1. Che cos'è l'Open Source

1.1 La storia dell'open source e la sua filosofia

"I consider that the golden rule requires that if I like a program I must share it with other people who like it. I cannot in good conscience sign a non disclosure agreement or a software license agreement."

"Ritengo che la Regola Aurea richieda questo: se mi piace un programma lo devo condividere con le altre persone a cui piace. Non posso – rimanendo con la coscienza pulita – firmare un accordo di non divulgazione o una licenza software."[1]

Nel 1983 Richard Stallman – un ricercatore del MIT - con un messaggio di posta elettronica sui newsgroup net.unix-wizards e net.usoft annunciava il suo intento di realizzare un sistema GNU[2] Unix-compatibile: le fondamenta del concetto di software a codice aperto a tutti erano

1L'annuncio originale di Richard Stallman è reperibile al link: http://groups.google.com/groups?q=GNU+RMS&hl=en&lr=&ie=UTF-8&oe=UTF-8&as_drrb=b&as_mind=1&as_minm=9&as_miny=1983&as_maxd=31&as_maxm=10&as_maxy=1983&selm=771%40mit-eddie.UUCP&rnum=1 (Trad. mia).

2GNU: acronimo che sta per "GNU is Not Unix." Un gioco di parole per descrivere allo stesso tempo il forte legame di GNU con UNIX ma anche per puntualizzarne le diversità.

state gettate. Fino a quel momento il sistema operativo più utilizzato nei laboratori di ricerca delle università era Unix[3]. Dagli anni sessanta fino alla fine degli anni settanta si è assistito ad una vera e propria rivoluzione nei sistemi informatici: è stato scritto il linguaggio C, sono state poste le basi della attuale rete Internet e sono arrivati sul mercato i primi minicomputer (poi soppiantati dalle workstation a bassi costi hardware) e – soprattutto – si è assistito alla nascita di una nuova comunità di persone: gli *hackers*. Come si legge nel Jargon File[4]: "*hacker* [...] *a person who enjoys exploring the details of progammable systems and how to stretch their capabilities, as opposed to most users, who prefer to learn only the minimum necessary*" ovvero "una persona che si diletta di esporare i dettagli di un sistema programmabile e come spingere ai limiti le sue capacità, figura opposta a quella del più degli utenti, che preferiscono imparare solo il minimo necessario". Ed è proprio grazie a questo manipolo di pionieri che si deve la nascita di molta della moderna tecnologia (stampanti laser, interfacce visuali a icone, tecnologia delle reti). Molti di questi ricercatori universitari già da tempo sviluppavano software per i calcolatori presenti nel MIT e nelle altre università ad esso collegate tramite ARPANet[5].

3La differenza tra la dicitura "Unix" e "UNIX" è costituita dal fatto che mentre il secondo è un marchio registrato che indica la realizzazione di Unix creata da USL (Unix System Laboratories) della Novell, con il primo ci si riferisce al complesso di tutti i sistemi operativi che si rifanno allo UNIX originale, anche se non sono stati costruiti a partire dagli stessi codici sorgenti.

4http://www.catb.org/~esr/jargon/ "This is the Jargon File, a comprehensive compendium of hacker slang illuminating many aspects of hackish tradition, folklore, and humor."

5La madre di Internet: voluta dal Dipartimento della Difesa americano, la gestione venne affidata ad ARPA che costruì una rete di soli 4 nodi basata sulla pariteticità dei nodi stessi, ovvero sull'assenza di un nucleo centrale di controllo in modo da garantire il funzionamento di ogni sotto rete in caso di guerra. ARPANet divenne un mezzo di comunicazione per le università solo verso la fine degli anni settanta, quando inglobò al suo interno la neonata Internet Protocol Suite – un mezzo di comunicazione per sistemi di trasmissione eterogenei. Nel 1984, data l'enorme mole di dati presente su di essa, ARPANet venne scissa in due: Milnet, per scopi prettamente militari e Arpanet per scopi scientifici. Il Dipartimento di Difesa americano smise di finanziare la rete universitaria,

Il software veniva scambiato liberamente e i programmatori venivano pagati per l'atto della programmazione, non per i programmi veri e propri. Soltanto con l'entrata sul mercato dei computer all'inizio degli anni ottanta i programmatori cominciarono a limitare i diritti d'uso dei loro programmi e a farsi pagare per ogni copia venduta. Il flusso di comunicazione interna al gruppo di informatici era dunque interrotto da esigenze legate ai diritti che i nuovi proprietari vantavano sui software di nuova produzione.

In questo quadro Stallman si inserisce dunque come una figura di rottura: il libero uso dei software si pone in netta contro tendenza con la nascita e l'esponenziale sviluppo del settore. Così scrive lo stesso Stallman sulla nascita della Free Software Fundation (Fondazione per il software libero): "Man mano che l'interesse aumentava, altre persone parteciparono al progetto GNU, e decidemmo che era di nuovo ora di cercare finanziamenti. Così nel 1985 fondammo la Free Software Fundation, una organizzazione senza fini di lucro per lo sviluppo di software libero. [...] La FSF accetta donazioni, ma gran parte delle sue entrate è sempre stata costituita dalle vendite: copie di software libero e servizi correlati. [...] Finanziammo lo sviluppo di questi programmi perché il progetto GNU non riguardava solo strumenti di lavoro o un ambiente di sviluppo: il nostro obiettivo era un sistema operativo completo, e questi programmi erano necessari per raggiungere quell' obiettivo."

Usando tutto il lavoro svolto da Stallman e da altri programmatori della comunità della FSF, Linus Torvalds sviluppò quella che sarebbe stata la prima collezione di pacchetti completamente libera e liberamente distribuibile di UNIX: GNU/Linux[6]. Egli scrisse la parte mancante del sistema

che venne gestita dalla National Science Foundation – cambiando nome prima in NFSnet e poi in Internet.

6La prima versione libera di UNIX è BSD, un marchio registrato di Berkeley Software Design, Inc. Ma questa versione utilizza ancora tutte le utility dello UNIX vero e proprio.

ovvero il kernel[7] Linux. Erroneamente ora ci si riferisce a intere distribuzioni di software libero come 'Linux', ma in realtà – come appena spiegato – questi rappresenta unicamente il kernel mentre GNU/Linux è il modo corretto per indicare l'insieme di questo e di tutto il software di gestione scritto dalla comunità del progetto GNU – senza il quale il sistema operativo non sarebbe pienamente funzionante e utilizzabile. A questo proposito, Stallman in uno dei suoi articoli scrive: "Molti utenti di computer fanno girare ogni giorno una versione modificata del sistema GNU, ma non se ne rendono conto. Per una singolare combinazione di eventi, la versione di GNU attualmente più diffusa è più spesso nota come "Linux" e molti utenti non sono consapevoli di quanto sia collegata al Progetto GNU. Esiste davvero un Linux; si tratta di un kernel, che queste persone stanno utilizzando. Ma non si può utilizzare un kernel 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 e ha Linux come kernel."[8]. All'interno di questo OS non vi è alcun codice licenziato in modo proprietario o commerciale e questo lo rende il candidato ottimale per poter utilizzare un computer con solo ed esclusivamente software libero. Nell'ottobre del 1991 Torvalds rilasciò la prima versione di Linux definita 'ufficiale': la 0.02. Era il primo passo verso un sistema operativo completo, compatibile con gli UNIX proprietari, ma più robusto, sicuro e totalmente libero[9]. Oltre al kernel venivano distribuiti una scarna (ancora per poco) shell di comandi

7Il **kernel** è la parte centrale di un sistema operativo. Il suo compito è quello di gestire processi, memoria, periferiche e accesso al file system. È l'anello di congiunzione tra l'hardware e i programmi che agiscono su di esso.

8Cfr. http://www.gnu.org/gnu/linux-and-gnu.it.html.

9Vi era stato in precedenza un'altra piccola versione di Unix creata da Andy Tanenbaum: Minix. Questa era stato creata più per scopi didattici che a fini di reale utilizzazione. Proprio da questo infatti Torvalds prese ispirazione e annunciò sul newsgroup comp.os.minix di voler rilasciare per le architetture i386 'a better Minix than Minix' (un Minix migliore di Minix stesso) e chiudeva il suo messaggio con "[...]I didn't want to stop until I could chuck out Minix" (e non smetterò finchè non avrò sorpassato Minix).

denominata *bash* e il compilatore *gcc* entrambi rilasciati dal progetto GNU. Molti programmatori aderiranno entusiasti al progetto, e daranno vita a tutte le moderne distribuzioni di GNU/Linux che ancora oggi vengono sviluppate attivamente. Tra le quali, sicuramente va menzionata per la sua importanza 'filosofica' la distribuzione di Ian Murdock e Bruce Perens: Debian. Questa infatti è l'unica distribuzione che ha come obiettivo primario la sua totale libertà dal software proprietario, non vi si troveranno infatti software gratuiti e/o soggetti a licenze non libere. Ma questo non significa che non possono comunque essere installati tramite appositi server internet comunque compresi nei lista dei server ufficiali della distribuzione: solo bisognerà rispondere esplicitamente – in modo affermativo, naturalmente - alla domanda 'vuoi che del software non libero venga installato nel tuo computer?'.

"La gente dovrebbe avere più libertà e dovrebbe imparare ad apprezzarla" - così Bruce Perens, l'autore della carta dei diritti di utilizzo e divulgazione della distribuzione Debian GNU/Linux ossia il Contratto Sociale Debian e la Guida Debian del Free Software (1997), riporta le premesse di Stallman nel suo Open Source Definition; lo stesso Stallman aveva lasciato addirittura il suo impiego al MIT per poter garantire la completa libertà del suo lavoro.

L'evento che ha dato il via al diffondersi dell'espressione "open source" è stato il rilascio nel 1998, del codice sorgente del browser Netscape da parte di Eric Raymond. Quest'ultimo, insieme a Bruce Perens, prese contatti con uomini di affari della neo nascente industria Linux e modificò la carta di Perens per eliminare ogni riferimento specifico a Debian: nacque la Open Source Definition.

La nascita dell'interesse di Raymond nei confronti dell'OS è descritta nel suo famoso libro-manifesto "La Cattedrale e il bazar": come si legge dal titolo, egli paragona la normale fase di progettazione software tipica dell'industria proprietaria ad una cattedrale 'da costruire

in riverenza e silenzio' e dall'altro lato il metodo usato per Linux ad un bazar, ovvero un luogo caotico dove tutti collaborano senza dover sentire la necessità intrinseca di un unico centro di controllo. Grazie all'esempio pratico, proposto da Linus Torvalds, secondo Raymond viene a cadere la convinzione che oltre un certo limite sia necessario un approccio centralizzato a priori. Dopo questa presa di coscienza, era necessario mettere nero su bianco: "Fin dall'inizio pensammo di avere bisogno di una definizione, di una specie di una meta-licenza che definisse il termine di Open Source. Elaborammo così il documento Open Source Definition che si ispira alla Debian Free Software Guideline originariamente scritte da Bruce Perens."[10] Vi furono però dei dissapori all'interno della stessa comunità del freesoftware. All'inizio per la scelta del nome, che molti ritenevano inadatto per varie ragioni, e successivamente per la mancanza di enfasi sulla libertà del progetto denunciata dallo stesso Stallman che ne era stato escluso nonostante ne fosse il progenitore. Erroneamente gli operatori del settore contrapposero la figura di Raymond a quella di Stallman, contribuendo a identificare il primo con la realtà business di Linux, e con il secondo con il lato più filosofico e politico. Di fatto, sia il movimento Open Source sia l'associazione per il Software Libero avevano e avranno sempre un nemico in comune: il software proprietario, cioè quello il cui utilizzo, ridistribuzione o modifica sono proibiti o richiedono un permesso.

1.2 Software libero e proprietario: vantaggi e svantaggi, dubbi e certezze

Si può definire il *codice sorgente* come il linguaggio "umano" in cui vengono scritti i programmi, in contrapposizione al programma

10 http://www.apogeonline.com/openpress/doc/cathedral.html traduzione di Bernardo Parrella per Apogeo srl.

binario, ossia "compilato", che è invece una sequenza di codici esadecimali difficilmente comprensibili all'uomo senza l'utilizzo di tecniche particolari (il cosiddetto *reverse engineering)*. Avere a disposizione solamente il programma compilato comporta quindi una serie di svantaggi così riassumibili:

- non conoscere nulla riguardo alla sua struttura;
- ignorare quali potrebbero essere i suoi scopi oltre a quelli mostrati dal suo output ufficiale;
- non poter apportare alcuna eventuale modifica volta a migliorarlo e/o ad adattarlo a proprie esigenze specifiche.

Viceversa, la Open Source Definition è molto chiara al riguardo, e un qualsiasi software per poter essere definito "open source" deve necessariamente:

"[...] includere il codice sorgente e deve consentirne la distribuzione tanto in codice sorgente che in forma compilata. [...] Il codice sorgente deve essere la forma preferenziale nella quale un programmatore modifichi un programma. [...] 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."

Dall'altro lato la definizione di software libero della Free Software Foundation (FSF) è molto simile in quanto garantisce questi quattro diritti fondamentali per ogni program ma:

- eseguirlo;
- studiarlo e adattarlo alle proprie esigenze;

- copiarlo e ridistribuirlo liberamente;
- effettuare modifiche volte a migliorarlo e quindi distribuirne i cambiamenti a tutta la comunità.

Dal punto di vista prettamente pratico non vi sono differenze sostanziali tra i programmi open source e il software libero. È l'idea del mondo della programmazione software che ne è alla base ad essere diversa: il primo ha un taglio più pratico, rivolto alle aziende, mentre il secondo è più "filosofico", rivolto ai programmatori e agli addetti ai lavori. Come scrive lo stesso Stallman: "[...] Siamo in disaccordo sui principi di base, ma siamo più o meno d'accordo sugli aspetti pratici. [...] Non vediamo il movimento Open Source come un nemico. Il nemico è il software proprietario. Noi non siamo contro il movimento Open Source, ma non vogliamo essere confusi con loro."[11]. Di fatto la confusione temuta c'è stata: abbiamo assistito nel corso degli anni alla completa commistione dei due termini. Vero è anche che, nonostante la genesi diversa dei due movimenti, la fioritura degli stessi si è intrecciata più volte nel corso del tempo e in alcuni punti è legata indissolubilmente.

La polisemia della parola *free* in inglese, che significa sia "libero" sia "gratuito", ha fatto sì che la definizione di software libero – in inglese *free software* appunto – ingenerasse più di una volta confusione tra il prezzo del programma e le sue libertà di utilizzo. Per fruire a queste ultime, è necessario accedere al codice sorgente del programma: ciò però non vuol dire necessariamente che il software libero debba essere gratis o non commerciale. A riprova di ciò, molti tra i produttori di software libero lavorano nel campo software commerciale.

Inoltre, anche se il software commerciale è sviluppato a scopo di lucro, ciò non implica necessariamente che non possa essere open source. Se infatti è vero che la maggior parte del software commerciale

11Cfr. http://www.gnu.org/philosophy/free-software-for-freedom.it.html.

è anche proprietario, è anche vero che esiste software libero commerciale e software non commerciale ma non libero.

Non bisogna pensare che il software libero sia esclusivo appannaggio di pochi programmatori addetti ai lavori; l'utente finale trae svariati vantaggi nel preferire software open source a software proprietario.

Innanzitutto i costi di acquisto di una licenza software closed source – di solito ad una licenza è associato uno e un solo computer – non sussistono quando si parla di software libero, quindi si ha una minore spesa iniziale; spesso e volentieri anche le spese dei vari servizi di assistenza (migrazione, formazione, installazione e gestione) sono molto ridotti rispetto a quelli della loro controparte closed source, quindi si ha anche un investimento più oculato nel futuro.

Un altro vantaggio rilevante è costituito dall'indipendenza dell'utente dal fornitore del software. Trattandosi di software open source infatti, non si è legati ad un unico installatore di servizi, che può costringere ad un aggiornamento oppure ad un programma di assistenza vincolata. Chiunque può subentrare con offerte più vantaggiose rispetto a quelle del precedente fornitore senza dover migrare da un software ad un altro e senza dover perdere tutto il lavoro già realizzato. Questo avviene grazie all'uso di formati aperti, al contrario di quelli che si utilizzano di default nei programmi proprietari. L'uso di formati chiusi in effetti trasferisce la proprietà del documento al proprietario del formato: l'utente non ha più controllo sulle informazioni effettivamente contenute nel file che sta distribuendo, rischiando una perdita di privacy non indifferente.

D'altro canto il formato aperto è garanzia di massima raggiungibilità dei destinatari: chiunque potrà aprire il documento, senza essere vincolato all'acquisto di un software specifico. Per fare un esempio, all'interno della pubblica amministrazione l'uso dei formati

aperti in un primo tempo slega i cittadini dall'acquisto dei software necessari alla lettura dei documenti e, cosa ancora più importante, libera le stesse amministrazioni dalla catena dell'*acquisto per comunicazione*: non è necessario rinnovare il parco software per poter permettere ad un ente pubblico di leggere i documenti inviati dal comune stesso.

Ogni azienda tiene in gran conto la sicurezza dei propri dati e di quelli dei propri clienti. Conoscere il codice sorgente di un software rappresenta un ulteriore vantaggio, in quanto, padroneggiando esattamente tutto il funzionamento interno del software installato, si possono effettuare agevolmente controlli interni per risolvere eventuali malfunzionamenti o mancanze che potrebbero essere utilizzate come facili vie di accesso da un ipotetico aggressore. Inoltre si può far controllare da terzi la soluzione software acquistata allo scopo di valutarne le caratteristiche, quali ad esempio la robustezza oppure la portabilità. È sbagliato credere che nascondendo informazioni sul software utilizzato si abbia una maggiore sicurezza, poiché un codice libero permette a chiunque di controllarlo e soprattutto di apportare delle modifiche per aumentarne la sicurezza. Premesso che nel corso degli anni si è giunti alla consapevolezza che non esiste un software totalmente sicuro, se solamente poche privilegiate persone hanno accesso ai sorgenti il software sarà ancora più svantaggiato rispetto ad un possibile aggressore, poiché basterà un errore umano per comprometterne tutta la sicurezza. Viceversa molti software liberi sono stati testati e utilizzati così intensamente dalla comunità open source che si possono definire ragionevolmente sicuri.

Come già accennato in precedenza, utilizzando software open source vi è maggiore libertà di personalizzazione e di espandibilità della soluzione software scelta, anche da parte di terzi e non necessariamente da parte del fornitore iniziale - il tutto senza che vi siano da pagare

licenze particolari.

Fino a qualche tempo fa la critica che veniva mossa più frequentemente al software libero era la sua bassa compatibilità con gli standard commerciali: nell'ambito dell'attuale mercato IT, il modello di distribuzione del software open source è rischioso poiché esso è, economicamente parlando, basato sull'offerta di servizi piuttosto che sulle licenze. Bisogna ricordare che questi timori si vanno dissipando da quando grandi multinazionali del mercato dell' IT si sono avvicinate al mondo del software libero, utilizzandolo per fornire soluzioni di gestione, installazione e assistenza a basso costo. Questo avvicinamento ha fatto sì che gli utenti fossero garantiti con un'assistenza adeguata per le loro soluzioni software; prima infatti si parlava di Customer Service e di assistenza post-vendita scarsi o del tutto assenti, ora sono nati molti servizi, anche a pagamento, che offrono assistenza per tutte le necessità.

Dal punto di vista della compatibilità con il corrispettivo mondo proprietario, il software libero ha fatto passi da gigante negli ultimi anni: tutti i principali formati dei dati sono supportati e questo ha permesso a molte aziende di migrare dalle loro precedenti soluzioni software proprietarie a quelle libere senza perdere nulla nella migrazione, sia dal punto di vista dei dati, sia da quello delle funzionalità.

Ultimamente anche la compatibilità con alcune piattaforme è stata risolta e la portabilità del software libero è stata notevolmente elaborata per consentirne un utilizzo anche sulle piattaforme a venire. Purtroppo, siamo ancora lontani dalla liberalizzazione delle specifiche delle periferiche hardware, per quanto in relazione a questo aspetto esista già un progetto (proprio del su citato Bruce Perence) denominato *Open hardware*. Infatti la mancanza di driver per le periferiche hardware – soprattutto quelle più recenti - resta una nota dolente per gli utenti di Linux. Fondamentalmente i produttori di periferiche rilasciano immediatamente i driver per i sistemi operativi proprietari, e solo

successivamente coprono il fabbisogno di quelli liberi. Anche nel settore hardware si sta tuttavia modificando questa tendenza specialmente da quando il software libero, ed in particolare Linux, sta conquistando delle discrete fette di mercato dell'IT.

Particolare attenzione da parte delle aziende è data alla curva di apprendimento di un nuovo software, che comunque costringe i dipendenti ad un periodo di inattività causa aggiornamento; anche in questo verso sono stati effettuati grandi miglioramenti da parte della comunità open source in modo tale da avvicinarsi il più possibile a quegli standard proprietari che hanno tenuto il mercato in un quasi completo monopolio fino ad oggi.

1.3 L'Open Source e le azioni dell'Unione Europea

Il piano d'azione italiano definito nell'ambito del Sesto Programma Quadro, è volto ad incentivare gli investimenti nel settore dei servizi pubblici all'interno dell'Unione Europea, incrementando l'occupazione e modernizzandone le strutture, per favorire l'accesso dei cittadini alla *information society.* All'interno di questo piano d'azione viene considerato anche l'apporto positivo del software *open source:* "Le attività di ricerca promuovono inoltre la normalizzazione delle tecnologie al fine di diffondere l'uso di standard aperti e di software *open source."*[12] e ancora, più sotto "I modelli o linee guida avranno carattere modulare, saranno adattabili in funzione del tipo di utente e consisteranno di norma in una metodologia corredata di una serie di strumenti e di software di

12 COMUNICAZIONE DELLA COMMISSIONE AL CONSIGLIO, AL PARLAMENTO EUROPEO, AL COMITATO ECONOMICO E SOCIALE E AL COMITATO DELLE REGIONI, eEurope 2005: una società dell'informazione per tutti, Piano d'azione da presentare per il Consiglio europeo di Siviglia, 21 e 22 giugno 2002 consultabile al sito http://europa.eu.int/information_society/eeurope/2002/news_library/documents/eeurope2005/_eeurope2005_it.pdf.

tipo *open source".*

Il Sesto Programma Quadro riporta: "The development of open standards and open source software will be encouraged when appropriate to ensure interoperability of solutions and to further innovation"[13], ma già dal 1998 era stato istituito il *Working group on Libre software*[14], che aveva prospettato le enormi potenzialità di sviluppo di quelle imprese che avessero adottato software libero, e conseguentemente, nel successivo Quinto Programma Quadro (FP5) sono stati approvati diversi progetti in questo settore.

Per quanto riguarda le applicazioni generiche per l'educazione nell'ambito appunto del FP5, ricordiamo qui il progetto ITCOLE (Innovative Technology for Collaborative Learning and Knowledge Building)[15] centrato sullo sviluppo di modelli di insegnamento innovativi nell'ambito del *collaborative learning* attraverso l'utilizzo di apposite piattaforme informatiche.

Segnaliamo infine per ragioni di completezza, due documenti che sono utili per approfondire le tematiche dell'OS a livello europeo, ossia un glossario di termini chiave[16] in cui vengono riassunte le differenze tra Free software e OS, e i risultati di un censimento dell'utilizzo dell'OS nei governi e nelle istituzioni dei paesi europei[17], il *Free/Libre and Open Source Software: Survey and Study* condotto con il patrocinio dell'UE da Germania e Paesi Bassi. Nel glossario viene esemplificato attraverso un diagramma il rapporto tra OS e Free Software (figura 1).

13Cfr. http://www.cordis.lu/ist/workprogramme/fp6_workprogramme.htm.
14Cfr. http://eu.conecta.it/.
15Cfr. http://www.euro-cscl.org/ .
16Cfr. http://europa.eu.int/information_society/activities/opensource/doc/pdf/key_terms.pdf
17Cfr. http://www.infonomics.nl/FLOSS/report/.

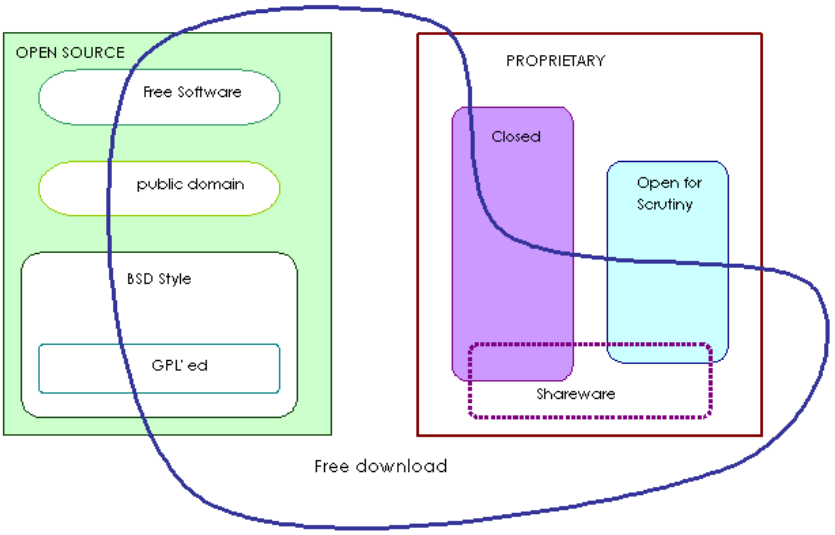

Figura 1 – *Rapporto tra Free Software e Open Source*[18]

Come si vede nella figura 1, Free Software e Open Source rappresentano due facce della stessa medaglia. La differenza più consistente è nelle diverse motivazioni enfatizzate dai due movimenti.

L'OS è in un certo senso una applicazione pratica della filosofia del software libero. Lo stesso Richard Stallman sostiene "The fundamental difference between the two movements is in their values, their ways of looking at the world".

Ad un livello pratico, a patto di trascurare dei dettagli secondari, entrambi gli appartenenti alle comunità dell'Open Source Software e del Free Software sono concordi su una base comune di licenze.

18 Fonte: http://europa.eu.int/information_society/activities/opensource/doc/pdf/key_terms.pdf.

Va comunque ricordato che avere l'accesso al codice sorgente per analizzarlo non basta per poter definire un dato software "Open Source", come finora è stato inteso questo termine. In realtà sono i diritti che si associano normalmente alla definizione di OS o FS - come la libertà di adattamento, test o di ridistribuire ad altri - che fanno la differenza tra la mera possibilità di visualizzare il codice sorgente e poter definire un software Open Source o Free Software.

Ad esempio, in passato vi sono state delle iniziative di software house commerciali e proprietarie che hanno permesso di visionare il codice sorgente dei loro programmi ma ciò richiedeva di firmare un accordo esclusivo prima di poter avere l'accesso ai codici suddetti e questo non conferisce assolutamente i diritti di Open Source Software o Free Software a quei programmi.