, anzi...
- Modelli di sviluppo: a *cattedrale* ma soprattutto a *bazaar*
 - ★ Possibile evoluzione dall'uno all'altro modello
 - ★ Sviluppo collaborativo (traduzioni, documentazione, sistemi di gestione delle versioni e di *bug tracking*, *mailing list* e gruppi di discussione)
- Potenzialità del *feedback* sfruttate al massimo
 - ★ Esame, modifica, distribuzione (quasi) incondizionati del codice
 - ★ *Quick fixes e patches*
- Un progetto può nascere dalle più svariate esigenze ed avere evoluzioni diverse ed anche imprevedibili...
- ...o morire di morte naturale (*non violenta*), ma il lavoro fatto non va perduto

I vantaggi sociali

- Patrimonio pubblico
 - ★ Il software libero si configura come bene pubblico a disposizione di tutti
- Accesso alla tecnologia
 - ★ Il software libero aiuta a superare il divario digitale (*digital divide*) tra i paesi più ricchi e quelli più poveri, mettendo tutti sulle stesse basi di partenza
- Valore formativo
 - ★ La possibilità di studiare e modificare i sorgenti permette a tutti di imparare ed operare con software allo stato dell'arte. Lo sviluppo collaborativo permette una partecipazione diretta.
- Condivisione della conoscenza
 - ★ Il carattere pubblico dello sviluppo e la condivisione dei risultati permettono una diffusione del patrimonio delle conoscenze che non restano appannaggio di industrie o centri di ricerca.

L'hacker

Potremmo definire così colui che si identifica
sia nel ruolo di produttore sia in quello di consumatore (*attivo!*)
secondo il modello precedente

Modo produttore: un nuovo progetto di software libero!

Come nasce?

- Tradizionalmente per *passione*
 - ★ Imparare (sfruttando l'esperienza di altri!)
 - ★ Assai importante la *ricerca* e l'individuazione di materiale da cui attingere
- Per necessità...
 - ★ di fare qualcosa di nuovo (con molti *tool* di uso generale a disposizione)
 - ★ di migliorare qualcosa di già esistente
- L'idea commerciale
 - ★ Modelli economici basati sul software libero
- Rilascio con licenza libera di software già proprietario
 - ★ Il caso *OpenOffice.org*

Gli strumenti liberi

- I sistemi operativi
 - ★ Per lo più *Unix-like* - GNU/Linux, (Free|Open|Net)BSD
- I linguaggi di programmazione ed i compilatori
 - ★ Linguaggi compilati: C, C++, Objective-C, Java, Fortran, Ada ...
 - ★ Interpretati: Perl, PHP, Python, Tcl/Tk, Unix shell ...
 - ★ Il compilatore per eccellenza: GCC (*The GNU Compiler Collection*)
 - * C, C++, Objective-C, Fortran, Ada, Java
 - * *Multi- e cross-piattaforma*
- Le librerie di sviluppo
 - ★ Insieme pressoché infinito per le più varie funzionalità
 - ★ Librerie di interfaccia grafica (e non solo)
 - * Gtk/Gnome
 - * Qt/KDE
 - ★ Interfacce di programmazione ad applicativi (API)

Gli strumenti liberi

- Gli ambienti di sviluppo integrati (IDE)
- I sistemi di sviluppo concorrente, di manutenzione e gestione dei contenuti
- Gli strumenti di localizzazione ed internazionalizzazione
- I sistemi ed i formati liberi di documentazione
- I sistemi di bug-tracking
- I servizi per l'indicizzazione e l'hosting di progetti in rete
 - ★ Sourceforge (hosting)
 - ★ Freshmeat (indicizzazione)
 - ★ I gruppi di utenti (LUG, ecc.)

Al lavoro!

- Codifica
- Documentazione
- Traduzioni
- Distribuzione
- Recepimento del feedback degli utenti, bug tracking
- Manutenzione
 - `while (1) {`
 - ★ Correzione dei bug ed aggiunta funzionalità
 - ★ Testing e distribuzione nuova versione
 - ★ Nuove traduzioni
 - ★ Feedback (suggerimenti, patches)
 - ★ Possibilità di *fork* del progetto
 - `}`

Modo consumatore: per l'utente

- I sistemi operativi liberi (per lo più **nix flavours*)
 - ★ GNU/Linux. Disponibile in numerosi gusti (o *distribuzioni*)
 - * Debian
 - * Slackware
 - * Red Hat / Fedora
 - * Mandrake
 - * ...
 - ★ *BSD
 - * FreeBSD
 - * OpenBSD
 - * NetBSD
- Gli applicativi liberi
 - ★ Le distribuzioni escono con grandi quantità di software (libero o anche non libero) precompilato e *pacchettizzato*
 - ★ Sistemi di gestione/installazione/configurazione pacchetti

Per l'utente

- Gli aggiornamenti
 - ★ Le maggiori distribuzioni mettono a disposizione i pacchetti corretti in corrispondenza di bug e/o vulnerabilità
 - * Sistemi di aggiornamento automatico integrati nella gestione pacchetti
 - * Da citare: *security.debian.org*. Tempo medio (dichiarato) di pubblicazione del fix dalla segnalazione della vulnerabilità: circa 48 ore
 - ★ Specificità degli aggiornamenti
 - * A livello di singole applicazioni/moduli/librerie compresi nella distribuzione
 - * In diffusi sistemi proprietari, gli aggiornamenti vengono spesso raggruppati in non meglio definiti “pacchi di manutenzione” il cui scopo non è sempre chiaramente documentato

Il codice sorgente per l'utente

- Esame del codice
 - ★ È fondamentale in contesti critici (es. applicazioni crittografiche per autenticazione, privacy e integrità dei dati)
- Curiosità!
- Personalizzazione
- Incorporazione di procedure di terze parti
- Mancanza o incompatibilità dei binari rilasciati
- Ottimizzazione per l'hardware e il software ospite
- Esigenze di sicurezza per cui si rendono troppo lunghi i tempi fisiologici di pubblicazione del binario corretto
 - ★ Nelle mailing list o gruppi di discussione vengono spesso pubblicati quick fixes e patches contestualmente all'annuncio della vulnerabilità

Alcune insidie

- La brevettabilità degli algoritmi (e delle idee in genere)
 - ★ Possibile in USA e Giappone (in Europa ha subito un temporaneo stop)
 - ★ Inadeguata al ciclo di vita del software
 - ★ Gli sviluppatori di software libero non hanno in genere le risorse per pagare le royalties, e le condizioni di sfruttamento possono essere restrittive
 - ★ Alcune implementazioni libere di algoritmi sono state sviluppate in Europa proprio perché non bloccate da brevetto
- Le nuove normative sul diritto d'autore (DMCA in USA, EUCD nella UE)
 - ★ Tra le altre cose, viene reso illegale l'aggiramento di misure software a protezione di materiale coperto da diritto d'autore, indipendentemente dall'effettiva violazione di tale diritto
 - ★ Impossibilità di creare software interoperante per *accedere legittimamente* a formati proprietari

Enjoy free software!!!

/giannibi

Riferimenti

- Associazione Software Libero - AsSoLi
<http://www.softwarelibero.it>
- Free Software Foundation Europe
<http://www.fsfeurope.org>
- Free Software Foundation (USA)
<http://www.fsf.org>
- La storia del progetto GNU
<http://www.it.gnu.org/gnu/thegnuproject.it.htm>
- Steven Levy, *Hackers, heroes of the computer revolution*,
<http://mosaic.echonyc.com/~steven/hackers.html>
- E. Raymond, *The Cathedral and the Bazaar*,
<http://www.tuxedo.org/~esr/writings/cathedral-bazaar>

Riferimenti

- R. Chassel, *Un'economia del software libero: vantaggi e pericoli*,
<http://www.softwarelibero.it/altri/economia-sl.shtml>
- A. Rubini, *Materiale sul software libero*,
<http://www.linux.it/GNU>
- Il kernel Linux
<http://www.kernel.org>
- The history of Linux
<http://ragib.hypermart.net/linux>
- Sourceforge
<http://www.sourceforge.net>
- Freshmeat
<http://freshmeat.net>

Riferimenti

- Debian GNU/Linux
<http://www.debian.org>
- OpenOffice.org
<http://www.openoffice.org>
- L'Associazione Software Libero sulla EUCD
<http://www.softwarelibero.it/progetti/eucd>

Ack

- Simone Piccardi
- Simo Sorce
- Alessandro Rubini
<http://ar.linux.it>
- Szymon Stefanek a.k.a. Pragma
<http://www.kvirc.net>
- Firenze Linux User Group - FLUG
<http://www.firenze.linux.it>

Typeset using L^AT_EX