

L'Open Source come modello economico

Stefano Stabellini
Diego Piacentini



INTRODUZIONE

Il fenomeno e la sua storia

L'Open Source è un fenomeno di larga diffusione relativamente recente e ancora oggi oggetto di accese polemiche in ambito politico, tecnico ed economico. Prima di parlare del suo risvolto in quest'ultimo campo è opportuno definire al meglio di cosa si tratta.

Sin dagli anni '60 era una pratica abbastanza diffusa negli ambienti di ricerca americani vicini al mondo UNIX, lasciare completamente libero accesso al codice sorgente del software sviluppato, e questa pratica continuò indisturbata sino agli anni '80 quando l'AT&T iniziò a reclamare i propri diritti intellettuali sul codice sviluppato dai propri dipendenti.

Per questo motivo si iniziò a sentire il bisogno di proteggere il codice prodotto, pur mantenendo la possibilità di renderlo accessibile a tutti.

Richard Stallmann, un ricercatore del MIT, decise di fondare un'organizzazione per la diffusione del "Software Libero", la Free Software Foundation, oltre ad un'organizzazione per la produzione di tale software, la GNU (GNU's Not UNIX). Con l'aiuto di alcuni specialisti, scrisse una licenza che avrebbe fatto storia, la GPL (GNU General Public License) per veicolare la distribuzione di tale software. I programmi che vengono distribuiti sotto licenza GPL sono Software Liberi e, come garantisce la licenza stessa, gli utenti possono leggere il codice sorgente, modificarlo, ridistribuirlo a piacere, anche nella versione da loro modificata. Il punto fondamentale però è che ogni versione da loro modificata del codice deve comunque essere ridistribuita con licenza GPL: per questo motivo tale licenza è stata da molti definita "virale".

Garantendo questi diritti agli utenti il software è veramente libero quindi "free, like in a speech and not like as a beer", come Stallman si preoccupava di far sapere, data l'ambiguità del termine inglese (in realtà in italiano le parole ci consentono di specificare meglio: libero, non gratuito). Per chiarire al meglio le idee portiamo l'esempio di un software che è diventato standard perché "relativamente aperto", gratuito, ma non libero: l'Acrobat Reader. E portiamo anche l'esempio di un software che è libero, ma non è assolutamente gratuito: Red Hat Enterprise Linux.

Il fatto che il software libero sia generalmente anche gratuito è un fatto abbastanza incidentale, sebbene sia dovuto certamente anche alla difficoltà di vendere, secondo il senso tradizionale del termine, i propri prodotti, visti i diritti che si è costretti a garantire ai clienti (in particolare la possibilità di ridistribuire il codice).

Negli anni successivi alla Free Software Foundation di Stallman si affiancò l'Open Source Initiative che, in maniera un po' più pragmatica, cercò di applicare i principi fondamentali prima esposti al mondo del software, definendo in maniera un po' più precisa quali caratteristiche dovesse soddisfare una licenza software per essere considerata software aperto. D'ora in poi ci riferiremo al software Open Source senza addentrarci in varie definizioni e piccole differenze.

La vera e propria rivoluzione che ha portato il movimento al successo, aggiungendone nuove e importanti caratteristiche, è però in realtà dovuta a Linus Torvalds, uno studente universitario finlandese che lanciò nel 1991 il progetto Linux, che avrebbe dovuto portare alla realizzazione di un intero sistema operativo conforme agli standard UNIX. Egli introdusse infatti il concetto di Bazaar.





BAZAAR

Lo schema di sviluppo aperto

Normalmente il software commerciale viene sviluppato tramite un'organizzazione a "cattedrale", completamente chiusa; le versioni rilasciate del software sono quelle complete e la responsabilità si distribuisce in maniera gerarchica, così come le decisioni sullo sviluppo. Torvalds invece portò avanti Linux valorizzando il contributo degli utenti, in maniera collaborativa, rilasciando il prima possibile.

Se si vuole seguire il Bazaar come modello di sviluppo, innanzi tutto è importante la presenza di un leader carismatico del progetto che, più che grandi abilità di programmazione (che comunque sono importanti), deve avere grandi abilità di comunicazione per tenere unito il progetto e destare l'interesse degli utenti, favorendone la collaborazione.

Egli, dopo aver sviluppato un primo prototipo del progetto, ma già funzionante nei suoi aspetti basilari, deve rilasciare il più di frequente possibile nuove versioni con piccoli miglioramenti, chiedendo di volta in volta il parere degli utenti e spingendoli a collaborare. In questo modo deve favorire la nascita dell'interesse da parte dei suoi clienti, gestire accuratamente le loro aspettative nel progetto, e quindi creare una fitta rete di sviluppatori e collaboratori tenuta insieme da internet e completamente decentralizzata.

Ottenuta una comunità virtuale di sviluppo, è importante continuare a mantenere attivo l'interesse premiando ogni piccola collaborazione e mantenere unito il gruppo basandosi sul "principio di comprensione condivisa", ovvero rapporti alla pari, poco formali, e basati su spirito collaborativo.

Questo di certo non significa non saper premiare i migliori, anzi: il fatto che il frutto del proprio lavoro sia praticamente in esposizione al mondo intero, con la possibilità che tutti possano vederlo e criticarlo, è un fattore determinante, come vedremo meglio in seguito.

Sfruttando il fatto che il codice è aperto si ottengono altri benefici fondamentali: per incominciare è possibile riusare il codice altrui già prodotto, e questo è uno dei fattori più importanti, che partecipa in maniera fondamentale anche alla riduzione dei costi.

Distribuire presto e spesso ha proprio come scopo quello di minimizzare la duplicazione di lavoro propagando rapidamente le soluzioni giunte col feedback degli utenti.

Inoltre questi ultimi nel momento in cui provano il software sono in grado di trovare ed identificare al meglio gli eventuali problemi e bug riscontrati, diventando un'autentica risorsa per il team di sviluppo. In altri termini "il debug viene parallelizzato" migliorando complessivamente la qualità del software prodotto ed ancora una volta i costi per produrlo.

Il tutto si materializza quindi in una completa e incessante attenzione nei riguardi dell'utente, che viene eletto a una sorta di co-sviluppatore.

Ma cosa spinge decine o centinaia di utenti a collaborare a un progetto altrui? Si tratta di altruismo? A lungo molti non sono riusciti a spiegarsi l'innata capacità di scoprire talenti dell'Open Source, non capendo che alla base vi è un diverso sistema di incentivazione dei programmatori.

Prima di tutto bisogna chiarire che coloro che decidono di partecipare a progetti di questo tipo sono solitamente persone pienamente interessate allo sviluppo dello stesso, quindi più motivate e appassionate, a partire dal leader. Questo è il motivo per cui il vantaggio della prima mossa nello sviluppare prodotti simili nell'ambito dell'Open Source è ancora più marcato, perché le persone più interessate e probabilmente anche le più indicate a collaborare saranno state già coinvolte nel primo progetto, e difficilmente lo abbandoneranno per un altro.

Ma indaghiamo più a fondo su quali sono i benefici che un programmatore qualunque ha nel partecipare ad un progetto, open o closed source che sia.

Vi sono innanzitutto una serie di benefici a breve termine, come una ricompensa in denaro, e il miglioramento del software stesso, che può essere utile anche per il soggetto in questione; contro una necessaria perdita di tempo nell'occuparsi dello sviluppo. I vantaggi sembrano a favore del

software commerciale che può meglio garantire una stipendio allo sviluppatore, ma non bisogna trascurare il fatto che nei progetti aperti il costo di sviluppo risulta essere decisamente inferiore grazie alla succitata parallelizzazione del debugging e al fatto che il software aperto, proprio per il fatto di essere aperto, viene insegnato nelle scuole e nelle università, e perciò risulta più familiare al programmatore. Fattore chiave in questo senso è, come abbiamo già visto, anche una decisiva possibilità di riuso dei terabyte di codice open source oramai a disposizione. Come afferma Raymond in “The Cathedral and the Bazaar”:

“I bravi programmatori sanno cosa scrivere. I migliori sanno cosa riscrivere (e riusare).”

Il vero vantaggio dell'Open Source nei confronti del software proprietario negli incentivi sugli sviluppatori si ottiene però nei “benefici a lungo termine”, che riguardano benefici di carriera e di acquisizione di fama presso la comunità.

Vari studi hanno portato alla luce che tali vantaggi sono tanto più forti quanto più è visibile il lavoro svolto agli utenti, più tale lavoro è significativo nei riguardi del talento dello sviluppatore, e maggiore è l'impatto degli sforzi effettuati sul prodotto. In un modello di sviluppo Open Source c'è sicuramente una maggiore possibilità di effettuare analisi sulla performance del proprio lavoro da parte di tutti; la possibilità di avere piena iniziativa nei riguardi della sottoparte del progetto di cui si ha responsabilità; e infine maggiore possibilità di trasferire conoscenza e risorse umane tra progetti differenti. Ne consegue che i programmatori più sensibili a questo tipo di incentivi sono decisamente più portati a partecipare allo sviluppo di software aperto.

Questo è ciò di cui parla Raymond in “The Cathedral and the Bazaar”:

“The utility function Linux hackers is maximizing is not classically economic, but is the intangible of their own ego satisfaction and reputation among other hackers. [...] Voluntary cultures that work this way are actually not uncommon; one other in which I have long participated is science fiction fandom, which unlike hackerdom explicitly recognizes egoboo (the enhancement of one's reputation among other fans).”

Il sistema degli incentivi non è cosa da sottovalutare e a dimostrare la sua importanza vi è il fatto che numerose imprese commerciali abbiano tentato, spesso con scarso successo, di riprodurre questo tipo di benefici nei riguardi dei loro dipendenti: Eudora, il noto client mail, ha iniziato ad esporre i nomi degli sviluppatori dei suoi prodotti, cosa che di certo è di per sé controproducente, in quanto aumenta la possibilità che altre imprese si interessino ai suoi talenti.

Altri hanno provato ad aumentare la circolazione del codice all'interno della stessa società, ottenendo spesso scarsi risultati, visto il differente approccio che sta alla base del modo di lavorare dei programmatori in ambiente aperto e non.

Dalla discussione emerge, in maniera ancora più decisiva, l'importanza della presenza di un leader sufficientemente carismatico in progetti Open Source, perchè tenere unito un gruppo spinto dal desiderio di acquistare visibilità può rendersi difficoltoso, e il rischio di forking, ovvero la nascita di nuovi progetti da quello in corso, per seguire un diverso percorso di sviluppo, è sempre dietro l'angolo. Un esempio famoso è quello dei BSD UNIX, sistemi operativi anch'essi Free sviluppati sotto licenza BSD: in questo caso dal progetto originario (NetBSD) sono nati 2 progetti figli (OpenBSD e FreeBSD), la cui presenza è stata appunto spiegata con la mancanza di un leader sufficientemente forte.

Un'altra importante conseguenza del sistema di incentivazione, ma ancor di più dell'intero sistema di sviluppo che aderisce al modello aperto, è una maggiore attenzione per gli utenti esperti rispetto a quelli inesperti. I primi infatti, sono maggiormente in grado di contribuire al progetto, e apprezzano al meglio gli sforzi dei singoli sviluppatori. Torvalds stesso sostiene che il modello dà il massimo in



quei settori in cui l'utenza del prodotto che si va a sviluppare è un'utenza esperta o sono altri sviluppatori, come per esempio il software per i server e gli amministratori di sistema.

Il fatto che storicamente il target dei propri progetti sia esperto, ha portato nel tempo gli sviluppatori a disegnare delle interfacce utente tipicamente a carattere o comunque comode per i sistemisti ma purtroppo decisamente scomode per i newbie. Molte sono le imprese che a considerazione di questo hanno basato il loro business sulla vendita di GUI (Graphical User Interface) più comode o in generale tools per la semplificazione delle operazione per gli utenti. Questo aspetto può essere utilizzato per operare un corretto versioning del proprio prodotto, e va considerato strategicamente; nella stragrande maggioranza dei casi l'utenza base non può e non deve essere trascurata, e ormai da diversi anni i team di sviluppo dei progetti Open Source hanno imparato, e stanno imparando, ad ascoltare anche questo tipo di clientela.

La nascita da diversi anni di progetti come KDE e Gnome ne è la testimonianza, e il fatto che i risultati che hanno ottenuto e stanno ottenendo siano di ottima qualità, è un ulteriore prova che non bisogna porre limiti al campo di applicazione del modello a bazaar.



IL MODELLO ECONOMICO

I conti in tasca all'Open Source

A questo punto ci troviamo a dover prendere in considerazione un aspetto cruciale: preso atto dell'esistenza del fenomeno Open Source, come dobbiamo porci nei confronti di esso? La sola presenza di software Open Source sul mercato, e la facile reperibilità di questo grazie alla Rete, ci dovrebbero spingere a rivolgere la nostra attenzione ad esso, in qualsiasi ruolo noi ci troviamo.

Per prima cosa, proviamo ad osservare la situazione "dall'interno".

Non possiamo non notare la differenza che contraddistingue la struttura dei costi di un'azienda che sviluppa software secondo il modello a cattedrale, descritto sopra, rispetto a quella di un'organizzazione che si appoggia su software sviluppato secondo il modello a bazaar.

Innanzitutto, dobbiamo mettere in conto che in entrambi i casi per portare avanti lo sviluppo di un software abbiamo bisogno di un gruppo di programmatori che si dedichino a tempo pieno al progetto. E qui finiscono le analogie.

Come abbiamo visto, il ruolo di questi programmatori è diverso nei due casi: in un ambiente di sviluppo a cattedrale, a questo gruppo è richiesto di svolgere tutto il lavoro, dalla progettazione al debugging, fino al momento in cui il software non sia giunto ad un livello di efficienza e di funzionalità soddisfacente per gli obiettivi che ci eravamo preposti, e tale da permetterne la commercializzazione.

In un bazaar avremo ugualmente bisogno di un team di sviluppatori, in quanto non possiamo fare affidamento solo sulla comunità: per portare avanti efficacemente un progetto, è importante che ci sia qualcuno che coordina e raccoglie le fila del lavoro di tutti, e queste persone rappresentano per l'azienda un costo fisso.

Ma se in un progetto sviluppato a cattedrale questo gruppo deve necessariamente essere composto da un certo numero di persone, nel caso di un bazaar ci si può permettere di appoggiarsi su un gruppo più ridotto, a cui vanno affiancati tutti i potenziali cosviluppatori rappresentati dagli utenti, o dai programmatori che sapranno dimostrare le proprie competenze e vorranno autoproporsi per aiutare lo sviluppo attivamente.

I costi si contraggono significativamente anche grazie alla miglior efficienza del processo di sviluppo, che si manifesta nelle varie forme sopra esposte; vi è poi una riduzione delle spese dovuta alla totale assenza di costi di distribuzione poichè essa può avvenire direttamente on-line.

Tutt'altro che trascurabile, se si sviluppa per un sistema aperto come Linux, è il potere di distribuzione delle varie Distribuzioni appunto, che raggiungono mese per mese le case di migliaia di utenti. Esse potranno risultare particolarmente utili anche solamente per farsi conoscere al grande pubblico.

Il marketing va ripensato, si possono ridurre i tradizionali canali promozionali, poichè l'ambiente Open Source è provvisto dei propri, anche qui il più delle volte a costo zero. Il sistema stesso di sviluppo, volto completamente a catturare l'attenzione degli utenti, può essere considerato un meccanismo per guadagnare popolarità, tramite usenet, chat, mailing lists, o nella fattispecie di siti di news specialistici. Non bisogna trascurare poi la visibilità data dal considerevole numero di riviste del settore, che si occupano solo di parlare di progetti aperti.

Certamente gioca un ruolo fondamentale la potenzialità di Internet. Ci veniamo a trovare in una condizione in cui il costo di distribuire e promuovere il nostro prodotto è veramente basso, rappresentato unicamente dalla banda che acquistiamo al nostro ISP.

Un'altra significativa riduzione dei costi fissi dell'impresa deriva dall'assetto organizzativo totalmente decentralizzato, che permette risparmi sugli stabili e dal più efficiente sistema amministrativo, basato su rapporti orizzontali di fiducia poco formalizzati. E' incredibile come un sistema organizzativo molto semplice come questo, possa essere in grado di gestire l'estrema complessità derivante da un team di sviluppo di centinaia di persone come quello del Kernel Project.

Alla base di tutto c'è un'affermazione che molti danno per scontato, ma che è bene definire con chiarezza: “siamo costretti a regalare il nostro prodotto?”

In realtà, le licenze Open Source ad oggi esistenti, in testa a tutte la GPL, non limitano affatto la possibilità di farsi corrispondere un pagamento per il prodotto; però non è più possibile vendere il software nel senso tradizionale del termine, dal momento che tra i diritti che si è costretti a garantire agli utenti, c'è anche la possibilità di ridistribuire il codice. Per questo è necessario affiancare al semplice programma un valore aggiunto, che può essere costituito da servizi, consulenza, o prodotti complementari, come vedremo in seguito.

Utilizzare un modello Open Source a bazaar garantisce quindi un vantaggio di costo non indifferente; ora passiamo all'analisi di altri due aspetti fondamentali del modello che ancora una volta possono produrre sensibili vantaggi.

Una fortissima motivazione che spinge molti utenti a migrare verso l'Open Source è l'azzeramento dei lock-in. Sappiamo che tanto maggiore è il grado di apertura di un sistema, tanto minore è il rischio che gli utenti che hanno aderito a un determinato standard, o che hanno iniziato ad utilizzare un certo formato dati, si trovino legati ad esso. E il massimo dell'apertura possibile si ha proprio con l'Open Source. Gli utenti di questo tipo di software sanno che potranno sempre contare sul supporto del sistema da parte della comunità per gli aggiornamenti, o nel caso peggiore, la disponibilità del codice consentirà loro di approntare soluzioni “fatte in casa” per risolvere i loro problemi.

Se non si contano i costi di esperienza, dovuti all'apprendimento del vecchio sistema e allo sforzo necessario per imparare il funzionamento del nuovo, orientandosi verso l'utilizzo di una tecnologia Open Source ci sono bassissimi costi di transizione, e ci si preserva anche dal rischio di rimanere intrappolati in un lock-in in futuro. Il rovescio di questa medaglia è costituito dal fatto che noi stessi, se forniamo ai nostri clienti servizi basati su sistemi Open Source, non li vincoliamo a noi in alcun modo, consentendo loro di rivolgersi in futuro ad imprese nostre concorrenti con maggiore facilità.

Eppure, paradossalmente, proprio il fatto che i clienti siano meno esposti ai rischi di lock-in gioca a favore il più delle volte delle imprese che utilizzano e promuovono software Open Source, rendendole più appetibili di quelle che utilizzano software proprietario: questo in sostanza fa del Sistema Open Source il fornitore ideale di qualsiasi impresa o cliente.

E' evidente che a parità di tutte le altre condizioni, chiunque sia in grado di effettuare una libera scelta, preferirebbe un'opzione che in futuro di per certo non si rivelerà un vicolo cieco, che esporsi al rischio di essere “sfruttato” dall'impresa che ha scelto, perchè non più in grado di liberarsene.

Sviluppando secondo una strategia aperta ci si trova anche maggiormente al riparo dai rischi che deriverebbero dall'entrata sulla scena di un concorrente, se la nostra tecnologia è di buona qualità. Molto difficilmente, per i motivi già esposti, un altro progetto Open Source analogo al nostro vedrà la luce; ma anche la nascita di una tecnologia proprietaria in concorrenza con la nostra è meno probabile di quanto lo sarebbe se il nostro progetto fosse chiuso. Sempre per l'assenza di lock-in, se la nostra tecnologia è valida, chiunque abbia interessi nel nostro stesso settore preferirà investire nel progetto che noi abbiamo iniziato piuttosto che iniziarne uno proprio. Senza considerare che oltre ai costi di transizione, anche quelli di transazione, legati alla scrittura del contratto e al monitoraggio che questo venga rispettato, sono davvero minimi. In pratica, ad un'impresa che intende rafforzare un suo prodotto che si basi su qualche progetto Open Source, non serve altro che fare qualche donazione al progetto stesso.

Molti quindi possono decidere di appoggiarsi attivamente al software Open Source per le loro necessità, portando avanti progetti paralleli oppure finanziando il gruppo iniziale di sviluppatori. Si viene a creare, insomma, un vero e proprio network di imprese, ognuna indipendente dall'altra, ma tutte in collaborazione attiva attorno ai progetti di interesse comune: chiaramente, anche per questi valgono le stesse considerazioni fatte in precedenza, sull'appetibilità dell'Open Source in quanto nessuna delle imprese in collaborazione potrà mai far valere strani diritti di proprietà sul software utilizzato dalle altre.

Ci sono molti esempi di organizzazioni che vivono su questo modello: in primis l'Apache Software Foundation, la software house che tra gli altri progetti, sviluppa anche il server web più diffuso al mondo. L'IBM e la

Sun, entrambe imprese che sono attive nel mercato dei server di rete, hanno preferito investire nel progetto Apache per favorire la crescita del prodotto principale sul quale esse vendono hardware, servizi e supporto. La stessa cosa accade per i progetti Gnome e KDE, aventi come obiettivo quello di creare un Desktop Manager bello e semplice da usare; sono lautamente sovvenzionati da tutte quelle Distribuzioni che hanno bisogno di vendere nei mercati Workstation e Desktop, in cui l'interfaccia utente è importante.

Un altro fattore critico, che sfrutta fortemente la mancanza di lock-in, è la maggiore facilità di creazione di reti virtuali di utenti, di particolare importanza in quei settori soggetti a forti esternalità di rete. Sin dalle prime fasi dello sviluppo, i nostri potenziali utenti saranno informati dell'esistenza del nostro progetto e, se interessati, seguiranno le migliori man mano che questo viene portato avanti, magari partecipandovi attivamente. Si può venire a creare, in questo modo, una rete di utenti di un determinato software ancora prima che il software stesso sia pronto da utilizzare, utenti che saranno spesso sul sito su cui vengono pubblicati notizie e release, e che quindi possono diventare una risorsa anche in termini di introiti derivanti dalla pubblicità.

In questo modo è molto più facile raggiungere la massa critica ed innescare così il feedback positivo, che permetta alla nostra tecnologia di imporsi.

Certo, una necessità alla base della creazione di una rete estesa che possa dare realmente valore al software è la capacità di spingere gli utenti a credere in quel determinato progetto; come abbiamo infatti già sottolineato, la figura del leader è fondamentale.

Sarà sicuramente anche più facile convincere gli utenti a provare una tecnologia dando loro la possibilità di analizzarla fin nei dettagli del codice sorgente, sia che poi essi sfruttino realmente questa possibilità sia che non lo facciano: la sola consapevolezza di poterlo fare ha di per sé un valore.

A questo punto abbiamo capito che se siamo sufficientemente abili ad applicare il modello a bazaar di sviluppo di software Open Source riusciamo ad ottenere ottimi vantaggi di costo ed una larga base di utenti; il problema principale è riuscire a capitalizzarla, perchè, come abbiamo visto, vendere il semplice programma può diventare molto difficile.

Tanto per incominciare dobbiamo pensare creativamente a come vendere prodotti complementari al solo software aperto, per esempio interfacce grafiche che ne semplifichino l'utilizzo e documentazione varia, anche cartacea, fino ad arrivare anche alla vendita dei gadgets destinati a quegli utenti più appassionati. Ci sono imprese, come la stessa Apache, che traggono profitti tramite la vendita di T-Shirts e perfino di mentine alla caffeina a forma di pinguino!

Un'altra possibilità è quella di fare un versioning del nostro prodotto, utilizzando una versione base gratuita per favorirne la diffusione segnalando ai nostri clienti la qualità che rendiamo loro disponibile, risolvendo tra l'altro anche il problema dei beni di esperienza e di informazione.

E' necessario per fare questo aggiungere valore alle versioni avanzate, per esempio tramite assistenza tecnica e servizi vari di consulenza e supporto agli utenti. Questo è il core business di molte aziende come la Red Hat, i cui clienti sono imprese



che hanno bisogno di garanzia di funzionamento dei loro server e che hanno interesse ad acquistare assieme al software anche supporto tecnico.

Non dobbiamo comunque precluderci la possibilità di fare versioning su altre dimensioni, come per esempio il ritardo sulla disponibilità del prodotto; strategia correntemente seguita da MandrakeSoft, che rilascia immediatamente l'ultima versione del proprio sistema operativo solo per gli iscritti a pagamento del club; per tutti gli altri occorre invece aspettare qualche mese.

Una forma molto interessante di versioning è quella di rilasciare lo stesso prodotto sotto due licenze differenti: si può distribuire una versione gratuita Open Source per gli utenti che non la utilizzino per scopi commerciali, ed un'altra versione identica alla prima, ma a pagamento e con una licenza commerciale, per coloro che non intendono utilizzare il prodotto per scopi di Software Libero o educativi.

Riteniamo che questa sia uno dei metodi migliori per ottenere profitti dai propri sforzi, in quanto si sfruttano le differenze naturali tra le necessità dei diversi utenti. Famoso è l'esempio della Trolltech, che sviluppa le librerie Qt e le rilascia appunto il proprio prodotto in dual licensing: una prima versione sotto QPL, una licenza Open Source, e una destinata allo sviluppo di applicazioni commerciali.

Anche se non riusciamo a garantirci un sufficiente volume di introiti con prodotti complementari o versioning, se abbiamo una buona base di utenti, possiamo comunque vendere l'accesso ad essa ad altri.

Principalmente questo può avvenire per scopi pubblicitari: è noto il caso della MandrakeSoft, che ha introdotto pubblicità nella fase di installazione della propria distribuzione per gli utenti non iscritti.

Un aspetto da non sottovalutare è la possibilità di vendere anche su supporto fisico i propri prodotti ad un prezzo superiore al solo costo di riproduzione.

Gli utenti potrebbero comunque essere invogliati a comprare la versione offline se si presta la giusta attenzione a come aggiungere valore a tale versione.

Netscape ha avuto fortuna con questo metodo perchè al tempo scaricare 5 MB di browser per molti era una problema.

Richard Stallman nei primi anni di vita della GNU, è riuscito a portare avanti il progetto soprattutto grazie alla vendita dei dischetti con Emacs, il celebre editor di testi e IDE di sviluppo, anche se questo era disponibile per il download anche sul suo stesso sito.

L'altra grande fonte di entrate, che ha salvato più di un progetto Open Source, sono le libere donazioni da parte di utenti, istituzioni o imprese. Può sembrare strano ma le caratteristiche delle licenze libere rendono il prodotto patrimonio dell'umanità.

Per questo motivo navigando su un qualsiasi sito di un progetto Open Source, anche di notevoli dimensioni e senza problemi di introiti, è quasi sempre possibile trovare comodi link su come effettuare trasferimenti di denaro on-line sul loro conto bancario.

Infine per la maggior parte dei progetti, e in particolare quelli su commissione del cliente, è sempre possibile guadagnare sulle eventuali modifiche e/o miglioramenti del prodotto di cui l'utente avrà inevitabilmente bisogno.

Anche personalizzazioni dello stesso possono essere effettuate dietro compenso, perchè se si mantengono prezzi sufficientemente bassi, il cliente non avrà motivo di rivolgersi ad altri, che dovrebbero infatti proporre dei prezzi maggiori, visti i costi che dovrebbero sostenere per la comprensione del nostro codice.

LINUX VERSUS MICROSOFT

Una guerra in corso



Portiamo in questa sede l'esempio di una guerra in corso tra una tecnologia proprietaria e una Open Source per il dominio del settore dei sistemi operativi, sia lato server e workstation che lato client. Il Linux Kernel Project è il progetto Open Source per eccellenza, sicuramente il più conosciuto, e probabilmente anche quello che è riuscito ad applicare al meglio lo schema di sviluppo a bazaar, essendone l'inventore. In questo caso ha il ruolo dell'innovatore, visto che la sua tecnologia non è compatibile con quella attualmente più diffusa, il famoso Windows.

Dall'altra parte c'è il colosso dell'informatica per antonomasia, la Microsoft, che possiede un invidiabile monopolio sul settore da anni ormai, ed è abilissima a sfruttarlo, utilizzando spesso geniali strategie di marketing e politiche volte ad estendere la sua influenza in un po' tutti i campi dell'informazione.

I sistemi operativi sono sicuramente un settore soggetto a fortissime esternalità di rete ed economie di scala dal lato della domanda; lo sa bene l'Apple che da alcuni anni ormai si è vista ridurre considerevolmente la quota di mercato per essere, proprio quest'anno, superata da Linux. La presenza poi di esternalità di rete favorisce anche la nascita di forti costi di transizione collettivi, mettendo gli eventuali concorrenti dell'impresa monopolista, in una posizione ancora più sfavorevole.

Inoltre i lock-in esistenti sono elevatissimi, soprattutto in termini di software e wetware. Molti sostengono che il problema principale sia infatti la maggiore facilità d'uso di Windows rispetto a Linux, ma, grazie ai progressi compiuti negli ultimi tempi dalle interfacce grafiche (in particolare pensiamo a KDE e Gnome), si tratta più che altro di un lock-in di esperienza: è stato dimostrato da più studi che il tempo che persone senza competenze nel campo hanno impiegato per imparare ad utilizzare i diversi sistemi è più o meno lo stesso; è più problematico passare a Linux per persone che hanno già imparato Windows. Per quanto riguarda il software, nonostante la presenza di emulatori come WINE (Windows Emulator), i programmi scritti per Windows generalmente non girano sotto Linux, ed è innegabile che in alcuni settori non esistano adeguati applicativi per quest'ultimo, essendo tutto il mercato rivolto alla piattaforma della Microsoft. Fino a qualche anno fa inoltre c'erano considerevoli problemi anche di compatibilità hardware: siccome le industrie dell'elettronica non rilasciavano i drivers per Linux, questi dovevano essere "fatti in casa" dagli utenti, quindi non di rado capitava di comprare accessori che poi non si riusciva ad utilizzare. Ora il fenomeno è decisamente più ridotto, perchè la recente impennata della popolarità del pinguino ha costretto un po' tutti a rivolgergli la propria attenzione.

Quindi com'è possibile pensare di poter scalzare un colosso come la Microsoft in un settore così soggetto a lock-in così abilmente sfruttati dall'impresa leader?

Per cominciare possiamo considerare la strategia di penetrazione di Linux come molto aggressiva, dal momento che il prezzo del sistema operativo è a tutti gli effetti pari a zero, essendo molte Distribuzioni gratuite, senza considerare tutti i vantaggi per gli utenti che derivano dal fatto di essere Libero. Sono innegabili inoltre le migliori prestazioni del sistema soprattutto in termini di affidabilità e stabilità, il che gli fornisce le caratteristiche perfette per essere adottato in ambiente server. Infatti in questo segmento, in cui le economie di scala dal lato della domanda sono molto più ridotte, Linux non ha avuto problemi ad imporsi: il suo tasso di crescita annua si aggira intorno al 100% e la sua quota di mercato si sta pericolosamente avvicinando a quella dell'impresa di Redmond.

Per quanto riguarda la gestione delle aspettative, entrambi i contendenti sono molto abili nel saperle sfruttare, la Microsoft usando campagne promozionali pubblicitarie molto insistenti, e

affidandosi alla potenza del suo marchio; Linux grazie all'affiatatissima comunità che lo sostiene più che attivamente, senza contare l'abilità di Torvalds nel diffondere negli ambienti a lui vicini l'idea di un'inevitabile diffusione del suo sistema operativo come dominante sugli altri: "Linux Total World Domination". Il pinguino può sfruttare anche una numerosa serie di riviste che lo supportano e lo distribuiscono e la fama di stabilità e di crescita continua che gli deriva dal segmento server. Un altro vantaggio di non poco conto che ha Linux è la sua maggiore dinamicità: molto di frequente escono nuove versioni del Kernel, dei Desktop Managers, di tutti i vari software, grazie al processo di sviluppo aperto a bazaar, dimostrando una maggiore capacità innovativa, e ciò contribuisce non di poco a trasmettere sicurezza ai suoi utenti che, vedendolo crescere di continuo, non temono che da un momento all'altro venga spiazzato e superato.

Da questo punto di vista bisogna notare che la Microsoft sta facendo proprio ora alcuni passi falsi: sono passati ormai 3 anni dall'ultima release di un nuovo sistema operativo per i desktop, correva l'anno 2001 e si parlava di Windows XP. Il suo prodotto di punta è vecchio, quello che dovrebbe diventare il suo successore, Longhorn, è indietro nello sviluppo e sicuramente prima di un paio d'anni non sarà rilasciato. Per non perdere il controllo sui suoi utenti, che proprio in questo frangente potrebbero decidere di passare alla concorrenza, la casa di Redmond sta vagliando la decisione di rilasciare prima di Longhorn un'altra versione di XP: vedremo quanto successo avrà questa strategia.

Anche la reputazione dei due contendenti sta giocando una parte importante: Microsoft ha certamente la fama di essere un'impresa vincente, ma di certo ha una pessima reputazione in fatto di apertura e prestazioni dei suoi prodotti; anche chi non conosce affatto Windows avrà di certo sentito qualcuno lamentarsi dei suoi bug. Anche per quanto riguarda la compatibilità dei suoi prodotti le critiche non sono rare: tutti pensano che il software di Microsoft funzioni solo con il software di Microsoft, e nella maggior parte dei casi ciò è vero. Tutto questo al momento giusto di certo non giocherà a suo favore.

Linux d'altra parte, essendo Libero, ha un'ottima fama presso una gran parte della comunità di sviluppatori, anche in termini di prestazioni, talvolta superiore alla realtà delle cose. Per contro, questa fama lo ha a lungo caratterizzato come sistema con una elevata complessità di utilizzo e per questo adatto solo ad utenti con una formazione molto approfondita; questo oggi non è più così vero, dato l'ormai ottimo livello di intuitività degli ambienti grafici.

Nel delicato gioco politico per la conquista del mercato, bisogna sempre analizzare anche il sistema di alleanze. Microsoft è sicuramente molto potente, e per molto tempo ha potuto contare sull'appoggio dei produttori di hardware e dei distributori di personal computer. I primi oggi, a differenza di un tempo, rilasciano i drivers anche per Linux (a nostro parere anche per evitare di rimanere loro stessi intrappolati, nell'eventualità di una crescita inaspettata del nuovo sistema operativo, a discapito dell'attuale monopolista) e, anche se la maggior parte dei sistemi desktop viene ancora venduta con Windows Preinstallato, si cominciano a vedere in giro offerte di pc con Linux o con entrambi i sistemi operativi.



Insomma, oggi la Microsoft non può più godere dell'appoggio incondizionato di tutti i produttori hardware e software, come in molti casi accadeva fino a ieri, forse anche perchè non tutti vedono come negativo un suo indebolimento, data la reputazione e l'aggressività con cui ha schiacciato concorrenti come Netscape in passato. Per contro, proprio per indebolire lo scomodo avversario, molte sono le imprese che hanno massicciamente investito nel pinguino, anche per il fatto che, come chiarito precedentemente, essendo Libero, è il fornitore ideale per chiunque. Fra gli altri ricordiamo Silicon Graphics per le workstation, IBM, uno dei

sostenitori più attivi, HP e Motorola, quest'ultimo per il mercato dei palmari e cellulari. Stanno avendo un ruolo chiave, e probabilmente lo avranno sempre di più nel corso della guerra, le pubbliche amministrazioni e le politiche governative; che a poco a poco iniziano a schierarsi apertamente dal lato di Linux. Dato che il software Libero è patrimonio dell'umanità ciò non dovrebbe affatto sorprendere.

Dall'anno scorso si sono iniziate a leggere, con una frequenza quasi mensile, notizie di passaggio del sistema informatico delle pubbliche amministrazioni all'opzione Open Source: è famoso il caso di Monaco.

Ancor di più il controllo dei mercati emergenti sta diventando di importanza strategica: Cina e India, che si stanno informatizzando in questi anni, per il timore di subire in un futuro non troppo lontano forti ingerenze da parte di Microsoft, attratte dal basso prezzo dell'alternativa, hanno scelto Linux come loro piattaforma di sviluppo. La Cina in particolare ha deciso di influenzare deliberatamente il mercato con apposite politiche governative, volte a tagliare fuori la Microsoft, in favore di Distribuzioni locali.

Anche altri mercati emergenti come quelli dei cellulari e dei pda, controllati da case produttrici di hardware del calibro di Nokia, Motorola e Sony\Ericsson, stanno sfuggendo dalle mani di Bill Gates: tali imprese non hanno nessuna intenzione di cedergli il controllo del settore. Per questo preferiscono optare ancora una volta per Linux, per sistemi aperti e multipiattaforma come java, o per altri sistemi operativi standardizzati (Symbian OS).

Tali mercati in crescita possono aiutare moltissimo il pinguino a raggiungere la massa critica e innescare così il feedback positivo.

Un segnale forte del fatto che la strategia sta funzionando, è che ultimamente un po' tutte le software house stanno prendendo in considerazione il rilascio di versioni dei propri prodotti per Linux, in particolare in ambito server la situazione è capovolta: si sviluppa principalmente per il pinguino e si cerca successivamente un'eventuale compatibilità con Windows

Chi vincerà al momento è impossibile dirlo, ma una considerazione si può sicuramente fare: Linux senza una strategia Open Source non sarebbe mai riuscito ad arrivare dove è arrivato, e sono incredibili i passi che è riuscito a fare dato il totale controllo del mercato da parte di Microsoft.

Per sottolineare quest'ultimo fatto portiamo l'esempio di alcuni altri sistemi operativi che hanno tentato di rompere il monopolio della casa di Redmond:

- 1) OS/2 di IBM
- 2) BeOS

nessuno dei due è mai riuscito ad arrivare ad una quota di mercato vicina all'1%, mentre Linux ora ne possiede quasi il 5%.



JAVA

Una decisione strategica da prendere

Come secondo esempio di quanto sia importante strategicamente optare per l'Open Source, abbiamo deciso di portare l'esempio di Java, di cui si è molto parlato ultimamente.

Java è un linguaggio di programmazione che permette la creazione di software multiplatforma, tramite l'utilizzo di una virtual machine, fornita insieme a un corposo sistema di librerie. Essere multiplatforma lo rende molto importante, non solo per gli evidenti vantaggi legati a questo fatto, ma anche perchè può ridurre moltissimo i lock-in degli utenti nei confronti del sistema operativo, come per esempio Windows. Non c'è da meravigliarsi che la Microsoft si sia adoperata attivamente per togliere a Java questo vantaggio, rilasciando la propria versione dell'ambiente di sviluppo Java con un package di librerie che impedivano qualsiasi programma creato con esso di girare in ambiente non Microsoft. La cosa si è poi risolta in sede legale: Microsoft ha alla fine perso la battaglia giudiziaria legata a questo fatto in favore della Sun. Quest'ultima ha sempre cercato di promuovere la sua tecnologia tramite una strategia aperta e una larga alleanza con gruppi di imprese molto potenti, come Nokia e IBM; promettendo sin dal principio di mantenerla il più aperta possibile. Ma è proprio quest'ultimo punto l'argomento di discussione. Perchè se è vero che sia la virtual machine che il compilatore e le librerie necessarie per scrivere software in Java sono sempre state gratuitamente disponibili sul sito della Sun, quindi a costo zero, non sono Liberi. Anche se l'impresa californiana gode tutto sommato di una buona reputazione negli ambienti Open Source perchè supporta attivamente Linux e altri progetti a sorgente aperto di sua iniziativa, essa potrebbe da un momento all'altro rendere l'SDK a pagamento, creando enormi problemi a tutte le altre imprese che hanno puntato sulla sua tecnologia per i propri prodotti. Inoltre fino a poco tempo fa Java era aperto solo di nome, perchè in realtà non c'era nessuna specifica formale sull'interfaccia e le API. Spinta da forti pressioni tanto dai partner che dai suoi stessi utenti la Sun ha deciso di standardizzare Java e le sue interfacce in modo tale che chiunque possa implementare una virtual machine e un ambiente di sviluppo compatibili, ma non ha reso ancora Open Source i suoi. Questa è una politica sufficientemente aperta per molti ma non per tutti, e rende freddi alcuni possibili importanti clienti, anche considerando il fatto che vere e proprie alternative all'SDK di Sun al momento non ci sono perchè l'implementazione Libera GNU di Java, chiamata Kaffe, è ancora in fase iniziale.

Inoltre la minaccia della piattaforma .NET di Microsoft si fa sempre più concreta e c'è la reale possibilità che prenda il sopravvento conoscendo le potenzialità della casa di Redmond in fatto di marketing e di abilità nello sfruttare il monopolio sui sistemi operativi.

Eric Raymond stesso recentemente ha scritto una lettera alla Sun per chiedere di aprire il codice della sua tecnologia, a cui sono seguiti una risposta non risolutiva, altre richieste similari da parte di IBM, e una lunga serie di commenti della comunità sulla questione.

Noi riteniamo che la politica della Sun non sia abbastanza chiara e sembra un po' nella posizione di chi vuole tenere il piede in due scarpe. I suoi timori di un eventuale scissione di un progetto nato da una costola di Java, o di perdita dello sponsor sono probabilmente infondati; basterebbe utilizzare una licenza forte come la GPL e continuarne attivamente lo sviluppo e il supporto.

Cercare invece di lucrare sulla standardizzazione della tecnologia può diventare particolarmente rischioso se non si prendono decisioni forti in merito.

Questa vicenda dimostra come ormai considerare l'opzione di rendere Open Source la propria tecnologia sia fondamentale, e che molte grandi potenze del settore dell'informazione ne hanno appieno compreso il valore.

Una strategia semplicemente aperta ma non Open Source spesso non basta e non fare la scelta verso un sistema Libero quando se ne ha la possibilità può diventare il fattore determinante per la propria crisi. Vedremo se Sun farà in tempo la scelta che noi riteniamo giusta, se invece anche la sua politica avrà successo oppure se Java cadrà in favore di .NET.

CONCLUSIONI

La domanda che ci siamo posti inizialmente è: quando è conveniente adottare come modello di sviluppo e modello economico l'Open Source, o eventualmente passare ad esso se si ha già un'attività avviata?

Non vi è chiaramente una risposta universale a questa domanda, ma proveremo ad identificare una serie di casi tipo a cui poter far riferimento.

Se vogliamo iniziare una nuova attività, e la tecnologia che domina il settore in cui ci vogliamo andare ad inserire è di tipo proprietario, allora è probabile che la strategia vincente sia quella di puntare sull'Open Source. Questo ci garantirà subito una base di utenti maggiore e una maggiore visibilità, e spesso può essere l'unico modo per convincerli a cambiare, visti i notevoli lock-in che in questi settori sono sempre presenti. E' il caso del nostro esempio di Linux che abbiamo sopra esposto.

Nel caso in cui invece vogliamo andarci ad inserire in un settore in cui domina un progetto Open Source, allora la scelta Libera diventa veramente l'unica possibile, perchè convincere gli utenti a passare a noi, senza garantirgli gli enormi privilegi che essi stessi manterrebbero col loro vecchio fornitore è praticamente impossibile. Va anche detto che provare a scalzare il dominio di un progetto Open Source va fatto solamente se si possiede davvero una tecnologia migliore, perchè solamente in questo caso si avrebbe un vero vantaggio sul concorrente.

Se invece abbiamo un progetto già avviato, conviene comunque chiederci se ha senso passare al Software Libero o rimanere commerciali. Il cambiamento può infatti favorire l'aumento dei nostri utenti a discapito dei nostri concorrenti; se questi ultimi sono anch'essi di tipo commerciale o proprietario, la strategia può essere vincente. Se siamo noi ad avere la più larga quota di mercato, conviene aspettare le mosse degli avversari: la strategia commerciale può garantirci profitti superiori al momento, ma per non essere sorpassati da qualche nuovo entrante più aperto, è importante stare sempre all'erta, ed eventualmente essere pronti a diventare Open Source e ristrutturare la nostra organizzazione per non essere spiazzati.

Tra aprire il codice di tutti i nostri progetti e tenerli strettamente closed, naturalmente c'è un'ampia gamma di possibilità da considerare; come già accennato prima, si può sfruttare il versioning del prodotto, rendere Open Source la versione base e mantenere più chiusa quella avanzata. Bisogna però stare attenti a questo tipo di mosse strategiche per non violare le licenze utilizzate in quanto

mischiare codice aperto con codice commerciale può essere particolarmente problematico. Concludendo si può affermare che saper sapientemente scegliere una strategia Open Source è fondamentale per il successo della nostra impresa: se, come nella maggior parte dei mercati, è possibile aggiungere valore al nostro software tramite la fornitura di servizi o in tutti gli altri modi che abbiamo illustrato prima, è senz'altro da prendere in considerazione l'adesione al modello Open Source; se, poi, si tenta la creazione di un nuovo progetto, o l'ingresso in un nuovo mercato, è decisamente più conveniente puntare sul software libero. Il problema semmai è riuscire ad applicare al meglio il modello a bazaar sapendosi costruire una larga, fedele e affiatata comunità di utenti. Scegliere le modalità con cui ricavare profitti dai nostri clienti è altrettanto importante e spesso notevolmente difficoltoso. Naturalmente le strategie da utilizzare sono da studiare sin dall'inizio, per sapersi muovere correttamente e non lasciarsi sorpassare dagli eventi.

BIBLIOGRAFIA

Eric Steven Raymond – The Cathedral and the Bazaar

Josh Lerner (Harvard Business School), Jean Tirole (Institut D'Economie Industrielle) - The simple Economics of Open Source

Philp E. Varner (Virginia University) – The Economics of Open Source Software

Barbara Scozzi (Politecnico di Bari) - Le comunità di sviluppo di Open Source Software: analisi delle pratiche di coordinamento

Carl Shapiro, Hal R. Varian (University of Berkeley) – Information Rules

Asif Khalak (MIT, Massachusset Institute of Tecnology) - Economic model for impact of open source software

ELISS (European Libre Software Survey) - Un analisi del fenomeno Open Source (OSS) in Italia

Gli archivi della lista FLUG (Ferrara Linux User Group) - <http://liste.ferrara.linux.it/archive/flug/>

<http://foundation.gnome.org/>

<http://www.kde.org/support/thanks.php>

Vario materiale disponibile in rete.

LICENZE E DIRITTI D'AUTORE:

Questo documento è disponibile sotto licenza FDL (<http://www.gnu.org/licenses/fdl.html>)
Tutti i loghi e i marchi presenti sono di proprietà dei legittimi proprietari.