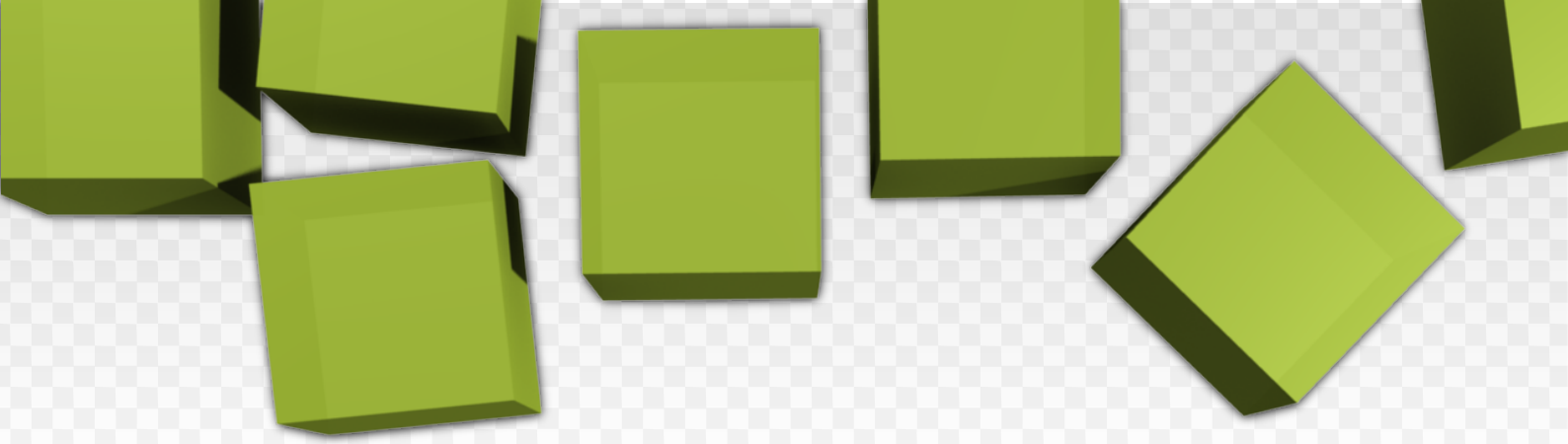


Aspetti economici e modelli di business del software libero

Autori:
A. Albos Raya
L. Bru Martinez
I. Fernandez Monsalve

Coordinatori:
D. Megias Jimenez

Free Technology Academy

5

Aspetti economici e modelli di business del software libero

David Megías Jiménez (coordinador)
Amadeu Albós Raya
Lluís Bru Martínez
Irene Fernández Monsalve

PID_00145008

UOC
Universitat Oberta de Catalunya
www.uoc.edu

David Megías Jiménez

Laureato ingegnere informatico all'Universidad Autónoma de Barcelona (UAB). Master in Tecniche avanzate per l'automatizzazione presso la UAB. Dottorato in informatica presso la UAB. Professore per l'area degli Studi sull'informatica e sui Multimedia della UOC.

Amadeu Albós Raya

Laureato ingegnere in informatica nell'Universitat Oberta de Catalunya in collaborazione con l'Università di Andorra, con specializzazione in sicurezza, reti e sistemi di distribuzione. Attualmente è professore, coordinatore ed amministratore dei sistemi informatici in un centro di studi, è membro del Grup de Recerca en Seguretat Informàtica i Programari Lliure dell'Università di Andorra, ed è consulente per il master universitario sul Sotware libero che impartisce l'Universitat Oberta de Catalunya.

Lluís Bru Martínez

Irene Fernández Monsalve

Laureata in Scienze Ambientali presso l'Universidad Autónoma de Madrid. Master internazionale in Software libero presso l'Universitat Oberta de Catalunya. E' socio fondatore di Haicku, S. Coop. Mad., impresa per la quale si occupa di gestione ed implementazione del software libero nei Centros de Acceso Público a Internet della Sierra Norte de Madrid.

Prima edizione: settembre 2009

Av. Tibidabo, 39-43, 08035 Barcelona
Impaginazione: Manel Andreu
Realizzazione editoriale: Eureca Media, SL
ISBN: 978-84-691-8694-7
Codice di registrazione: B-2.239-2009

Prefazione

Negli ultimi dieci anni il software è diventato una risorsa strategica. L'arrivo del software libero, che è entrato in scena nei principali settori del mercato ICT, sta cambiando radicalmente l'economia dello sviluppo e delle applicazioni del software.

Il sostare libero, chiamato anche "Open Source" or "Libre Software" – può essere usato, studiato, modificato e distribuito liberamente. Offre la libertà di imparare e insegnare senza legarsi a un solo fornitore. Queste libertà sono considerate una pre-condizione per lo sviluppo sostenibile e per la crescita di una società dell'informazione inclusiva.
Anche se c'è grande interesse verso il software libero (Free Software e Open Standards), solo poche persone hanno competenze valide in questo campo. FTA cerca di rispondere a questo bisogno

Che cosa è FTA?
La Free Technology Academy (FTA) è un'iniziativa comune di diverse istituzioni educative di vari paesi. Il suo obbiettivo è di contribuire a una società che permette a tutti gli utenti di studiare, partecipare e costruire conoscenza senza restrizioni.

Che cosa offre FTA?
FTA offre un master online con corsi sul software libero. Gli studenti scelgono di iscriversi a corsi individuali o a tutto il programma. I corsi vengono svolti tramite la piattaforma online di FTA, e sono condotti dal personale docente delle università partner. I crediti ottenuti tramite FTA sono riconosciuti da queste università.

Chi c'è dietro FTA?
FTA è iniziata nel 2008 sostenuta dal programma Life Long Learning (LLP) della commissione europea, con il coordinamento del Free Knowledge Institute e la partnership di tre università europee: Open Universiteit Nederland (Olanda), Universitat Oberta de Catalunya (Spagna) e University of Agder (Norvegia).

A chi si rivolge FTA?
FTA si rivolge in particolare a professionisti IT, formatori, studenti e decisori.

E la licenza?
Tutti i materiali usati e sviluppati da FTA sono Open Educational Resources, pubblicati sotto licenza copyleft, che permette di usarli liberamente, modificarli e ridistribuirli. Analogamente, il software usato nel campus virtuale di FTA è libero, e sviluppato su framework Open Standards.

Evoluzione di questo testo

FTA ha riusato materiale didattico già esistente della Universitat Oberta de Catalunya, che era stato sviluppato in collaborazione con LibreSoft staff della Universidad Rey Juan Carlos. Nel 2008 questo libro è stato tradotto in inglese con l'aiuto del progetto SELF (Science, Education and Learning in Freedom), sostenuto dal Sesto Programma Quadro della Commissione Europea. Nel 2009, questo materiale è stato rivisto e integrato dalla Free Technology Academy. Inoltre, FTA ha sviluppato una guida di studio e delle attività di apprendimento disponibili nel campus FTA.

Partecipazione

Incoraggiamo tutti i lettori di questo testo a fornire feedback e suggerimenti di miglioramento. Uno spazio apposito è stato creato sul sito di FTA. Tutti i suggerimenti saranno presi in considerazione per le prossime versioni. Inoltre, FTA dà il benvenuto a tutti colore che vorranno distribuire questi materiali, o farne nuove versioni o traduzioni. Informazioni aggiornate e dettagliate sui materiali sono disponibili all'indirizzo http://ftacademy.org/materials/fsm/.
Per ulteriori informazioni e per iscriversi ai corsi online di FTA, potete visitare il sito di FTA: http://ftacademy.org/.

Spero che questo testo vi aiuti nel vostro percorso di sviluppo personale, e che vi permetta ad aiutare altri.

Buon apprendimento!

Wouter Tebbens
President of the Free Knowledge Institute
Director of the Free technology Academy

Introduzione

Nella società odierna, le industrie della tecnologia dell'informazione e della tecnologia della conoscenza (TIC), ovvero le telecomunicazioni, l'elettronica (di produzione e di consumo) e l'informatica, hanno ottenuto un'importanza sconosciuta solo poco tempo fa. In esse si sono succedute innovazioni ad un ritmo strepitoso, accompagnate da un processo di convergenza delle stesse, attraverso la manipolazione, la trasmissione e la riproduzione dell'informazione con processi in comune: le tecnologie digitali.

Ad ogni modo, ancor più dei cambiamenti prodotti nelle industrie direttamente legate alle TIC, acquisiscono estrema importanza i cambiamenti e le migliorie che hanno influito su altre attività economiche, soprattutto le più tradizionali, all'interno delle quali il modo di procedere ha sperimentato una ristrutturazione profonda. Le conseguenze sul tessuto economico hanno propiziato l'apertura di nuove aree di business come risposta a nuove necessità del mercato, inedite tempo addietro.

Ora, non è cambiata nessuna legge economica e nessuno dei fenomeni economici legati alle TIC è qualitativamente nuovo, quello che sì è cambiato è l'importanza relativa di determinati effetti economici nella nostra società, come ad esempio le politiche di proprietà intellettuale o di compatibilità dei prodotti. Arriviamo a una conclusione simile anche se ci focalizziamo più concretamente sugli effetti economici rilevanti nell'industria del software.

In questo senso, l'industria del software ha visto negli ultimi tempi il lancio e la diffusione al pubblico in generale del software libero. Il software libero si è posizionato in diversi settori del mercato del software generico come un'alternativa valida e praticabile, e si è indiscutibilmente inserito anche in alcuni settori specializzati (ad esempio, il software dei server). Questa situazione ha inciso sull'industria del software basata sul modello della proprietà, e ha provocato la comparsa di nuovi modelli di business legati alla diffusione, allo sviluppo, al supporto e all'affermazione del software libero.

Questo corso *Aspetti economici e modelli di business del software libero*si propone l'obiettivo di fornire le conoscenze necessarie a comprendere e a mettere in pratica l'economia del software libero mediante lo studio e l'analisi degli aspetti economici e dei modelli di business ad esso legati, così come mediante lo studio e l'analisi delle opportunità che offre questo nuovo mercato.

Nel primo modulo del corso vengono descritti i concetti economici necessari allo studio della struttura economica dell'industria del software in generale, e necessari in particolare per strutturare un modello di business a partire dal software libero.

Nel secondo modulo, il corso descrive brevemente le caratteristiche principali del mercato del software con riferimento ai modelli di business consolidati e osserva quali sono i potenziali clienti dei progetti imprenditoriali basati sul software libero.

Nel terzo modulo il software viene presentato dal punto di vista del business, con l'attenzione incentrata sul mercato nel quale competono le diverse soluzioni, e sulla strategia imprenditoriale da usare per attrarre i potenziali clienti.

Nel quarto modulo vengono studiati i modelli di business legati al software libero. Si comincia analizzando le classificazioni proposte da alcuni autori di riferimento, e si passa poi a definire un metodo di classificazione proprio del corso, tenendo in considerazione le caratteristiche economiche più rilevanti del business.

Nel quinto modulo si analizza lo sviluppo del software libero dal punto di vista imprenditoriale. Si considerano qui tanto le caratteristiche più rilevanti del progetto di software quanto le peculiarità legate alla comunità degli utenti del software libero e la loro gestione funzionale e legale.

Nel sesto modulo vengono presentati i principali vantaggi e gli inconvenienti riconducibili al modello del software libero come strategia imprenditoriale, considerando tanto la prospettiva del cliente quanto quella della stessa impresa ed il modello di business che sfrutta.

Nel settimo ed ultimo modulo, il corso si propone di studiare il software libero come modello economico, concentrando la propria attenzione tanto sulle fondamenta della sua esistenza quanto sulle implicazioni attuali e future relative al business basato sul software libero.

Obiettivi

Al termine del corso, lo studente dovrà essere in grado di:

1. Identificare e comprendere i fattori economici più rilevanti legati al settore tecnologico, ed, in particolare, al settore del software libero.

2. Capire i diversi modelli di business sfruttabili nell'industria del software e le loro principali caratteristiche.

3. Saper approfondire i diversi modelli di business vincolati al software libero e le sue opportunità di business.

4. Comprendere e mettere in relazione i fattori economici, tecnici e legali necessari a creare e a far funzionare un'impresa basata sul software libero.

5. Analizzare casi di sfruttamento del software libero nel settore privato e la loro relazione con il mercato della tecnologia.

6. Saper approfondire le strategie di sviluppo e di sfruttamento del software libero a fini di lucro.

7. Pianificare ed ideare business efficaci e realizzabili basati sullo sfruttamento del software libero.

Contenuti

1. Le basi del modello
2. Le caratteristiche del modello del software libero
3. L'efficacia e la validità del modello di software libero

Risorse

Associazioni di imprese del software libre:

- Asociación Catalana de Empresas por el Software Libre: (www.catpl.org).
- Asociación de Empresas de Software Libre de Canarias (www.eslic.info).
- Asociación de Empresas de Software Libre de Euskadi (www.esle-elkartea.org).
- Asociación de Empresas Gallegas de Software Libre (www.agasol.org).
- Asociación Madrileña de Empresas de Software Libre (www.solimadrid.org).
- Open Source Business Organisations of Europe (www.obooe.eu).

Dare vita ad un business:

- Centro Europeo de Empresas e Innovación del Principado de Asturias: Guía para la creación de empresas y elaboración del Plan de Empresa (http://www.guia.ceei.es).

Conferenze (software libero ed impresa):

- Open Source Business Conference (OSBC) (http://www.osbc.com/live/13/events/13SFO07A/SN441958/CC141932/ y http://www.osbc.com).
- WhyFLOSS (http://www.whyfloss.com/es/conference).

Dati delle imprese:

- Camera di Commercio (http://www.camerdata.es).
- Hoovers (http://www.hoovers.com).

Casi di studio:

- Avanzada7: "Business models based on Asterisk: The case of Avanzada7" (http://www.whyfloss.com/es/conference/madrid08/getpdf/64).

- IBM migration to Free Desktops (http://www.channelregister.co.uk/2007/02/13/ibm_open_client).

- Liferay: "Liferay Enterprise Portal: The project, the product, the community and how to extend it" (http://www.whyfloss.com/es/conference/madrid08/getpdf/66).

- Openbravo: "Openbravo: keys to success in free software application development" (http://www.whyfloss.com/es/conference/madrid08/getpdf/49).

- Red Hat and JBoss: "Is Open Source viable in Industry? The case of Red Hat and JBoss" (http://www.whyfloss.com/es/conference/madrid08/getpdf/68).

- Casi vari (www.opensourceacademy.gov.uk/solutions/casestudies).

Blog sul software libero, l'innovazione e l'impresa

Galopini, R. *Commercial open source software* <http://robertogaloppini.net/> [Consulta: marzo 2009]

Oswalder, A. *Business model design and innovation blog.* <http://business-model-design.blogspot.com/> [Consulta: marzo 2009]

The 451 group. *451 caos theory: A blog for the enterprise open source community.* <http://blogs.the451group.com/opensource/> [Consulta: marzo 2009]

Bibliografia

Sieber, S. (2006). *Factores que influyen en la adopción del código abierto*. IESE <www.iese.edu/es/files/Art_Computing_Sieber_Factorescódigoabierto_May06_tcm5-5510.pdf> [Consultazione: marzo 2009]

Sieber, S.; Valor, J. (2005). *Criterios de adopción de las tecnologías de información y comunicación*. IESE <www.iese.edu/en/files/6_15211.pdf> [Consultazione: marzo 2009]

Dixon, J. (2007). "The Beekeeper: Crossing the Chasm Between the Cathedral and the Bazaar". *A Description of Professional Open Source Business Models*. <http://wiki.pentaho.com/pages/viewpageattachments.action?pageId=8346> [Consultazione: marzo 2009]

Woods, D.; Guliani, G. (2005). *Open Source for the Enterprise: Managing Risks, Reaping Rewards O'Reilly Media*. <http://books.google.com/books?id=f4qJiKytvgC&dq=open+source+for+the+enterprise>

Nozioni di base di economia

Lluís Bru Martínez

PID_00145050

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

In questo primo modulo, presenteremo i concetti principali collegati con l'economia dei prodotti ed, in particolare, le peculiarità del business delle tecnologie dell'informazione e della comunicazione. L'obiettivo dell'introduzione di questi concetti è quello di fornire una base attraverso la quale comprendere i diversi comportamenti ed i diversi modelli di business adottati dalle imprese e che verranno dettagliati più avanti.

Nella prima parte parleremo delle nozioni di base sul valore del prodotto a partire dalla domanda e dall'offerta, oltre che del vantaggio competitivo rispetto alla concorrenza come strumenti fondamentali per la sostenibilità del business.

Nella seconda parte illustreremo i principali effetti economici legati alle caratteristiche dei prodotti tecnologici e del software in generale. Vedremo in dettaglio come l'impresa può agire sul fronte del mercato attraverso una politica di manipolazione di questi effetti volta alla creazione di uno scenario che le sia il più favorevole possibile rispetto ai suoi concorrenti.

Obiettivi

Al termine di questo modulo, lo studente dovrà essere in grado di:

1. Riflettere sui meccanismi di funzionamento della domanda e dell'offerta, specie rispetto ai concetti relativi alla creazione di valore.

2. Identificare ed esaminare le principali caratteristiche economiche dell'industria del software.

3. Approfondire e mettere in relazione tra loro gli effetti economici relativi al mercato del software.

4. Identificare ed analizzare gli effetti economici suscettibili di trasferire del valore o un vantaggio competitivo ai prodotti sviluppati a partire dal software libero.

5. 1) Approfondire le politiche e le strategie di gestione del mercato del software libero.

1. La creazione del valore

Perché un determinato business sia sostenibile è necessario che vi siano persone o imprese disposte, in qualità di clienti, a pagare per il prodotto o il servizio che viene loro offerto, e che questi pagamenti compensino ai fornitori le spese che implica fornire il prodotto o il servizio. La prima cosa che vedremo in maniera molto semplice saranno i concetti economici fondamentali che scaturiscono dall'interazione tra l'impresa che gestisce il business e i suoi eventuali clienti.

1.1. La domanda di un prodotto

Per cominciare, dobbiamo descrivere alcune possibili regole di comportamento delle imprese e delle famiglie, entrambe obiettivi del nostro business.

Un consumatore (se si tratta di un bene di consumo) o un'impresa (se si tratta di acquisire macchine, materie prime, ecc.) decideranno di comprare un determinato prodotto o un determinato servizio se la quantità di denaro (il prezzo) che si chiede loro in cambio sembra loro ragionevole.

In questa situazione, il possibile compratore sta facendo il seguente ragionamento:

1) Innanzitutto, gli sembra ragionevole pagare al massimo una quantità V di denaro per avere in cambio il prodotto o il servizio offerto. Pertanto, se gli viene richiesta una quantità di denaro P inferiore a V, il compratore ritiene che valga la pena acquistare il prodotto. Quindi, perché una persona decida di diventare nostro cliente, è necessario che si verifichi

Assegnazione di valore al prodotto ? prezzo del prodotto = V- P > 0

In altre parole, un'impresa non riuscirà a farsi pagare più di V per il suo prodotto, e tuttavia il richiedere meno di V non implica automaticamente che il cliente compri il prodotto.

2) In secondo luogo, il cliente confronterà la nostra proposta con le altre alternative disponibili. Tra due o più prodotti simili, il consumatore sceglierà quello nel quale la differenza V ? P sia maggiore.

Esempio pratico

Una famiglia decide di comprare un'auto. La famiglia ritiene che il modello della marca A valga 40.000 euro, Va = 40.000 euro, ed il suo prezzo di vendita è 30.000 euro, Pa = 30.000. La famiglia ritiene che il modello della marca B valga meno; supponiamo che ritenga che valga, a causa delle sue prestazioni inferiori (ad esempio, perché è un veicolo più piccolo), 35.000 euro, Vb = 35.000 euro.

La famiglia dell'esempio comprerà il modello della marca A anche se è più caro, sempre che l'auto della marca B venga venduta per più di 25.000 euro; viceversa, comprerà il modello di B se è abbastanza economico, cioè, se il suo prezzo è inferiore a 25.000 euro.

Ne deriva che:

Compra il prodotto A solo se

$$Va ? Pa = 40.000 ? 30.000 > Vb ? Pb = 35.000- Pb,$$

cioè, solo se Pb > 25.000.

Compra il prodotto B solo se

$$Va ? Pa = 40.000 ? 30.000 < Vb ? Pb = 35.000 ? Pb,$$

cioè, solo se Pb < 25.000.

La **domanda** di un determinato prodotto consiste nell'insieme di clienti che si riesce ad avere per ogni prezzo possibile del prodotto in questione.

Nel nostro esempio, se tutte le famiglie la pensano come la nostra, per prezzi superiori a 25.000 euro, non c'è domanda per il prodotto della marca B, mentre per prezzi inferiori, si ottiene la domanda data dalla somma di tutte le famiglie che la pensano come la nostra.

Da cosa dipende il valore V che un potenziale cliente assegna ad un prodotto o ad un servizio? Innanzitutto, ovviamente, dalla qualità intrinseca del prodotto capace o meno di soddisfare i bisogni del cliente; ma anche:

1) Dalla capacità del cliente di assegnare il giusto valore al prodotto, cosa che dipende soprattutto dalla sua formazione e dalla sua educazione.

Sarà difficile che un cliente assegni il giusto valore al sistema operativo GNU/Linux, ad esempio, se non sa nemmeno cos'è un sistema operativo, e se non ha mai pensato che un computer non deve per forza avere installato il sistema operativo Microsoft Windows.

2) Dall'importanza della possibilità di disporre di prodotti secondari che siano il giusto complemento al prodotto principale che ci viene offerto (un'auto vale di più se le strade sono migliori e se è facile trovare un benzinaio, vale invece di meno se le strade sono intasate dal traffico, se i mezzi pubblici sono di alta qualità, se è difficile trovare benzinai, ecc.)

3) Dalla diffusione reale del prodotto che ci viene offerto, cioè, dal numero di persone che ne possiede uno: il telefono o la posta elettronica valgono di più se è di più la gente che li usa.

1.2. L'offerta di un prodotto

Da parte sua, l'imprenditore si dedicherà ad un determinato prodotto se può trarne un profitto ragionevole, che gli permetta di coprire due aspetti essenziali:

1) I costi che implica servire il cliente.

2) La cifra che guadagnerebbe se si dedicasse ad un'altra attività.

Esempio pratico

Supponiamo che una coppia decida di aprire un bar. Dopo un anno di attività, hanno ottenuto entrate per 150.000 euro, mentre i costi sostenuti per servire il cliente, per pagare l'affitto del locale, ecc., sono ammontati a 120.000 euro. In questo caso, il primo requisito è stato soddisfatto, dato che le entrate hanno superato abbondantemente le uscite; un ragioniere ci direbbe che abbiamo ottenuto un profitto, dal momento che le entrate hanno coperto i costi.

Immaginiamo però che, per aprire il bar, questa coppia abbia rinunciato ai rispettivi impieghi, che garantivano loro entrate annuali pari a 40.000 euro. Queste entrate alternative sono quelle che gli economisti chiamano costo opportunità legato all'apertura del bar. Il nostro secondo requisito, in questo caso, non è soddisfatto:

Il business non è realmente conveniente, in quanto

Entrate ? costi = 150.000 ? 120.000 = 30.000 < costo opportunità = 40.000

< Coste de oportunidad = 40.000

Ovviamente, questa coppia può comunque preferire il bar all'impiego precedente, ed il loro sacrificio economico ci sembrerà anche ragionevole se compensa in loro la soddisfazione di poter gestire un proprio business. Il nostro pensiero è semplicemente che, in primo luogo, non stanno ottimizzando le loro possibilità economiche; ed in secondo luogo, che la loro decisione ci sembra ragionevole sempre che sia presa lucidamente, cioè, presa pur essendo consapevoli del sacrificio economico implicato; non ci sembra ragionevole se non si rendono conto del fatto che disporranno di meno soldi.

Perché aprire il bar sia realmente un buon affare, bisogna che i benefici che si ottengono eccedano il 'beneficio' dell'attività alternativa, o il costo opportunità.

Se le entrate annuali del bar fossero pari a 180.000 euro, ad esempio, allora sì che staremmo parlando di un buon affare:

Siamo dinanzi ad un buon affare perché

Entrate ? costi = 180.000 ? 120.000 = 60.000 > costo opportunità = 40.000

Arriviamo dunque a concludere che, perché un determinato business venga mantenuto, è necessario che i benefici che si ottengono eccedano il costo opportunità di dedicarsi ad attività alternative.

1.3. La creazione di valore e il vantaggio competitivo

A partire da quello che abbiamo visto, siamo già in grado di stabilire quali sono i requisiti che vanno rispettati perché un business sia conveniente.

In primo luogo, è necessario che si crei del valore, ovverossia, che il valore V che i possibili clienti assegnano al prodotto che viene loro offerto ecceda i costi dell'offerta:

Perché un business sia conveniente, è requisito imprescindibile che l'assegnazione del Valore al prodotto # Costi # Costo opportunità > 0

Solo quando si verifica questa condizione possiamo dire che un'impresa crea valore e che può mantenersi col suo business, perché solo in questo caso possiamo trovare un prezzo che sia ragionevole sia per il cliente che per l'impresa.

Esempio pratico

Se il valore di un prodotto per un cliente è pari a V = 100 euro, ed i costi sostenuti per offrirglielo sono pari a C = 60 euro, si può trovare un prezzo soddisfacente per entrambi, ad esempio fissando P = 80, prezzo che renderebbe valido:

V – P > 0

e

P – Costi totali > 0

Tuttavia, il fatto che venga soddisfatta la condizione V # C > 0 non è sufficiente per stabilire che un business sia sostenibile. Per capirne la ragione, si può ricorrere all'esempio del paragrafo 1.1 (relativo alla domanda del prodotto), ed estenderlo al punto di vista di due imprese concorrenti che cerchino di attrarre un cliente:

Esempio pratico

Supponiamo di avere due costruttori di automobili che offrono due modelli simili. Ricordiamo che la famiglia assegnava un valore all'auto di A pari a Va = 40.000, ed all'auto B un valore Vb = 35.000.

Immaginiamo ora che i costi di fabbricazione del costruttore A siano pari a Ca = 20.000, mentre per l'impresa B siano Cb = 10.000. Entrambi i costruttori producono a costi sensibilmente inferiori ai rispettivi valori Va e Vb.

Se non esistesse un concorrente, senza dubbio entrambe le imprese starebbero facendo un business redditizio.

Immaginiamo che il costruttore B decida di vendere i suoi veicoli ad un prezzo Pb = 18.000. La 'soddisfazione' del cliente sarebbe pari a

Vb – Pb = 35.000 – 18.000 = 17.000.

Il costruttore A deve offrire una 'soddisfazione' maggiore (o almeno uguale) per attrarre il cliente:

L'impresa A 'vince' il cliente se:

Va – Pa > Vb – Pb = 17.000 solo se Pa < 23.000.

Il costruttore A, quindi, ha l'opportunità di attrarre clienti ed allo stesso tempo di coprire le sue spese. Però dobbiamo ricordarci che quest'impresa è alla mercé del suo concorrente:

Se l'impresa B decide di abbassare i suoi prezzi al di sotto di 15.000 (ad esempio Pb = 14.000), l'impresa A non può continuare a 'vincere' clienti senza incorrere in perdite:

Vb – Pb = 35.000 – 14.000 = 21.000 e

Va – Pa > Vb – Pb = 21.000 solo se Pa < 19.000, ma allora

Pa – Ca < 0 !

In questo esempio, l'impresa B ha un vantaggio competitivo rispetto al suo rivale, l'impresa A. La conseguenza è che si verificherà una delle due situazioni che seguono:

1) L'impresa B si accaparra tutta la clientela, come succede se fissa Pb = 14.000, o piuttosto

2) Le due imprese si dividono la clientela, ma l'impresa B guadagna di più per ogni cliente:

Si dividono la clientela se Va # Pa = Vb # Pb, ma questo significa che Pa # Ca < Pb # Cb, ad esempio se Pb = 18.000 e Pa = 23.000.

In sintesi, l'impresa che ha un vantaggio competitivo ha la sopravvivenza assicurata, ed in ogni caso guadagna più dei suoi rivali.

Nell'esempio precedente, l'impresa B godeva di un vantaggio competitivo relativo ai costi: anche se il suo prodotto non era il più adeguato a rispondere ai bisogni dei clienti, Va > Vb, era però in grado di produrre un prodotto ragionevole con costi di gran lunga inferiori a quelli della sua rivale.

Inditex

Un esempio interessante per l'obiettivo di questo corso è l'impresa Inditex (proprietaria della catena di negozi Zara). L'industria dell'abbigliamento alla moda, nel quale si inserisce quest'impresa, è un settore altamente competitivo, nel quale le imprese possono copiare i modelli delle altre senza alcuna restrizione (e questo succede, nonostante l'alto tasso di innovazione che lo caratterizza, rappresentato da nuovi modelli in ogni stagione, anno dopo anno), nel quale sono presenti una grande quantità di imprese concorrenti, e nel quale i prezzi devono essere concorrenziali. In qualità di clienti, quindi, possiamo godere dei vantaggi offerti da un'industria caratterizzata da alta competitività e da grande innovazione.

Nonostante tutte queste difficoltà, Inditex riesce ad ampliare ogni anno la sua quota di mercato (accaparrandosi una parte sempre maggiore della clientela) grazie al suo vantaggio competitivo relativo ai costi, che consiste fondamentalmente in: (1) indovinare rapidamente quali sono i modelli che si vendono di più in una stagione e (2) correggere di conseguenza la produzione verso questi modelli, con la conseguenza che i costi di Inditex sono minori perché non produce abbigliamento che non si vende, e vende molti dei capi che piacciono in una determinata stagione.

A quanto sembra, per altro, riuscire a fare questo non è una cosa banale, in quanto i suoi concorrenti non ne sono capaci (almeno non nelle stesse proporzioni).

Un'impresa con un vantaggio competitivo che risiede nei costi riesce ad attrarre più clienti e ad ottenere maggiori profitti perché può vendere i suoi prodotti a prezzi più bassi.

In alternativa, un'impresa potrebbe avere un vantaggio competitivo relativo alla sua unicità, offrendo, cioè, un prodotto al quale il cliente assegni un valore maggiore rispetto a quello della concorrenza ad un prezzo ragionevole.

L'azienda Adobe

L'azienda Adobe con il suo programma di software Acrobat è sicuramente un buon esempio di un prodotto che si presenta come più attraente ad un prezzo ragionevole.

Questa assegnazione di valore superiore può essere comune, nel senso che tutti i possibili clienti considerano questo prodotto di maggior qualità (è il caso delle auto tedesche di lusso, per esempio), ma può essere anche di nicchia, quando si tratta di un prodotto specializzato, adeguato per una clientela specifica (qualsiasi negozietto di paese soddisfa questo requisito: è un negozio orientato ad una clientela specifica, i residenti del paese, ovverossia gli unici ai quali risulta più comodo comprare in quel negozio il pane o il giornale).

Il vantaggio competitivo relativo all'unicità permette all'impresa di vendere il prodotto più caro senza per questo perdere clienti.

1.4. Riassunto

Abbiamo visto in maniera semplificata in cosa consiste, dal punto di vista economico, creare un **business redditizio**. Consiste, detto in breve, nel creare un prodotto o un servizio appetibile per i clienti, di modo da poter far pagare per ottenerlo, tenendo i costi sotto controllo.

Dal punto di vista della possibilità di creare un business a partire dal software libero, la domanda economica cruciale è: che prodotto/servizio si può far pagare alla clientela? Prima di entrare in quest'aspetto, nella prossima parte del modulo scopriremo una serie di caratteristiche economiche rilevanti dell'industria del software, rilevanti per comprendere la risposta alla domanda appena formulata.

2. Caratteristiche economiche dell'industria del software

Come dicevamo all'inizio, non è cambiata nessuna legge economica, e nessuno dei fenomeni economici legati alle industrie delle tecnologie dell'informazione e della comunicazione (TIC) è qualitativamente nuovo. Se qualcosa è successo, si può dire che ciò che è cambiato è l'importanza relativa di determinati effetti economici nella nostra società. In particolare, nelle industrie che hanno a che fare con le TIC, all'interno dell'interazione di mercato tra imprese e clienti, assumono una grande importanza una serie di fenomeni economici che possono distorcere il funzionamento di questi mercati. A breve ed in breve illustreremo i seguenti effetti:

1) I costi della copia e della distribuzione della tecnologia digitale.

2) L'economia delle idee e la proprietà intellettuale.

3) Complementarietà.

4) Effetti di rete.

5) Prodotti compatibili e standard.

6) Costi di cambio e clienti 'imprigionati'.

7) Politiche di compatibilità e di standardizzazione all'interno di una piattaforma e tra piattaforme.

Un esempio recente di quest'ultimo punto è la **compatibilità tra piattaforme e politica** praticata dall'impresa di **software di proprietà Microsoft**, che ha provocato l'intervento della Commissione Europea in difesa della **libera concorrenza** tra imprese. Per la sua importanza relativa al corretto sviluppo dei modelli di business basati sul software libero, analizzeremo anche, seppur in breve, il punto di vista della Commissione Europea.

2.1. I costi di produzione, copia e distribuzione della tecnologia digitale

La tecnologia digitale presenta una struttura di costi molto specifica: sviluppare un prodotto specifico è molto caro, richiede grossi investimenti e non può essere sviluppato a metà.

Quanto meno, realizzare copie di alta qualità del prodotto sviluppato e distribuirle è relativamente economico.

Pertanto, servire un cliente in più è molto economico; quello che è caro è l'investimento iniziale che permette di sviluppare un prodotto attorno al quale organizzare un business.

L'aviazione commerciale

Un'impresa di aviazione commerciale deve fare un grande investimento in un aereo se vuole stabilire una connessione frequente tra due aeroporti; mezzo aereo non è sufficiente, deve averlo intero. Successivamente, però, servire un cliente in più, fino a riempire completamente l'aereo, risulta molto economico per la compagnia.

Ovviamente, l'enorme riduzione dei costi di copia e di distribuzione dei prodotti e dei servizi mediante la tecnologia digitale ha provocato dei cambiamenti importanti in determinate industrie.

L'industria discografica

Un esempio paradigmatico era l'industria discografica che si fondava sul controllo delle copie (s'intende una copia di qualità, dato che con le tecnologie analogiche una copia realizzata su cassetta aveva una qualità del suono nettamente inferiore ad un disco in vinile o a un CD) e della distribuzione del prodotto (basicamente attraverso negozi specializzati).

2.2. L'economia delle idee e la proprietà intellettuale

Le TIC si caratterizzano per il fatto che permettono di manipolare, trasmettere e riprodurre informazioni o idee. Pertanto, il progresso di queste tecnologie ha come effetto principale quello di **rendere più facile la diffusione delle idee ed il loro utilizzo**.

Le idee posseggono, come bene economico, la peculiarità di essere **beni non rivali**: il fatto che una persona utilizzi un'idea non significa che le altre persone non possano farne uso.

Beni non rivali

Se Pietro mangia una mela, Giovanni non la potrà mangiare. Al contrario, nel caso di una ricetta di cucina, se Pietro la usa, anche Giovanni potrà usarla.

Le industrie delle TIC dedicano una grande quantità di risorse economiche a **sviluppare nuove conoscenze**, con lo scopo di trarre un vantaggio economico dallo sfruttamento di queste idee. Dal punto di vista dell'interesse della società nel suo insieme, ogni volta che viene prodotta una nuova conoscenza (che sia una scoperta scientifica, una nuova tecnica, ecc.), la diffusione di questa nuova idea pone di fronte ad un dilemma. Da un lato, è evidente che, una volta che si disponga di questa nuova conoscenza, l'interesse della società è che venga diffusa il più possibile. Tuttavia, le imprese che hanno sviluppato l'idea, lo hanno fatto per poterne trarre un vantaggio economico, e questo lo possono

ottenere soltanto restringendone l'accesso. Senza una certa **protezione contro la diffusione** immediata di queste conoscenze, si corre il rischio che le imprese decidano di non investire più nella ricerca e nello sviluppo.

Le società avanzate hanno creato diverse istituzioni e dei meccanismi per rendere più facile la creazione di nuove conoscenze scientifiche e tecniche. La ricerca scientifica viene finanziata basicamente attraverso fondi pubblici. Lo sviluppo e il finanziamento di una conoscenza più pratica ed applicata, diretta alla creazione di nuove tecniche di produzione e di nuovi prodotti, viene lasciato generalmente in mano ai privati. In questo caso, ciò che fanno le istituzioni pubbliche è favorire l'attività del settore privato mediante la protezione della proprietà intellettuale, attraverso l'istituzione di una serie di figure giuridiche, essenzialmente **i diritti di copia** (**copyright**), **le patenti** ed **il segreto industriale**.

Il *copyright* protegge l'espressione specifica di un'idea

Ipotesi di *copyright*

L'esempio tipico è il diritto che l'autore di una canzone o di un libro detiene sulla sua opera, perché nessuno possa pubblicarla o distribuirla senza il suo consenso. La persona o impresa che fa una scoperta utile può richiedere che le venga applicata una patente, che vieta, per un certo numero di anni (di norma 20), che altri ne facciano uso senza il suo consenso. Infine, abbiamo la figura del segreto industriale: le imprese possono mantenere il segreto su questa nuova conoscenza, e ricevono una protezione legale contro il furto di essa. In quest'ultimo caso, ovviamente, l'inventore non viene protetto se altri arrivano alla stessa scoperta attraverso il proprio lavoro e in maniera del tutto indipendente.

Anche se un uso adeguato di alcune di queste figure di protezione della proprietà intellettuale può effettivamente favorire il progresso tecnico ed economico, sfortunatamente ci scontriamo con due problemi: che è tutto da verificare che queste figure giuridiche proteggano realmente lo sviluppo delle idee, e che negli ultimi anni molte imprese stano facendo un uso incorretto delle figure giuridiche che potrebbero essere utili. Invece di protegere legittimamente l'innovazione che hanno creato, molte imprese stanno utilizzando i diritti di *copyright* e le patenti di cui dispongono come strumenti anti-concorrenza, per proteggere il loro potere di mercato e rendere più difficile l'entrata di rivali più innovatori.

Nel caso del software, ciò che ha permesso la comparsa di sistemi di proprietà è la facilità per le imprese di mantenere il segreto industriale grazie alla possibilità di distinguere tra il codice fonte ed il codice binario del software. Possiamo utilizzare un programma, ovverossia, possiamo far sì che l'hardware– (un computer, un telefono cellulare, una console, un bancomat, ecc.–) funzioni con un programma informatico, se inseriamo il codice binario nel computer senza disporre del codice fonte. Pertanto, le imprese di software proprietario hanno un modello di business che consiste nel far pagare per rendere disponibile una copia del codice binario del loro software. La conseguenza è che,

senza il codice fonte, non potremo sapere perchè il programma funziona in una certa maniera e non in un'altra, e chiaramente non lo potremo modificare perchè ci permetta di fare altre cose.

Il segreto industriale (non rivelare il codice fonte), quindi, permette alle imprese, da una parte, di nascondere ai rivali il prodotto sviluppato, e dall'altra, nonostante tutto, di vendere un prodotto ai consumatori (il codice binario del programma informatico).

Il **software libero**, in maniera diametralmente opposta, è basato sulla **condivisione del codice fonte** del programma. Come vedremo, questo implica lo sviluppo di un modello di business del tutto diverso, basato sull'offerta di un servizio: la capacità di modificare e di adattare il software ai bisogni del cliente, a partire dalla competenza e dalle conoscenze delle quali dispone l'ingegnere informatico.

Il *copyright*, le patenti e l'innovazione

P. Non è nemmeno un sostenitore delle patenti per il software...

R. Diciamo che sono molto scettico sul fatto che servano a quello per cui sono state ideate. Il software è un'industria nella quale l'innovazione è sequenziale. Ogni nuova scoperta o miglioria viene costruita su quello che è stato sviluppato in precedenza, come se fosse una torre. Una patente applicata ad un determinato livello della torre frena gli sviluppi successivi. Nella pratica, funzionano come un monopolio.

Da un'intervista a Eric Maskin, premio Nobel per l'Economia 2007, pubblicata su *El País*, 29-06-2008.

Lettura consigliata

Potete leggere l'intervista completa nell'articolo pubblicato da *El País*, 29-06-08 "E' difficile prevedere una bolla"

È vero che un autore sarebbe privo di protezione senza l'esistenza di *copyright* o di patenti applicabili alla sua idea? Molti autori pensano di sì. Il famoso cuoco Ferran Adrià, ad esempio, in un colloquio con il responsabile dell'impresa Bimbo, pubblicata su *El País* l'11 agosto 2006, rifletteva:

'Una delle cose che non sono state risolte in questo Paese è la protezione della creatività. Ti possono copiare senza problemi. La ricerca e lo sviluppo non hanno molto senso. Nei ristoranti accade lo stesso'.

...

'Inventi una cosa e dopo un mese te la copiano! Nella vita ci sono cose che sono fatte male, che non funzionano, e questa è una di quelle. Come si può sopportare che tu lavori per anni con infinita dedizione e dopo un mese arriva uno e ti ruba l'idea senza spaccarsi la testa...'

Lettura consigliata

Potete leggere il dialogo completo nell'articolo pubblicato su *El País*, 11 agosto 2006.

È davvero così facile copiare le sue idee? Questo significa che il suo modello di business non può funzionare? Che il suo business funzioni è un fatto. Cos'è che permette a Ferran Adrià di non restare senza clienti?

1) In primo luogo, quello che realmente vende Ferran Adrià ai suoi clienti non è un'idea (una ricetta) ma un piatto cucinato. L'idea, per essere consumata dai suoi clienti, dev'essere applicata ad un piatto concretamente cucinato, così come succede ad uno che compra un'auto: non compra l'idea dell'auto, ma l'auto fisica.

2) In secondo luogo, legato al fatto che consumiamo o usiamo prodotti e servizi che sono la realizzazione concreta di un'idea, non è sufficiente avere l'idea, la 'ricetta', a portata di mano, ma è necessario disporre, per poterla trasformre in un piatto, della perizia e delle conoscenze, oltre che degli strumenti, adeguati. Rispetto a questi ultimi, lo stesso Adrià è solito dire che gli spettatori non devono pretendere di ripetere i piatti da lui cucinati nel suo ristorante, perchè le cucine delle case non dispongono degli utensili necessari. In casa ci raccomanda di cucinare piatti semplici.

Pertanto, l'investimento necessario per gli strumenti che permettono di riprodurre l'idea già di per sè limita il numero dei possibili imitatori, e di conseguenza il numero di copie autentiche, cioè, i piatti effettivamente cucinati da professionisti, che possono competere con i suoi. Questo è un aspetto fondamentale da tenere a mente in qualunque industria. Copiare l'idea non è così ovvio, infatti trasformarla in un prodotto o in un servizio richiede delle conoscenze (siano la perizia derivata dall'esperienza o la conoscenza acquisita studiando, o entrambe le cose) e degli investimenti in macchinari, strumenti, materie prime, ecc. che limitano il livello dell'effettiva concorrenza nell'industria.

Il tecnico professionista

Questa è una cosa che può succedere in qualsiasi attività professionale. Possiamo cambiare o anche aggiustare i termosifoni di casa nostra, ma probabilmente non disporremo degli strumenti che invece possiede l'idraulico (sarebbe una spesa eccessiva comprarli per cambiare un termosifone ogni tot anni), se davvero pensiamo di essere capaci di farlo.

3) In terzo luogo, come ci avverte Maskin per il caso del software (è lo stesso che avviene per il disegno e per lo sviluppo del software), le novità in campo culinario si producono in maniera sequenziale e per accumulazione: ogni nuova ricetta non nasce dal nulla, ma si basa sui risultati precedentemente raggiunti. Questa è però una cosa che lo stesso Adrià notava in una serie di articoli scritti insieme con Xavier Moret e pubblicati nell'agosto del 2002 su *El País*, mentre passava in rassegna i suoi viaggi in svariati paesi:

'Attualmente abbiamo adottato i viaggi come metodo creativo. Prendiamo ispirazione, cerchiamo la scintilla per una nuova idea, prendendo spunto da altre cucine che possano far evolvere la nostra. (...) Questo comportamento, il desiderio e la ricerca per poter conoscere quello che fanno gli altri, è qualcosa di vitale in qualsiasi attività in cui si voglia progredire e migliorare.'

Vista così, l'innovazione non sembra partire da zero, ma, ogni volta che propone una nuova ricetta, si ispira in maggior o minor misura a quelle dei suoi predecessori, che provengano dalla tradizione culinaria dello stesso paese o dalla cucina di altri paesi. La sua reputazione di inventore di ricette e di buon

esecutore di esse (una reputazione costruita attraverso i commenti di coloro che sono stati clienti nel suo ristorante) gli permette di godere di quello che nel paragrafo 1.3 chiamiamo vantaggio competitivo dovuto all'unicità, vantaggio che gli consente di far pagare un prezzo più alto rispetto ad altri cuochi (spesso suoi imitatori) e di non perdere per questo motivo nemmeno un cliente.

In alternativa, un'impresa può basare il suo vantaggio competitivo sui minori costi, come ricordavamo nel caso di Zara: può essere che non sia l'impresa più innovatrice del suo settore, ma si ispira ai modelli di altre imprese o ne adatta i disegni con una certa grazia (il che significa che alla gente piace vestire con i capi che trova nei suoi negozi) ed è capace di mantenere le sue spese ad un livello più basso rispetto a quelle dei rivali.

2.3. Complementarietà

Quando parliamo di software, dobbiamo tener presente che quello cui diamo un valore non è in realtà il prodotto preso separatamente, ma bensì l'insieme dei prodotti complementari tra loro. In effetti, il software non è altro che uno dei pezzi del sistema che ci serve.

L'esistenza di complementarietà è frequente nei prodotti e nei servizi legati alle TIC.

Prodotti complementari

Esistono televisioni di qualità molto diversa, ma in realtà perfino il migliore è un apparato perfettamente inutile se non si dispone di una connessione con i canali tv, di un lettore DVD, ecc.

La complementarietà dei prodotti informatici

Non vogliamo semplicemente poter disporre di un computer (intendendo come tale il semplice oggetto fisico, come quando parlavamo del televisore) ma vogliamo poter disporre anche degli altri oggetti fisici che complementano il computer, come per esempio stampanti, macchine fotografiche digitali, scanner, ecc. E nemmeno tutti questi oggetti ci bastano: abbiamo bisogno anche del software. Abbiamo bisogno di disporre di tutto ciò che fa funzionare il computer (ovvero, il sistema operativo), del software e delle sue applicazioni, che ci permettono di utilizzare il computer per svolgere diversi compiti. Sono esempi di applicazioni il server di internet, la posta elettronica, ecc.

Per questo motivo, la complementarietà dei diversi prodotti che formano un sistema in qualsiasi tecnologia digitale (non solo il computer) comporta che ogni elemento preso separatamente non serva poi a molto. Questo vuol dire che diventa fondamentale che tutti questi pezzi parlino la stessa lingua e che funzionino correttamente tutti insieme. In poche parole, diventa fondamentale che le varie componenti siano compatibili tra di loro.

2.4. Effetti di rete

Diciamo che emergono degli effetti o delle esternalità di rete quando il valore di un prodotto o di un sistema per tutti coloro che lo usano diventa maggiore quanti più sono gli utenti. Le esternalità di rete possono essere di due classi diverse:

1) Dirette.

2) Indirette o virtuali.

L' **esternalità diretta** dovrebbe essere più intuitiva: molto spesso un prodotto ci sembra valere di più quanto più è diffuso, perchè allora possiamo condividerne l'utilizzo con più persone.

Esternalità diretta

Sono esempi quasi banali di questo concetto il telefono, il fax, la posta elettronica, ecc. Inoltre, perchè davvero possiamo godere appieno di tutti questi utenti di apparati come il nostro, è fondamentale che il loro ed il nostro siano compatibili tra loro. Non ci servirà granchè avere un fax e che anche gli altri lo abbiano, se poi il loro fax non accetta o non legge i messaggi che gli invia il nostro.

Come vedremo più in dettaglio nel prossimo paragrafo, i possibili effetti di rete non vengono sfruttati se non si produce un processo di standardizzazione che assicura che gli oggetti in mano a persone diverse siano compatibili tra loro, perchè solo allora potremo davvero comunicare con molta gente.

La telefonia mobile

Negli Stati Uniti le diverse imprese che offrono servizi di telefonia mobile non si misero d'accordo sull'utilizzo di un sistema comune. Pertanto, negli Stati Uniti un telefono cellulare è molto meno utile che in Europa, dove la Commissione Europea ha promosso l'uso di un unico standard comune a tutti i paesi. L'immediata conseguenza è che la telefonia cellulare è molto meno estesa negli Stati Uniti, e questo a discapito dell'intero settore, delle imprese e dei clienti.

Le **esternalità di rete indirette** sono un effetto economico più sottile. Quando un prodotto è di fatto un sistema che si compone di diversi pezzi che s'incastrano, ed ognuno di questi non vale molto senza gli altri, il valore di un prodotto finisce per dipendere dalla sua popolarità, perchè disporremo di più complementi (o di componenti di maggior qualità) quanta più gente sarà interessata al prodotto.

Ad ogni modo, gli effetti indiretti e quelli diretti hanno qualcosa in comune: di nuovo, è fondamentale che le altre persone e le altre imprese dispongano di **prodotti compatibili**.

In questi casi è fondamentale, perchè i mercati di questi prodotti decollino, che si presenti una di queste due situazioni: o che l'Amministrazione Pubblica intervenga, o che prenda l'iniziativa un agente economico che abbia il potere sufficiente per modificare da solo le condizioni del mercato e per disporre di risorse finanziarie sufficienti per sopportare gli anni di adattamento dei clienti.

Effetti indiretti e diretti

Due esempi dell'importanza di questi effetti per il decollo di prodotti che presentano esternalità di rete:

1) I nuovi formati video di alta definizione. Le imprese che fabbricano il nuovo disegno hanno cercato un compromesso con le grandi case produttrici del cinema, le quali hanno

assicurato che avrebbero diffuso le loro nuove produzioni in questo formato. In questo modo, viene garantito ai clienti che dispongono del complemento dei nuovi riproduttori il supporto disegnato per sfruttare la risoluzione di questi apparecchi.

2) Possiamo vedere per mezzo del prossimo esempio che questo effetto economico appare anche in altri settori, non solo nelle TIC. Non vorremo comprare un'auto che utilizzi i nuovi combustili biodiesel (quelli elaborati a partire da oli vegetali) se non possiamo trovare stazioni di servizio che lo vendano. A loro volta, le stazioni di servizio hanno poco interesse a modificare i loro impianti se non pensano che avranno molti clienti, quindi i costruttori non realizzeranno auto a biodiesel, ecc.

In questi casi, non ci rende nessun vantaggio diretto (contrariamente a quello che succede quando anche gli altri hanno un fax) il fatto che anche altre persone abbiano auto che vanno a biodiesel (non c'è nessun effetto diretto); solo se esiste una massa considerevole di persone che posseggano auto a biodiesel, le stazioni di servizio troveranno conveniente adattare i loro impianti al nuovo combustibile. Possiamo affermare che, indirettamente, ogni persona che compra un'auto biodiesel fa un favore a tutti gli altri acquirenti di auto biodiesel.

Le esternalità indirette spiegano l'importanza, per la crescita dell'uso di questo nuovo combustibile, del fatto che l'Amministrazione Pubblica sovvenzioni i costi di produzione, ed anche l'importanza del recente accordo tra **Acciona**, l'impresa spagnola tecnicamente più avanzata nel processo di produzione del biodiesel, e **Repsol**, che possiede la più importante rete di distribuzione di combustibili in Spagna. L'accordo tra queste due imprese assicura che le stazioni di servizio disporranno in un futuro prossimo di carburante biodiesel. A loro volta, adesso i costruttori ed i concessionari saranno invogliati a vendere auto a biodiesel, perchè potranno garantire agli acquirenti che il rifornimento di carburante non sarà complicato.

Quando ci riferiamo a prodotti e servizi che hanno complementarietà ed effetti di rete come caratteristiche importanti, la conseguenza più importante che ne scaturisce è che un prodotto non è utile se non si raggiunge una massa critica sufficiente di utilizzatori: al di sotto di una certa quantità di utenti, il prodotto non offre prestazioni sufficienti a renderlo un prodotto di valore, ed i possibili fornitori di prodotti complementari non apporteranno i necessari investimenti per metterli a disposizione dei clienti.

Il formato VHS

L'inerzia nell'uso di una certa versione può azzerare la sostenibilità di versioni alternative, almeno tecnicamente realizzabili. I riproduttori video **betamax** scomparvero quando tutti decisero di possedere al suo posto riproduttori **VHS**. Benchè la quantità totale di famiglie che avevano dei video crescesse di anno in anno, e che la quantità di film disponibili in video crescesse di pari passo, i proprietari di video betamax non potevano godere di questo incremento perchè la maggior parte dei nuovi titoli veniva offerta solo nel formato VHS, molto più popolare. I costruttori, inoltre, da un certo momento in poi, dedicarono i loro sforzi unicamente al miglioramento dei riproduttori VHS.

Un altro pericolo che proviene da questi effetti è che un'impresa matura, che dispone di una base di clienti considerevole, può bloccare il normale funzionamento della concorrenza attraverso azioni strategiche che rendono difficile o addirittura impossibile che i nuovi prodotti ed i nuovi servizi dei rivali ottengano una massa critica sufficiente.

Nel mondo del software, vedremo che la principale strategia anti-concorrenza consiste nel rendere incompatibile con il prodotto dei rivali il prodotto dell'impresa che domina il mercato.

2.5. Prodotti compatibili e standard

Possiamo definire uno standard come l'insieme delle specifiche tecniche che permettono che tutti i pezzi di un sistema siano compatibili tra loro.

Come abbiamo visto nei paragrafi precedenti, il valore di un prodotto dipende fortemente dall'esistenza di standard universalmente accettati e riconosciuti:

1) Quando un prodotto è fatto di più elementi diversi che si complementano.

2) Quando gli effetti di rete sono importanti.

Nell'industria delle TIC, per quanto riguarda l'hardware (gli apparecchi fisici) è evidente che il processo di standardizzazione, fortunatamente, ha fatto passi in avanti di proporzioni gigantesche. Finalmente, oggi quasi tutte le periferiche di un computer possono essere connesse in un'entrata del computer (ad esempio, un'entrata USB), e quando compriamo una stampante, per esempio, sappiamo di non doverci minimamente preoccupare: di sicuro una volta a casa potremo collegarla facilmente ai nostri computer.

L'obsolescenza delle componenti

Coloro che hanno una certa età, se ci pensano bene, ricorderanno che un po' di tempo fa le cose non erano così. Ognuno di noi ha vissuto l'esperienza di aver comprato un pezzo o un apparecchio elettronico od informatico che è diventato obsoleto per il semplice fatto di non potersi collegare alle altre componenti delle quali doveva entrare a far parte.

Anche i più giovani possono capire quello che dicevamo se pensano a tutti i caricatori che siamo obbligati a portarci dietro dappertutto (quello del cellulare, quello del portatile, ecc.) perchè il caricatore di ognuno di questi apparecchi non funziona con gli altri (molte volte perfino quando sono prodotti della stessa marca!). Se un giorno decidiamo di cambiare cellulare, sappiamo che, disgraziatamente, possiamo già buttare il caricatore di quello vecchio, perchè non ce ne faremo più nulla.

2.6. Costi di sostituzione e clienti intrappolati

Molto spesso ci sono prodotti pensati per offrire un servizio simile che sfortunatamente non sono compatibili tra loro. Fu il caso ai suoi tempi dei dischi in vinile e dei CD, e lo è stato poi dei riproduttori video VHS e DVD.

Oggettivamente, possiamo affermare che in questi due esempi una delle tecnologie è chiaramente superiore all'altra, e che, pertanto, se dovessimo scegliere tra le due tecnologie partendo da zero, non avremmo alcun dubbio su quella da impiegare.

Tuttavia, a causa dell'esistenza di complementarietà, il cambio costò molto caro a tutti coloro che usavano la tecnologia precedente. Coloro che avevano dischi in vinile e volevano passare ai CD, prima di tutto dovevano comprare un lettore CD, ed erano obbligati, oltretutto, a ricomprare i dischi, se volevano ascoltarli con il nuovo lettore.

In generale, a causa delle complementarietà e degli effetti di rete nel mondo delle TIC, passare da una versione del prodotto ad un'altra incompatibile è un processo costoso, fino al punto che, probabilmente, continueremo ad usare la vecchia tecnologia per molto tempo a meno che il miglioramento della qualità non ci sembri molto significativo.

Certamente, nel mondo dell'informatica e del software in particolare questi costi di cambio possono essere rilevanti. Includono i costi per apprendere ad utilizzare nuovi programmi quando siamo abituati ad una determinata versione. Da qui la tendenza dei programmatori a far sì che i nuovi programmi abbiano un'estetica ed un funzionamento simili ai programmi che già conosciamo.

Programmi simili

Il processore di testi OpenOffice imita Microsoft Word, che a sua volta imitava un programma anteriore, Wordperfect, che a sua volta faceva lo stesso con Wordstar (ogni volta, con il processore di testi più popolare del momento); Microsoft Excel imita Lotus 1-2-3, che a sua volta imitava un programma precedente, Visicalc. Potremmo fare molti altri esempi di questo meccanismo.

Dati questi costi per il passaggio da un prodotto ad un altro se i due presentano delle incompatibilità, le imprese mature, con una base di clienti solida, hanno la tentazione di accrescere artificialmente i costi di passaggio stessi, di rendere cioè più difficile per i clienti cambiare prodotto o fornitore.

I costi della sostituzione

Tempo fa, gli antichi monopoli della telefonia, quando stavano comparendo le prime imprese concorrenti, cercarono di obbligare i loro clienti a cambiare numero di telefono se volevano cambiare compagnia (l'idea era che i clienti non avrebbero voluto sostenere il costo implicato dal dover comunicare a tutti i conoscenti il cambio del numero).

Allo stesso modo, nel mondo del software, le imprese mature hanno la tentazione di rendere poco compatibili i propri prodotti con quelli dei rivali.

2.7. Politiche di compatibilità e di standardizzazione all'interno di una piattaforma o tra piattaforme

Come abbiamo visto, è fondamentale per la loro funzionalità che le diverse parti di un prodotto siano compatibili e che prodotti diversi siano tra loro compatibili. Per questo motivo, è importante che si stabiliscano degli standard che permettano di rendere i **prodotti compatibili** tra loro.

Molto spesso, il **processo di standardizzazione** prende inizio dal fatto che il formato di una parte imprescindibile di un sistema viene adottato da tutti. Questa parte imprescindibile che traccia il processo di standardizzazione viene talvolta chiamata piattaforma.

A volte questi processi di standardizzazione sono il risultato del lavoro di organismi creati con l'obiettivo di definire questi standard. Possono essere organismi statali e sovra-nazionali, o anche formati dai membri stessi dell'industria.

Nel mondo del software esistono diversi standard stabiliti attraverso alcuni di questi procedimenti, come per esempio tutti quei protocolli di comunicazione sui quali si regge il trasferimento di informazioni su internet.

Altre volte invece un'impresa dell'industria controlla una parte dell'industria.

Sony e Phillips

Queste due imprese riuscirono ad imporre con la forza dei fatti la loro tecnologia di CD. Questo accade quando tutte le case discografiche distribuiscono la loro musica su quel supporto digitale, quando tutti gli apparecchi musicali sono pensati per riprodurre quel formato, ecc.

Nel mondo del software, ovviamente, l'esempio principale di piattaforma nel senso che le abbiamo appena assegnato è quello del sistema operativo Microsoft Windows, presente nella stragrande maggioranza dei computer, che siano personali o server.

È importante comprendere gli interessi che muovono il proprietario di un prodotto che è diventato, in un modo o nell'altro, una piattaforma. In particolare, vedremo quali interessi guidano le politiche di compatibilità del suo prodotto con i prodotti che lo completano (politiche di compatibilità all'interno di una piattaforma) e con i prodotti che sono potenziali rivali (politiche di compatibilità tra piattaforme).

2.7.1. Politiche di compatiblità e di standardizzazione all'interno di una piattaforma

All'interno di una piattaforma, un ventaglio di applicazioni più grande può dare maggior valore alla piattaforma in due modi: i clienti possono trarre un maggior beneficio dalla piattaforma (e per questo saranno disposti a pagare di più), ed i creatori delle applicazioni, a loro volta, vedranno aprirsi più possibilità di business (avendo una base di potenziali clienti più amplia) e per questo realizzeranno applicazioni che funzionino su questa piattaforma; questo, a sua volta, attrarrà un maggior numero di clienti, ecc. In questo modo si crea un circolo virtuoso che alimenta la diffusione di questo prodotto.

Così, un maggior numero di applicazioni complementa la piattaforma e le conferisce maggior valore. All'inizio, lo sponsor della piattaforma dovrebbe avere interesse ad aprirla ai creatori delle applicazioni, ed infatti Microsoft è solita sostenere che persegue una politica di apertura, in quanto permette ai creatori delle applicazioni di vedere le parti del codice del software Windows (gli API) che questi devono conoscere perchè i loro prodotti siano compatibili con Windows.

Il promotore, tuttavia, avrà interessi contrapposti:

1) Anche se possiede applicazioni che gli garantiscono un alto rendimento, gli interessa peggiorare il rendimento dei prodotti rivali, rendendoli nel peggiore dei casi perfino incompatibili con la sua piattaforma.

2) Può avere timore che determinate applicazioni, a loro volta, possano trasformarsi in nuove piattaforme attorno alle quali sviluppare nuove applicazioni, senza più bisogno di dipendere dalla piattaforma che controlla.

Microsoft e Java

Questo è quello che successe con Netscape e con il linguaggio di programmazione Java: Microsoft realizzò manovre anti-concorrenza contro questi software perchè era preoccupata che, a partire dal loro sviluppo, potessero rimpiazzare Windows come piattaforma software dei PC.

Quest'ultimo esempio ci anticipa, in certa misura, il comportamento che terrà il proprietario di una piattaforma forte, ormai stabile nella sua posizione di standard di fatto, quando gli si parano innanzi altri prodotti che possano soffiargli la posizione di privilegio ottenuta.

2.7.2. Politiche di compatibilità e di standardizzazione tra piattaforme

Abbiamo visto poco sopra che, a causa dei costi di sostituzione, la fetta di clienti della quale dispone l'impresa che controlla la piattaforma può rappresentare una barriera all'entrata per i rivali, quando ci sono effetti di rete, se l'impresa rende incompatibile il suo prodotto con quello dei rivali. Naturalmente, chi esce sconfitto da queste tattiche anti-concorrenza non sono solo le imprese rivali, ma anche la società nel suo insieme, la quale vede immediatamente ridotte le possibilità tra le quali prendere una decisione, ed a lungo andare, la qualità dei prodotti disponibili, dal momento che ci sono meno imprese che osano impiegare risorse per l'innovazione ed il miglioramento dei prodotti.

Prodotti incompatibili, tattiche anti-concorrenza

L'esempio più noto di questa classe di atteggiamento è quello di Microsoft rispetto ai suoi due prodotti leader, il **il sistema operativo Microsoft Windows** ed il **pacchetto Microsoft Office**. Chiaramente, Microsoft fa tutto quello che può per evitare che sia compatible con altre piattaforme (soprattutto con il sistema operativo GNU/Linux). Su questa stessa linea, Microsoft ha perseguito sistematicamente una politica di non sottomissione agli standard creati per l'industria informatica, realizzando la sua propria versione degli standard, senza rendere conto in maniera corretta dei cambi che introduce. Molto spesso, quando vi sono programmi ed applicazioni che, apparentemente, non funzionano correttamente, è perchè la piattaforma non è conforme agli standard approvati dall'industria.

Il conflitto che le varie autorità che difendono gli interessi dei cittadini (negli Stati Uniti così come nell'Unione Europea) hanno scatenato nei confronti di Microsoft ha a che vedere principalmente con questa manipolazione fatta di proposito del processo di standardizzazione di una tecnologia, perpetrata modificando la capacità di comunicazione e di interazione tra piattaforme di informazione.

2.7.3. Politiche pubbliche rispetto al software

In questo paragrafo presentiamo brevemente alcune politiche pubbliche che possono favorire il corretto funzionamento dei mercati del software, e che possono consentire al software libero, in particolare, di competere ad armi pari ed a partià di condizioni come alternativa valida ed efficace rispetto al software di proprietà, in quei casi in cui quest'ultimo parte col vantaggio di disporre di una massa di utenti già formata.

Difesa della concorrenza

In primo luogo, le amministrazioni pubbliche devono garantire il corretto funzionamento della **concorrenza** all'interno del mercato del software.

L'azione principale delle autorità della concorrenza deve garantire che non sorgano incompatibilità artificiose (ovvero, che non abbiano una spiegazione tecnica) tra piattaforme tecnologiche diverse.

Il conflitto che la Commissione Europea ha attualmente in corso con Microsoft è dovuto al fatto che la suddetta impresa manipola il grado di compatibilità tra prodotti diversi, modificando la capacità di comunicazione e di interazione tra piattaforme di software differenti (in questo caso, la comunicazione tra i sistemi operativi che gestiscono server informatici e quelli che gestiscono computer personali).

La Commissione Europea chiede a Microsoft di facilitare a tutti (in particolare ai produttori di server informatici ed ai programmatori) i protocolli d'informazione del sistema operativo Windows, perchè gli altri sistemi operativi siano compatibili con questo sistema; cioè, perchè tutti gli altri sistemi operativi possano comunicare ed interagire con i server che funzionano col summenzionato sistema operativo.

Naturalmente, l'intenzione di Microsoft è quella di sfruttare del fatto che già dispone di un radicamento molto forte del sistema operativo Windows, facendo salire artificiosamente i costi del passaggio ad un altro software da parte dei suoi clienti.

Politiche di adozione e di sostegno del software libero. Rafforzare il rispetto degli standard

Abbiamo rimarcato l'importanza degli effetti di rete nelle TIC e la necessità di una massa critica di utenti perchè un software sia sostenibile. A causa di questi effetti di rete, le grandi imprese possono esercitare una leadership nell'impianto del software libero. Se l'Amministrazione Pubblica e le grandi imprese (per suo interesse o come servizio alla comunità) sostenessero il soft-

ware libero nelle loro organizzazioni, potrebbero ottenere la massa critica sufficiente perchè i cittadini possano essere agevolati nell'utilizzo del sofware libero.

Molto del software di proprietà che viene usato oggi in queste organizzazioni potrebbe essere rimpiazzato senza problemi dal software libero con prestazioni simili o a volte addirittura migliori, e l'unico ostacolo che potrebbe sorgere sarebbe dovuto al costo della sostituzione per ogni utente individuale, a causa dell'assenza di una massa critica sufficiente.

L'effetto di rete cui darebbe luogo questa politica delle organizzazioni menzionate è importante, soprattutto per gli effetti di rete indiretti che si verrebbero a generare: l'acquisizione di software libero da parte di queste grandi organizzazioni darebbe vita ad una fonte di business importante per le imprese informatiche che volessero strutturare il loro modello di business attorno al software libero ed alla prestazione dei servizi di perizia informatica che sono complementari al suo impianto.

In ogni caso, in primo luogo, questi organismi dovranno seguire un processo di acquisizione di software che esige il compimento di determinati protocolli e di standard di compatibilità. Stabilire dei procedimenti di acquisizione di software e di servizi informatici adeguati da parte delle amministrazioni pubbliche, ad esempio, richiederebbe la creazione di una agenzia pubblica incaricata di monitorare e di assistere i diversi dipartimenti governativi. Questi organismi possono implementare diversi meccanismi di sostegno all'uso del software libero all'interno degli organismi pubblici.

Riepilogo

Il business delle tecnologie dell'informazione e della comunicazione ha delle peculiarità specifiche che influiscono sul modello economico del business, e di conseguenza sul mercato.

Oltre che con la creazione di valore nei prodotti e con la gestione che porti ad ottenere un vantaggio competitivo rispetto ai concorrenti nel mercato, l'impresa può avere voce in capitolo sugli effetti economici del mercato mediante lo stabilimento di una politica strategica specifica:

- Mentre i costi di produzione sono elevati, i costi di copia sono minimi.
- Lo sfruttamento delle idee e la salvaguardia della proprietà intellettuale.
- Lo sfruttamento delle complementarietà del prodotto.
- L'effetto rete del prodotto, che sia relazionando il valore con l'utilizzo di massa o come promotore indiretto di complementi.
- La compatibilità tra prodotti rivali.
- Il controllo dei costi di sostituzione di fronte all'evoluzione del prodotto ed alla cattività dei clienti.
- La pianificazione e la messa in atto di politiche relative a compatibilità e standardizzazione all'interno ed all'esterno delle piattaforme.

Attraverso tutto questo, le sue peculiarità permettono al software libero di stabilire un nuovo formato di business che rompe le politiche abituali di un mercato tecnologico molto tradizionale quanto al posizionamento della concorrenza.

Bibliografia

Boldrin, Michele; Levine, David (2008). *Against Intellectual Monopoly*. Cambridge: Cambridge University Press. <http://levine.sscnet.ucla.edu/general/intellectual/againstfinal.htm>

Jaffe, Adam B.; Lerner, Josh (2004). *Innovation and Its Discontents*. New Jersey: Princeton University Press

Lerner, Josh; Tirole, Jean (2002). "Some Simple Economics of Open Source".*The Journal of Industrial Economics* (pag. 197-234).

Perens, Bruce (2005, octubre). "The emerging economic paradigm of open source". *First Monday.* Special Issue #2: Open Source. <http://firstmonday.org/issues/special10_10/perens/index.html>

Shapiro, Carl; Varian, Hal (1999). *Information Rules: A Strategic Guide to the Network Economy*. Boston: Harvard Business Press

Stampa

Adrià, Ferran (1 agosto 2002). "Cazadores de ideas". *El País*.

"Aquí unos amigos" (intervista a Ferran Adrià, 19 luglio 2008). *El País*.

"Does IT matter?" (1 aprile 2004). *The Economist*.

"Es difícil prevenir una burbuja" (intervista a Eric Maskin, 29 giugno 2008). *El País*.

<http://people.ischool.berkeley.edu/~hal/people/hal/NYTimes/2004-10-21.html>

"El tomo ha muerto, viva la red" (22 luglio 2007). *El País*. Negocios.

"Prince vuelve a enfurecer a la industria musical" (15 luglio 2007). *El País*.

"Star Turns, Close Enough to Touch"(12 luglio 2007). *New York Times*.

Varian, Hal (21 ottobre 2004). "Patent Protection Gone Awry". *New York Times*.

Il mercato del software

Lluís Bru Martínez

PID_00145048

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

In questo modulo si presentano le caratteristiche principali del mercato del software in generale, e come può inserirsi il modello del software libero in questo mercato.

Nella prima parte scopriremo che è piuttosto comune poter avere accesso a prodotti di libera diffusione o direttamente gratuiti, e studieremo il particolare funzionamento di questo business.

Nella seconda parte viene presentato brevemente il mercato obiettivo del software, così come i mezzi cui i potenziali clienti fanno più comunemente ricorso per ottenere il prodotto.

Obiettivi

Al termine di questo modulo, lo studente dovrà saper:

1. Conoscere le caratteristiche del mercato dei prodotti di libero accesso.

2. Comprendere la relazione tra il software libero e lo sfruttamento di modelli di business paralleli.

3. Capire le implicazioni della distribuzione del software nel modello di business.

4. Approfondire i concetti di sviluppo del software e relazionarli con le caratteristiche del software libero.

1. Business con caratteristiche simili al software libero

Dopo aver osservato le nozioni di economia basilari, adesso possiamo riformulare la domanda che abbiamo lasciato in sospeso nel paragrafo 1.4 del primo modulo 'Ricapitolazione':

Se il software è libero, ovvero, se per definizione tutti possono avere accesso a questo software -eventualmente a costo zero-, come può essere possibile che ci siano degli informatici (ed imprese d'informatica) che si dedicano a programmare il software libero? Possiamo essere certi che in futuro verranno dedicate risorse (tempo e denaro) al suo mantenimento ed al suo sviluppo?

1.1. E' davvero così scioccante il fatto che il software possa essere libero?

Detto in altro modo, è davvero così poco frequente che un prodotto sia distribuito liberamente o eventualmente gratis? Se facciamo un po' di attenzione, abbiamo sott'occhio alcuni modelli di business che si basano sull'offerta gratuita del prodotto alla clientela.

Prodotti gratis

In Spagna ci sono canali come Antena 3, Cuatro, Telecinco e La sexta che offrono gratis il loro segnale televisivo allo spettatore. Ovviamente, il business di queste televisioni consiste nel vendere pubblicità, ovvero nel fare da intermediario tra le imprese che vogliono far conoscere il loro prodotto e la potenziale clientela (ad esempio, i telespettatori che vedranno la pubblicità collocata prima, durante e dopo la trasmissione di una partita di calcio da parte di Cuatro).

Detto in generale, ogni impresa che opera come intermediaria tra altre imprese ed i loro clienti deve decidere la politica di prezzo da adottare. Probabilmente l'opzione migliore è quella di rinunciare direttamente a guadagnare con alcuni di questi clienti.

Diversi modelli di business basati sull'offerta gratuita

Se una televisione vuole ottenere delle entrate dalla pubblicità, deve assicurare ai suoi clienti solventi (le imprese che mettono annunci pubblicitari) il maggior numero possibile di telespettatori, e il modo migliore per trovarli consiste nel concedere gratuitamente la ricezione del segnale televisivo.

Allo stesso modo, se **Adobe** vuole avere clienti per il suo prodotto che serve a comporre file con estensione .pdf, Adobe Acrobat Professional, ha senso che offra gratis la versione semplificata del suo software, Adobe Acrobat Reader. In questo modo, Adobe può assicurare ai suoi clienti solventi che gli altri utenti potranno leggere davvero i documenti elaborati con la suddetta estensione.

Il discorso vale anche per **Amazon**. Amazon, oltre ad essere una libreria che vende su internet, ha tramutato la sua pagina web in una piattaforma attraverso la quale mette in contatto i suoi clienti con librerie di seconda mano, che offrono libri usati con uno sconto. Se consultiamo la disponibilità di un titolo, compare l'offerta diretta di Amazon insieme a quella delle altre librerie. In questo caso, Amazon offre gratuitamente ai suoi clienti la possibilità di consultare l'elenco dei libri disponibili, mentre chiede alle librerie di pagare per il suo servizio di intermediazione. Alle ragioni precedentemente espresse circa la validità di queste politiche di prezzo, si aggiunge che è conveniente che Amazon faccia pagare per le vendite delle altre librerie, perché altrimenti potrebbe essere tentata di offrire un servizio inaffidabile o distorto (per poter garantire la vendita dei suoi propri prodotti, invece che i libri venduti dai suoi concorrenti attraverso la sua pagina web).

In alternativa, un'impresa può offrire gratuitamente un prodotto al cliente, ma legato ad un altro prodotto, che è quello che si vuole in realtà vendere. Abbiamo un esempio di questa strategia:

> 'Tutti coloro che hanno comprato stamattina il settimanale britannico *Mail on Sunday* si sono portati a casa una copia gratis del nuovo lavoro di Prince, *Planet Earth*. In totale sono state vendute 2,9 milioni di copie.'
>
> [...]
>
> '*Planet Earth* sarà distribuito gratis anche a coloro che partecipino ad uno dei 21 concerti che il musicista di Minneapolis terrà a Londra tra il primo di agosto ed il 21 di settembre nell'arena 02.'
>
> *El País*, 15 de luglio 2007.

Lettura consigliata

Potete leggere l'articolo completo pubblicato su*El País*, 15 luglio 2007 "Prince vuelve a enfurecer a la industria musical".

Molto probabilmente, nel primo caso è il periodico che acquista il diritto a regalare delle copie insieme col suo giornale (una maniera di fare pubblicità al periodico), mentre nel secondo caso è l'artista che rinuncia a guadagnare con la distribuzione delle copie del disco (avversando così gli sforzi di discografici e negozi di musica, che vogliono mantenere intatto a tutti i costi il loro modello di business), per poter aumentare le entrate attraverso i concerti. (un'altra notizia, questa volta tratta dal *New York Times*, segnala che lo stesso musicista realizza concerti esclusivi in locali di piccole dimensioni, per 3000 dollari -12 luglio 2007, 'Star Turns, Close Enough to Touch').

Lettura consigliata

Potete leggere l'articolo completo pubblicato su*El País*, 12 lueglio 2007 "Star Turns, Close Enough to Touch".

1.2. Il software come parte di un prodotto

Il software è solo una componente di un prodotto (per quanto importante sia), un pezzo o un complemento del prodotto finale che vogliamo ottenere, e quello di cui vogliamo poter disporre è di tutti i pezzi congiuntamente (del computer e del software allo stesso tempo, per esempio).

Per questo, grandi multinazionali del settore informatico come IBM e Sun Mycrosystems finanziano degli informatici per lavorare sullo sviluppo del software libero. Egoisticamente (nel senso di pensare basicamente ad un incremento dei propri profitti), la loro motivazione è che pensano che in questo modo aumenteranno le vendite dei prodotti e dei servizi complementari per i quali fanno pagare i loro clienti.

Allo stesso modo, le principali imprese che fabbricano telefoni cellulari (Nokia, Motorola, Siemens, Samsung, ecc.) si sono associate (ed immettono delle risorse economiche) per creare il consorzio Symbian, che si occupa di creare del software libero, pensato per essere un programma che permette il funzionamento dei telefoni cellulari che fabbricano. In questo modo, tutti i fabbricanti di telefoni cellulari usano una stessa piattaforma (lo stesso sistema operativo), basata sul **sistema operativo GNU/Linux**,, che è sufficientemente flessibile da permettere ad ogni fabbricante di creare modelli di telefoni cellulari diversi da quelli dei concorrenti, apportando al sistema delle migliorie e delle modifiche che permettano loro di attrarre clienti (telefoni che sono anche macchine fotografiche, che permettono di mandare mail, ecc.). Ogni impresa modi-

fica l'aspetto dello schermo del telefono perché meglio di adatti alle prestazioni che offre, in quanto può accedere al codice fonte del programma che fa funzionare l'apparato telefonico. Così facendo, si fomentano l'innovazione ed il miglioramento del prodotto, perché le imprese credono molto nell'attrarre nuovi clienti creando una macchina (il telefono) che funzioni meglio di quella dei concorrenti.

Il fatto che delle grandi multinazionali abbiano integrato completamente il software libero come fondamenta delle loro attività è garanzia del futuro sviluppo di questo software. È anche garanzia del fatto che, su iniziativa personale, ci saranno ingegneri informatici che si dedicheranno a sviluppare il software libero. Come affermano Lerner e Tirole (2002), questi ingegneri, partecipando al miglioramento del software, possono dimostrare la loro capacità professionale alle imprese di questo settore. Questo farà sì che siano molto ricercati dalle imprese del settore informatico, e pertanto permetterà loro di migliorare le loro prospettive di lavoro.

1.3. Consegna del software. Distribuzione.

Che il software sia libero non significa che non possano nascere imprese esclusivamente dedicate all'attività di consegna dei prodotti e dei servizi ad esso legati.

Per cominciare, una possibile attività è quella di distribuire il software libero. Oltre a vendere cd che lo contengono, offrono anche supporto tecnico agli utenti ed alle imprese che decidano di usare il software libero (Red Hat è l'esempio più conosciuto di impresa che ha sviluppato questa linea di business). L'impresa, quindi, offre la sua esperienza e la sua conoscenza del software al cliente, assicurandogli il supporto tecnico di cui può aver bisogno.

Se ci pensiamo bene, questo modello di business non è infrequente. Ad esempio, sappiamo che l'editore Aranzadi ha organizzato un modello di business molto simile.

Aranzadi

Aranzadi offre ai suoi clienti (professionisti del diritto) una fonte esaustiva di informazioni giuridiche. In più, offre loro l'appoggio tecnico necessario per poter utilizzare agevolmente tutta questa informazione

L'informazione è sempre stata libera e disponibile (le leggi vengono pubblicate nella Gazzetta Ufficiale ed ogni ufficio legale ne riceve una copia). Tuttavia organizzare l'informazione in un modo utile è complicato; è questo il servizio che queste imprese editrici mettono a disposizione dei loro clienti. Ovviamente queste imprese hanno incluso le tecnologie digitali tra i servizi da offrire ai loro clienti, come si può leggere nella notizia che riportiamo qui sotto:

Gli uffici legali e quelli dei commercialisti continuano ad essere 'addobbati' di una moltitudine di solenni tomi giuridici. Ma questi stanno diventando sempre meno un adorno. La maggior parte dei professionisti del Diritto sono passati all'uso di internet per accedere alle informazioni necessarie al loro lavoro, una autentica rivoluzione favorita dalle grandi imprese editrici giuridiche, come Corporazione del Diritto, e che rappresenta un esempio di successo delle nuove tecnologie.

Corporazione del Diritto fornisce informazioni giuridiche ai commercialisti (contando su contributi del Ministero della Giustizia) ed informazioni tributarie di base all'Agenzia delle Entrate.

El País, 22 luglio 2007.

Lettura consigliata

Potete leggere l'articolo completo pubblicato su *El País*, 22 luglio 2007 "El tomo ha muerto, viva la red".

1.4. Consegna del software. Servizio

In termini più generali, l'ingegnere informatico che lavora sul software libero è un professionista simile al cuoco, al meccanico, all'idraulico o all'avvocato.

Un ufficio legale lavora con delle conoscenze e delle informazioni sulla legislazione che sono tanto libere e a disposizione di chiunque, come può esserlo il software libero. Chiaramente, il loro modello di business consiste nell'ottenere dei profitti tramite un prodotto complementare, che è la loro abilità, cioè la loro esperienza, la loro profonda conoscenza delle leggi e la loro capacità di organizzare l'informazione contenuta nelle leggi in modo tale da poter difendere gli interessi dei loro clienti, tutte cose che non si vede perché i loro clienti dovrebbero essere altrettanto capaci di fare.

Essenzialmente, l'avvocato unisce le idee giuste al prodotto giusto per il suo cliente (la difesa dei suoi interessi).

In maniera assai simile, l'ingegnere informatico che lavora con il software libero offrirà ai suoi clienti la sua abilità, la sua capacità di soddisfarne il bisogno di avere una informazione correttamente organizzata e di processarne i dati, a partire dalle possibilità intrinseche del software libero disponibile, o, eventualmente, sviluppandone un ulteriore e più specifico codice.

In questo modo abbiamo visto come un determinato settore economico (i servizi legali) presenta perfino diversi strati di informazione (Corporazione del Diritto e Aranzadi da un lato, gli uffici legali dall'altro), il che dà luogo a molteplici modelli di business che convivono allo stesso tempo.

2. Chi ha bisogno del software?

2.1. Il software, una necessità di base in qualunque impresa

Chi sono i clienti dell'industria del software? Potenzialmente, oggi, qualunque impresa. Come mette in evidenza Nicolas Carr in 'IT doesn't matter', le TIC sono diventate degli strumenti indispensabili per tutte le imprese, allo stesso modo in cui oggi tutte le imprese sono collegate alla rete elettrica per illuminare gli uffici e fornire energia alle loro macchine; tutte dispongono di una linea telefonica; o ancora tutte utilizzano auto o camion per il trasporto delle loro materie prime e dei loro prodotti.

Lettura aggiuntiva

N. Carr (1 aprile 2004). "Does IT matter?". *The Economist*. <http://www.nicholasgcarr.com/articles/matter.html>

Quando Carr diceva nel suo articolo che 'le TIC non sono più importanti', quello che voleva dire è che un'impresa non avrà più alcun vantaggio competitivo per il fatto di utilizzarle, dato che tutte le imprese le utilizzeranno.

Prenotazione di biglietti on-line

Un caso molto noto è quello delle compagnie aeree che hanno sviluppato i primi software di prenotazione di biglietti; allora, il poter disporre di questo software diede loro un enorme vantaggio rispetto ai loro concorrenti. Oggi, ormai, tutte le compagnie aeree possiedono una pagina web attraverso la quale si può fare la prenotazione e l'acquisto di biglietti aerei on-line; ovviamente, quindi, questo software non sarà più ragione di alcun vantaggio competitivo rispetto alla concorrenza.

Questa evoluzione nell'uso delle TIC può essere un vantaggio per lo sviluppo del software libero, in quanto riduce la possibilità che le imprese si lascino guidare dall'illusione che il disporre di un software di proprietà per i loro processi interni possa dare loro un vantaggio competitivo. Quanto più qualsiasi impresa può disporre di software con caratteristiche simili, tanto più (probabilmente) la cosa migliore sarebbe utilizzare un software libero, che possa godere degli sviluppi creati per altre attività, ma adattabili alle necessità concrete dell'impresa.

2.2. Paradigmi di sviluppo del software

Se abbiamo detto nel precedente paragrafo che tutte le imprese ormai avranno bisogno di usare le TIC, ed in particolare che avranno bisogno di usare un software, come fa un'impresa ad ottenere il software di cui ha bisogno per i suoi processi produttivi?

Secondo la classificazione proposta da Bruce Perens in 'The emerging economic paradigm of open source', possiamo classificare le varie maniere per ottenere il software così:

Lettura obbligatoria

B. Perens (2005). *The emerging economic paradigm of Open Source*. <http://www.uic.edu/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1470/1385>

1) Il modello di Microsoft o Adobe (il modello 'Retail' secondo Perens), nel quale un'impresa di dedica a sviluppare il software per conto suo e lo vende impacchettato ai suoi clienti.

Pertanto, visto dal punto di vista del cliente, quest'ultimo si libera dallo sviluppo del software e si limita a comprarlo finito.

Le conseguenze del modello al dettaglio (*retail*)

Naturalmente, questo sviluppo di software prende solitamente la forma di un software di proprietà (in cui il fornitore non mostra il codice ai suoi clienti). Dal punto di vista di chi acquisisce questo software, il primo problema evidente è che non è pensato per le sue necessità concrete (evidentemente, infatti, dev'essere venduto in maniera molto omogenea, perché possa interessare a più clienti diversi). Un altro problema potenzialmente grave è, come abbiamo visto prima, il pericolo di rimanere intrappolato dal fornitore, che renderà difficile il passaggio ad un altro software, oppure recuperare determinate basi di dati, ecc. Al contrario, sebbene con conseguenze simili, si corre il pericolo che l'impresa fornitrice scompaia e smetta quindi di fornire i servizi richiesti per il mantenimento e lo sviluppo del software.

2) Il modello in cui l'impresa che ha bisogno del software lo sviluppa per conto suo, o con informatici già facenti parte dell'impresa, o pagando una ditta specializzata perché le fornisca lo sviluppo del software (il modello 'In-House and Contract' secondo Perens).

I costi dello sviluppo

Senza dubbio, questa modalità di sviluppo del software necessaria per l'impresa può essere molto cara, e può portare a ripetere alcune parti della programmazione che era già stata sviluppata.

Nei due ultimi modelli di sviluppo secondo la classificazione di Perens, le imprese cercano altre imprese con le quali collaborare per lo sviluppo del software di cui hanno bisogno.

3) In questo modello il consorzio elabora un software che non è libero (non sarà, cioè, a disposizione delle imprese che non partecipano al suo sviluppo).

4) Nell'ultimo modello, le imprese del consorzio sviluppano del software libero, ovverossia con il codice aperto a qualunque altra impresa, anche se non partecipa al suo sviluppo.

Gli ovvi vantaggi di quest'ultimo sono legati alla possibilità di accedere agli sviluppi approntati dalla comunità di programmatori che si crea attorno al progetto, con una conseguente diminuzione dei costi di sviluppo.

Ovviamente, lo sviluppo di software libero non sarà gratis per le imprese del consorzio, e dovranno finanziare un gruppo iniziale di programmatori. Il pericolo di un consorzio (tanto di software di proprietà come di software libero) è che si abbia una mancanza di leadership nello sviluppo del progetto, perché

nessuna impresa vuole assumersi la responsabilità di assicurarne lo sviluppo, la quale impedisca la sua esecuzione (o fin dall'inizio, o quando dei nuovi sviluppi richiedano delle nuove spese).

Riepilogo

Nella nostra realtà più prossima ci troviamo immersi in uno scenario nel quale confluiscono molteplici modelli di business con differente politiche attraverso le quali raggiungere i propri obiettivi. Dalla promozione diretta del prodotto in sé, alla distribuzione di prodotti gratis per far conoscere un nuovo mondo di prodotti e di servizi complementari al cliente.

Il business del software libero adotta quest'ultima forma di mercato, essendo capace di stabilire business paralleli e complementari a partire dalla sua promozione. Sono molte le imprese e le multinazionali che oggigiorno hanno adottato una posizione chiara a favore dello sviluppo del software libero, soprattutto tenendo in considerazione che il software è un prodotto fondamentale e di base per qualunque impresa e che il modello di sviluppo del software libero offre loro garanzie per raggiungere questi obiettivi.

Bibliografia

Karminski, D. (1999). "Core Competencies: Why Open Source Is The Optimum Economic Paradigm for Software". <http://www.doxpara.com/read.php/core.html> [Consulta: febbraio 2009]

Perens, B. (2005). "The Emerging Economic Paradigm of Open Source". Pubblicato su *First Monday , Special Issue #2, 3/10/2005*. Cambridge: Cambridge University Press. <http://www.uic.edu/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1470/1385> [Consulta: febbraio 2009]

Stampa

"Prince vuelve a enfurecer a la industria musical" (15 luglio 2007). *El País*.

"Star Turns, Close Enough to Touch" (12 luglio 2007). *New York Times*.

Il software come business

Irene Fernández Monsalve

PID_00145051

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

In questo modulo presteremo attenzione alla visione più 'classica' del software come business. Incentreremo il testo sul punto di vista del software di proprietà, lasciando ad un modulo successivo lo studio delle possibilità addizionali che offre il software libero nel contesto di questo scenario. Anche se alcuni degli aspetti che tratteremo diventano irrilevanti nel momento in cui vengono applicate strategie di software libero, altri continueranno ad essere validi e vigenti.

Ripasseremo alcuni dei fattori che risultano essere chiave nel momento in cui si ordisce un business attorno al software, come la **scelta dell'attività principale** ed il focus generale dell'impresa (vendere prodotti o servizi), aspetti di **commercializzazione** e **marketing** (come scegliere un mercato e come entrarvi) e la **definizione dei suoi prodotti o servizi** (che tipo di prodotti o servizi verranno sviluppati e come verranno posizionati).

Obiettivi

Al termine di questo modulo, lo studente dovrà essere in grado di:

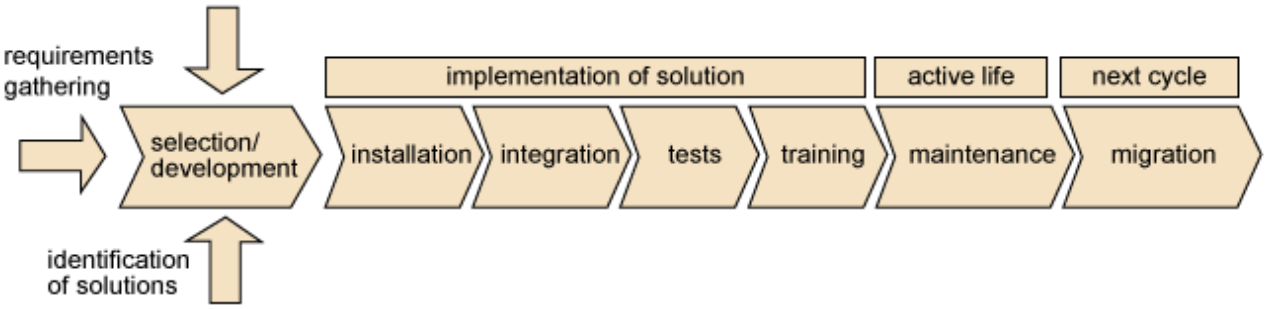
1. Maneggiare una visione globale delle possibilità di business legate al software.
2. Conoscere i modelli tradizionali delle imprese di software.
3. Comprendere le caratteristiche economiche e le differenze tra le imprese che offrono prodotti e quelle che offrono servizi.
4. Identificare i fattori chiave che deve tenere in considerazione un'impresa di software per posizionare i suoi prodotti sul mercato.

1. Possibilità di business legate al software

Tanto gli individui quanto le imprese hanno una necessità di software che rende appetibili molteplici opportunità di business.

Il compito centrale per soddisfare queste necessità sarà la creazione di questo software, il puro lavoro di sviluppo. Tuttavia non finiscono qui le necessità cui si può far fronte, al contrario non sono che appena iniziate. Una volta che si dispone di un prodotto, nascono una serie di bisogni ad esso legati, di consulenza, installazione, configurazione, manutenzione, supporto e formazione, per i quali certi clienti (soprattutto altre imprese) saranno pronti a pagare.

Lungo tutto il processo di adozione di una tecnologia, dall'identificazione delle necessità alla decisione di costruire o comprare, e fino al termine della sua vita utile, nascono molteplici necessità alle quali imprese diverse possono dare una risposta:



Processo di adozione di una tecnologia *(elaborato a partire da Carlo Daffara. "Sustainability of FLOSS-Based economic models". II Open Source World Conference.* Málaga. Disponibile su: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

D'altra parte, il processo stesso di creazione del software può essere inteso in due maniere diverse: come la creazione di un prodotto o come la prestazione di un servizio. La scelta tra le due sarà determinante al momento di dover definire il funzionamento dell'impresa ed il potenziale di generazione di entrate che può presentare, dando luogo a modelli di business molto diversi.

Questo cammino a due direzioni– -sviluppare software come prodotto o come servizio–- riflette a sua volta la prima domanda che un'impresa che utilizza software dovrà porsi quando giunga il momento di adottare una soluzione tecnologica: comprare un prodotto standard o commissionare uno sviluppo su misura.

Possiamo distinguere, pertanto, le seguenti attività imprenditoriali basate sul software:

- Sviluppo di applicazioni
  - Come prodotto: soluzioni standard (*shrink-wrapped*)

  - Come servizio: sviluppo su misura

- Prestazione di servizi attorno ad una o più applicazioni
  - Consulenza
  - Selezione
  - Installazione
  - Integrazione
  - Formazione
  - Manutenzione ed assistenza
  - ecc.

- Software come servizio (*software as a service*)

Questa classificazione non pretende di essere esaustiva nè esclusiva, per cui molte imprese implementeranno modelli ibridi che permettano loro di offrire soluzioni integrate ai suoi clienti.

Le caratteristiche delle imprese di software, così come le sue dinamiche di business, varieranno molto secondo le attività sulle quali si focalizzeranno, come vedremo in seguito, ma qualunque modello ha un potenziale adatto a generare tanto imprese forti come alti profitti.

1.1. Imprese che prestano servizi

Come si è detto in precedenza, le imprese potranno implementare più attività allo stesso tempo, specializzandosi su uno o più aspetti della catena del processo di adozione di una tecnologia.

In questo senso, per le imprese che includono diversi servizi nel ventaglio della loro offerta commerciale possiamo distinguere due tipi di specializzazione: verticale ed orizzontale.

1.1.1. Specializzazione verticale

In generale, le imprese che hanno come attività principale lo sviluppo tenderanno a presentare una **specializzazione verticale**. Se la loro strategia di business è centrata sullo sviluppo su misura, le sue attività includeranno naturalmente il resto dei servizi relazionati, come **installazione**, **integrazione** e **formazione**. Però, come vedremo più avanti, quelle imprese che hanno scelto una strategia di software come prodotto faranno bene a sfruttare anche il resto dei servizi associati, come maniera di assicurarsi un flusso costante di entrate.

	Pacchetto 1	**Pacchetto 2**	Pacchetto 3	Ecc.
Sviluppo	X	X		
Installazione	X	X		
Integrazione	X	X		
Certificazione	X	X		
Formazione	X	X		
Manutenzione ed assistenza	X	X		
Migrazione	X	X		

Specializzazione verticale (basata su Daffara. "Sustainability of FLOSS-Based economic models". *II Open Source World Conference.* Málaga. Disponibile su: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

È interessante notare come un'impresa che investe una certa quantità in licenze di software si attende di investire una cifra almeno pari per i servizi ad esse relazionati, come manutenzione ed assistenza, e per gli aggiornamenti. In questo modo, la vendita di prodotti ad un cliente impresa aprirà le porte alla stipula di contratti di servizi con il cliente stesso, e pertanto ad un flusso di entrate costante nel tempo.

1.1.2. Specializzazione orizzontale

Dal canto loro, le imprese che sfruttano le necessità derivate dall'uso generale di prodotti di software spesso includeranno servizi a vari pacchetti, concentrandosi su una o varie fasi dell'adozione di una tecnologia.

	Pacchetto 1	**Pacchetto 2**	**Pacchetto 3**	**Ecc.**
Selezione / sviluppi su misura				
Installazione				
Integrazione				
Certificazione	X	X	X	X
Formazione	X	X	X	X
Manutenzione ed assistenza				
Migrazione				

Specializzazione orizzontale (basata su Daffara. "Sustainability of FLOSS-Based economic models". *II Open Source World Conference.* Málaga. Disponibile su: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf).

Anche se esistono imprese specializzate nella formazione o nell'assistenza, spesso le imprese di servizi abbracciano varie delle fasi descritte, dando vita a tipologie come, ad esempio, quella della consulenza (concentrando selezione,

consulenza vera e propria e/o certificazione), o quella dei fornitori di soluzioni integrate, che assistono tutte le categorie, includendo sviluppi su misura e perfino la fornitura di hardware.

Le imprese che creano distribuzioni di GNU/Linux lo fanno secondo un modello di **prestazione di servizi con specializzazione orizzontale**.

Esempio di specializzazione orizzontale

Canonical, creatrice della distribuzione basata su **Debian Ubuntu**, realizza un'opera di selezione ed integrazione orizzontale, mettendo insieme un sistema operativo combinato con diverse applicazioni, con l'obiettivo principale di permettere una facile distribuzione, di installare e configurare secondo il motto "linux per gli esseri umani". Tuttavia, dato il carattere libero di Ubuntu, le entrate di Canonical provengono dai servizi relazionati, come assistenza, formazione e certificazione.

Queste imprese orientate ai servizi mettono spesso in evidenza che i loro clienti preferiscono ricevere soluzioni complete e avere a che fare con un unico fornitore di soluzioni tecnologiche. Per poter offrire questo tipo di servizi integrati, il più delle volte è necessario poter contare su una poderosa infrastruttura e su notevoli capacità tecniche, fattori che limiteranno l'entrata di imprese piccole e medie, che da sole non sarebbero capaci di dare risposta a tutti i bisogni.

Una soluzione comune è che l'impresa di servizi subappalti le parti che non riesce a gestire da sè. Un'altra soluzione piuttosto interessante è il 'modello di consulenza a piramide' che propone Daffara (*sustainability of FLOSS-Based economic models*) e che spiegheremo qui di seguito.

Web consigliato

Per ulteriori informazioni sul "modello di consulenza piramidale", potete consultare: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

Come norma generale, si può dire che nel campo dell'assistenza e della manutenzione informatica viene rispettata la regola 80/20: l'80% delle consulenze è facile e può essere risolto immediatamente e senza sforzo. Il 20% restante, al contrario, presenta problemi importanti e richiederà l'80% dello sforzo. Pertanto, una piccola o media impresa di servizi potrebbe caricarsi sulle spalle un elevato numero di clienti, facendosi carico dell'80% dei suoi problemi, e facendo pagare una modica quantità per il servizio. Per risolvere il rimanente 20%, avrà bisogno dell'assistenza delle imprese che hanno creato i software, alle quali dovrà pagare più di quello che fa pagare ad ognuno dei suoi clienti, ma meno del totale delle entrate che riceve dall'insieme dei suoi clienti.

Questo modello darà vita ad una collaborazione sostenibile tra le imprese sviluppatrici, verticalmente specializzate, e le imprese fornitrici di soluzioni integrate. Le prime potranno raggiungere un numero di utenti più elevato grazie alle consulenze orizzontali, che supporranno inoltre un'importante fonte di entrate. Le seconde potranno così gestire un'ampia base di clienti ed offrire loro assistenza di alta qualità relativa ad un'ampia gamma di prodotti, mantenendo un business redditizio sempre che la sua base di clienti sia sufficientemente larga.

1.2. Imprese sviluppatrici: creare prodotti o servizi?

Come abbiamo sostenuto in precedenza, un'impresa che desideri concentrare la sua attività sullo sviluppo avrà a disposizione due grandi aree tra le quali scegliere: potrà realizzare **prodotti standard**, creati per essere venduti a mercati di massa (quelli chiamati ***shrink-wrapped***), o potrà realizzare **sviluppi su misura**, specifici per ogni cliente in base ai suoi bisogni.

La prima opzione offre il grande potenziale di generare ampi margini di profitto, sebbene siano difficili da mantenere nel tempo, e presenta delle barriere all'entrata che possono risultare insormontabili. La seconda presuppone un'opzione molto più intensa quanto a manodopera, e con margini di profitto molto inferiori, anche se offre maggiori opportunità di creare delle fonti di entrate costanti nel tempo ed è meno sensibile a cambiamenti dell'ambiente macroeconomico.

Analizziamo in dettaglio gli aspetti che differenziano queste due opzioni.

Lettura obbligatoria

M. Cusumano (2004). *The Business of Software* (cap. 1, "The Business of Software, a Personal View").

Economie di scala e possibilità di grandi margini di profitto

Il processo economico di creazione di software presenta delle peculiarità che vedremo in altri settori e che lo portano ad avere delle **economie positive di scala** enormi.

Da un lato, le compagnie commerciali devono investire grandi quantità di denaro nello sviluppo prima di ottenere una versione commerciale di un prodotto da lanciare, e spesso devono investire altrettanto ogni due o tre anni per mantenere il loro flusso di entrate costante. Se si cerca di realizzare un prodotto standard, questo sviluppo implica un alto rischio, dal momento che non c'è alcuna garanzia di poter recuperare l'investimento attraverso successive vendite. Tuttavia, quando si è in possesso di un prodotto finito, il costo marginale di ogni copia addizionale venduta è prossima allo zero. La prima copia del software è molto cara, ma il resto non costa praticamente nulla.

Questo aspetto permette di sfruttare delle economie di scala, sul lato dell'offerta, enormi, e che si uniscono ad economie di scala altrettanto importanti dal lato della domanda: sia per il tempo che s'impiega ad acquisire sufficiente destrezza nell'uso di un'applicazione, sia per la possibile incompatibilità dei formati, cambiare da un prodotto ad un altro rappresenta un compito complicato e costoso. Di conseguenza, quanto più è grande la base degli utenti di un prodotto, tanto più semplice è che questa base cresca e perduri nel tempo. In questo modo, nel mercato del software si arriva a situazioni nelle quali 'il vincitore prende tutto' (*the winner takes it all*), dal momento che può generare enormi profitti ed allo stesso tempo impedire l'entrata di nuove imprese concorrenti.

Esempi di imprese che realizzano prodotti standard

Tra le imprese che hanno sfruttato queste grosse economie di scala, troviamo alcune tra le più importanti imprese del settore del software, come **Microsoft** che occupa tutti i mercati di sistemi operativi da scrivania ed **Oracle** attraverso l'acquisizione di **PeopleSoft** nel 2005. Esistono tuttavia piccole imprese, le Independent Software Vendors (ISV), che, inserendosi in nicchie specializzate, sono riuscite ad avviare dei business ben riusciti. Possiamo citare, come esempi, il 'Pretty Good Solitaire' sviluppato dalla microimpresa (due lavoratori) **Goodsol Development Inc.** (uno dei solitari che ha riscosso più successo), o l''HomeSite', un editor di HTML sviluppato dalla microimpresa Bradbury Software nel 1995, che fu poi comprato da Allaire Corp. (più tardi Allaire fu acquisita da Macromedia, che nel 2005 fu a sua volta assorbita da Adobe)

Al contrario, un'impresa che si concentri sullo sviluppo fatto su misura non sfrutterà alcuna economia di scala. Ogni nuovo cliente richiederà uno sviluppo specifico e pertanto un investimento in tempo ed energie piuttosto costoso, nonostante questo tipo di imprese tenda a riciclare sviluppi precedenti, qualora gli sia possibile.

1.2.1. Necessità d'investimento iniziale

Nel momento in cui si pensa di creare un'impresa focalizzata sull'idea tradizionale di prodotto, sorge un problema importante: **la necessità di un investimento iniziale**. Nelle prime fasi dell'impresa, dedicate allo sviluppo, non ci sarà un flusso in entrata, ma in uscita sì, almeno fino a che le prime versioni del software siano pronte per essere commercializzate. Oltre alle spese direttamente legate allo sviluppo, bisognerà tenere in mente i costi necessari e legati al marketing e alla commercializzazione. Di fronte a questo problema, ci sono due tipi di soluzione: **attrarre investimenti dall'esterno** o **iniziare con un altro tipo di attività** che generi delle entrate sufficienti a permettere lo sviluppo parallelo del prodotto.

Le **imprese che offrono sviluppi su misura** implicano **un rischio molto minore** e potranno dare avvio alla loro attività con un investimento molto minore (lo sviluppo viene iniziato solo dopo la stipula del contratto con il cliente), potendo così evitare la ricerca di investitori esterni.

Nella letteratura finanziaria si è soliti discutere di più su quelle imprese che si finanziano con investimenti in capitale di rischio, dal momento che sono un caso più stimolante. Questo modo di finanziarsi consentirà una crescita più rapida, essendo uno dei fattori più importanti per il successo secondo Cusumano. (Michael Cusumano, *The Business of Software*)

Riflessione

A questo punto, risulta interessante proporre la seguente riflessione: su quali parametri facciamo affidamento quando ci troviamo a dover giudicare il successo di un'iniziativa imprenditoriale? Gli investitori, così come fanno le pubblicazioni finanziarie, considerano di successo quelle imprese che riescono ad ottenere profitti anno dopo anno, oltre a quelle che dimostrano la propria crescita. Una compagnia la cui dimensione si mantiene invariata nel tempo e che non presenta profitti tra i suoi risultati non attrae l'attenzione né degli investitori né delle pubblicazioni finanziarie. Tuttavia, un'impresa di questo tipo può aver avuto molto successo nella creazione di posti di lavoro e nel loro mantenimento nel tempo, per esempio. Per molti imprenditori, questo può essere l'obiettivo principale.

Ottenere investimenti sufficienti dall'esterno può rappresentare uno scoglio invalicabile, però, anche se al contrario si riesce ad ottenerne, questo porterà certi svantaggi che vanno tenuti in considerazione. La presenza di investitori suppone una pressione sulle decisioni di gestione dell'impresa, e obbligherà l'impresa a generare profitti sufficienti alla restituzione con interessi di quanto investito. Questa situazione porta potenzialmente a delle limitazioni nell'autonomia e nella capacità di prendere decisioni per i fondatori.

Nemmeno l'altra opzione sarebbe così facile. L'impresa dovrebbe orientare la sua attività alla prestazione di servizi, facendo sì che questi generino entrate sufficienti a permettere allo stesso tempo lo sviluppo del prodotto. Come illustreremo più avanti, riuscire a far questo con la prestazione di servizi sarà difficile, dato che il margine dei profitti sarà minore e dato che tanto l'assenza di economie di scala, quanto la presenza di una concorrenza agguerrita, limitano la possibilità di tenere i prezzi dei servizi sufficientemente alti.

In questo campo, il software libero irrompe con le sua caratteristiche, nuove per il contesto, e modifica lo scenario. La possibilità di **tagliare i costi** grazie alla collaborazione di volontari, così come i nuovi schemi di diffusione e di commercializzazione che derivano da questa collaborazione, rappresentano una novità di rottura nel contesto degli scenari appena introdotti: offrono la possibilità di diminuire considerevolmente l'investimento iniziale necessario.

Diremo di più rispetto a questo punto nei moduli che seguiranno.

1.2.2. Mantenimento del flusso in entrata

Senza dubbio, una domanda fondamentale per qualsiasi impresa sarà non solo come ottenere entrate in un determinato momento, ma soprattutto come mantenerle nel tempo. Mentre per le imprese incentrate nella prestazione di servizi la continuità è la norma (in genere, se i clienti sono soddisfatti, richiederanno nuovamente e frequentemente lo stesso servizio), per le imprese incentrate sulla produzione di soluzioni standard il mantenimento di un flusso in entrata costante è di difficile attuazione, e si scontra con alcuni problemi.

1) Cicli del software

Cusumano, in *The Business of Software*, accosta il processo che porta a scrivere un prodotto di software di successo con quello dello scrivere un *best seller*. Riuscirci genera enormi profitti, ma, oltre ad essere difficile, questi profitti vengono creati solo in maniera puntuale. Il ciclo di vita naturale di un prodotto di software commerciale lo porterà, alla sua conclusione, a perdere la sua capacità di generare entrate.

In principio, le prime versioni avranno diversi errori ed il loro funzionamento non combacerà perfettamente con i bisogni degli utenti. Questo permetterà alla compagnia creatrice di mantenere un livello di entrate nel tempo attraverso il lancio di versioni nuove che migliorino progressivamente il prodotto, tanto grazie alla soluzione delle falle, quanto grazie all'informazione di gran lunga maggiore di cui dispongono proveniente dai *feed-back* dei suoi utenti e dei suoi clienti.

Come prevedibile, se le nuove versioni del prodotto apportano miglioramenti sufficienti ed appaiono come più attraenti rispetto alle precedenti, saranno una fonte costante di entrate. Tuttavia, una volta che gli utenti sentono di avere una versione già buona, scompare la molla che li spingeva a pagare per avere una nuova versione. Analogamente, cercare di mantenere costante il livello delle entrate di un *best seller* attraverso dei sequel ha un'efficacia puramente limitata e circoscritta.

Esistono delle strategie in grado di combattere queste tendenze e di far restare il flusso in entrata ad un livello costante: utilizzare le licenze di versioni successive. L'incompatibilità parziale o totale tra versioni successive dello stesso prodotto, unita a campagne di diffusione dell'ultima versione, porta ad una nuova situazione di economie di scala dal lato della domanda a favore delle versioni più recenti, cosa che obbligherà molti utenti a cambiare anche se il prodotto anteriore era già di per sé sufficiente a soddisfarne le necessità.

Tuttavia, per la natura stessa di alcuni prodotti di software, l'aggiornamento costante è un fattore necessario, a causa del rapido mutamento dei bisogni dei loro utenti.

Un esempio chiarificatore: le applicazioni per la contabilità e la tenuta delle tasse

Dato che la legislazione relativa alle imposte ed in materia di lavoro cambia di continuo, gli utenti avranno bisogno di aggiornamenti delle loro applicazioni ogni volta che questo succede, e per cui le entrate possono rimanere costanti nel tempo grazie ad aspetti congiunturali del sistema finanziario e del disegno legislativo.

D'altra parte, una volta che l'idea iniziale è stata sfruttata ed analizzata, si apre il cammino per altre imprese che vogliano produrre un software analogo senza doversi impegnare nella ricerca e nello sviluppo. Se queste riescono a realizzare il prodotto più velocemente, magari rendendolo più semplice e snello attraverso l'eliminazione delle funzioni non necessarie, allora saranno in grado di competere nello stesso mercato ma con un prezzo più allettante. Una volta che nel mercato penetrino abbastanza imprese (con prodotti intercambiabili tra loro -*commoditization*-) si giunge ad una situazione particolare: in assenza di altri fattori di differenziazione, i consumatori compreranno il prodotto più a buon mercato, dando vita ad una situazione di forte concorrenza.

Questo fenomeno è comune a qualunque tipo di prodotto, e dovrebbe essere possibile anche nel software. Tuttavia, certi fattori proteggono le imprese dominanti da questo processo, che in circostanze ideali contribuirebbe ad una maggiore diffusione tecnologica e offrirebbe maggiori benefici agli utenti (anche se rende più difficile per le imprese riportare grandi margini di profitto). Come abbiamo sottolineato in precedenza, esistono delle forti economie di scala dal lato della domanda, per cui non sarà così facile che gli utenti per-

cepiscano i prodotti della concorrenza come reali sostituti. Inoltre, l'uso del formato proprietario crea una situazione di 'prigionia' della quale è difficile liberarsi.

Il software libero sembra una forza positiva che spinge verso una situazione di **beni liberi perfettamente sostituibili**: la comparsa di un prodotto simile che viene distribuito in maniera libera, o anche gratuita, renderà più complesso il mantenere alte le entrate provenienti dalle licenze, e può essere una delle poche maniere per riuscire a rompere l'inerzia instaurata dal software proprietario.

Software libero come tecnologia di rottura

Il termine *tecnologia di rottura*, appiccicatogli nel 1999 da Clayton M. Christensen, fa riferimento ad innovazioni che grazie al loro basso costo ed alle loro prestazioni, o perché si concentrano su un nuovo tipo di clienti, riescono a farsi spazio tra soluzioni già consolidate sul mercato. Il software libero potrebbe essere, visto così, una tecnologia di rottura, data la possibilità di ottenerlo gratis e data la sua capacità di contribuire alla diffusione dell'uso del software attraverso le 'brecce' tecnologiche attuali.

La trasformazione del settore del software in uno scenario dove prendono posto beni sostituibili (*commoditization*), anche se limiterebbe grandemente le possibilità di tenere alti i profitti mediante il sistema delle licenze, potrebbe schiudere nuovi mercati, generando un ecosistema di necessità attorno al nuovo prodotto sostituibile e di facile adozione.

2) Dipendenza dai cicli economici

Le compagnie di software tradizionali, concentrandosi sul prodotto, possono sì generare enormi profitti, ma possono anche soffrire di enormi perdite nei cicli economici sfavorevoli. Anche se contavano su business e prodotti consolidati, dal 2000 al 2002 molte compagnie di software hanno perso tra l'80 e il 90% del loro valore, e perfino Microsoft soffrì di una perdita pari a due terzi del suo valore (Michael Cusumano, *The Business of Software*).

Nei periodi economici sfavorevoli il consumo scenderà, ed i prodotti di software saranno tra i primi a risentire degli effetti. Gli utenti smetteranno semplicemente di acquistare software, e le imprese di prodotti che si dedicano esclusivamente a questa attività ne risentiranno enormemente. È per questo che è difficile trovare un'impresa di prodotti pura, dato che la sicurezza dei suoi flussi in entrata sarebbe troppo discrezionale ed a rischio, a causa dei naturali ed inevitabili periodi di crisi del ciclo.

Sebbene qualunque attività economica risente di un tipo di scenario di questo tipo, le imprese incentrate sui servizi saranno più capaci di mantenere stabili le loro entrate, visti i contratti a lungo termine sui quali possono fare affidamento e visto che i loro clienti saranno in maggioranza altre imprese, le quali continueranno comunque ad aver bisogno di supporto relativo alla loro infrastruttura informatica. In svariate occasioni queste infrastrutture consentono all'impresa cliente di funzionare con più efficienza, facendo crescere quindi le sue possibilità di sopravvivenza in periodi di crisi, ed è questa la ragione per la quale non ridurranno i costi destinati ai servizi relativi alle nuove tecnologie.

1.3. Modelli ibridi

In realtà esiste una grande varietà di modelli ibridi, che combinano la vendita di prodotti standard con la prestazione di servizi, cercando di rendere compatibili le due attività. Si può pensare che il grado in cui un'impresa si dedica più ai prodotti o più ai servizi è indicativo del suo ciclo di vita, ed esiste una tendenza generalizzata di transizione verso i servizi.

Esempio di impresa ? modello ibrido

Pensiamo ad un'impresa che inizia con un modello puro di prodotto. Pensiamo poi che realizzi ottime vendite ed ottimi profitti, ma che capisce che sarà difficile mantenere questo livello di entrate. Per garantirsi una continuità, o in risposta a periodi economici di crisi, potrà iniziare a firmare contratti di servizio con alcuni dei suoi clienti, osservando così una riduzione considerevole nel ritmo di crescita dell'impresa, ma ottenendo una maggiore stabilità nel lungo periodo. Alla fine, può essere che l'impresa finisca per porre tutto il suo peso sul piatto dei servizi, essendo il mercato già saturo del suo prodotto originario.

Certamente questo non è che un esempio teorico, e molte imprese non arriveranno nemmeno alla fine di questo ciclo.

D'altro canto, però, la transizione verso i servizi non è così facile e può arrecare dei danni se non viene fatta con attenzione. Adottare un modello ibrido in risposta ad un momento di crisi, senza considerare attentamente la propria strategia di business, può portare ad un'impresa di prodotti un'infinità di problemi.

In periodi di entrate stentate, l'impresa può cedere alle pressioni di alcuni clienti che chiedono di sviluppare aggiornamenti molto specifici e però di difficile integrazione con il prodotto standard principale. Se questa pratica prende piede e l'impresa pretende di mantenere le proprie entrate attraverso la vendita del prodotto standard, la stessa può imbattersi in serie difficoltà circa la compatibilità delle nuove versioni e degli adattamenti personalizzati con il modello standard. L'opera di depurazione e di sviluppo si moltiplica, ed in certe occasioni può portare l'impresa a spendere più di quello che guadagna.

1.4. Software come servizio

Il concetto di "*software as a service*" (SaaS), o "**software come servizio**", prende le sue origini nel 1999 e nasce come un nuovo modo di implementare il software, concentrandosi sulle sue funzioni.

La visione più semplificata di questo concetto ci riconduce al fatto che, per gli utenti, il software diventa importante secondo la misura in cui si permette loro di risolvere un problema; cioè, nella misura in cui presta loro un servizio.

Il bisogno di acquisire un prodotto di software, quello di poter contare su un'infrastruttura di hardware e di software ad esso legata, ed il necessario lavoro di installazione e supporto che porta con sé non sarebbero altro che un disturbo per l'utente finale, secondo questo schema.

In un modello di software come servizio tutti questi inconvenienti scompaiono, ed il software passa dall'essere un prodotto che può essere comprato ad essere un servizio che può essere prestato. In questo senso, è importante distinguere tra le imprese di servizi delle quali abbiamo già parlato e che si dedicano a **prestare servizi sul software** (installazione, manutenzione, ecc.) e questo nuovo modello che si dedica a **prestare il software come un servizio** (prestazione concreta di un determinato software).

Per portare a compimento quest'idea, l'impresa fornitrice si farebbe carico dell'intera infrastruttura necessaria, fornendo il software richiesto ed offrendo il servizio via web. La presenza di un'infrastruttura di comunicazione sufficientemente potente è necessaria, ma il resto dei requisiti tecnologici dal lato del recettore del servizio diminuisce, consentendogli di concentrarsi unicamente sulle funzionalità offerte.

Prestazione di software come servizio

Sempre più imprese stanno usando questo modello per offrire software alle grandi imprese, come 37signals con Basecamp (strumenti di gestione di progetti), ed il popolare Salesforce.com (CRM, *customer relationship management*) che permette la personalizzazione del software in funzione dei bisogni del cliente.

Il modello di *software come servizio* rappresenta un'opzione dal basso costo per offrire del software alle imprese, rispetto alla vendita di prodotti tradizionale. Da un lato, i clienti risparmiano molto quanto alla manutenzione della sua infrastruttura tecnologica. Dall'altro, le imprese potranno fissare dei prezzi più bassi combinando le entrate ricorrenti derivate dalla prestazione di un servizio ed approfittando allo stesso tempo di un unico sforzo per garantire un servizio ad un gran numero di clienti.

Software via web

Troviamo anche numerosi esempi di software via web orientato ai consumatori privati, anche se questa tendenza è stata chiamata "Web 2.0". Molte hanno avuto un grande successo, come le numerose applicazioni di Google o E-bay.

La presenza di software libero così come di SaaS sta minacciando i venditori di software tradizionali, che sentono una forte pressione con l'entrata di questi nuovi concorrenti e dovranno per forza abbassare i prezzi dei loro prodotti.

I fornitori di software come servizio, inoltre, hanno molto da guadagnare dall'utilizzo del software libero. Da una parte, utilizzandolo per le loro infrastrutture di software riporteranno un notevole risparmio sui costi per licenze e sviluppo. Dall'altra, alcune stanno usando delle applicazioni libere, con licenza GPL, come base per sviluppare le loro applicazioni critiche per il business, e mantenendo invece chiuse le modifiche apportate, per garantirsi una protezione sull'unicità del proprio business. Si inseriscono, così facendo, in un vuoto che la GPL contiene: le modifiche del codice devono essere ridistribuite solo se si ridistribuisce il programma. Nel caso del software come servizio il codice non viene ridistribuito, è solo la funzionalità ad essere ridistribuita, e pertanto l'impresa non ha nessun obbligo di condividere i suoi aggiornamenti.

2. Imprese dominanti del settore

Come abbiamo visto, dirigere un'impresa verso prodotti o verso servizi darà luogo a dinamiche imprenditoriali molto diverse, ma entrambe possono risultare lucrative. Detto questo, mantenere attive imprese di prodotti pure sarà comunque estremamente difficile, e le barriere all'entrata resteranno enormi.

Nel sondaggio 'Software 500' del 'Software Magazine' (www.softwaremag.com), che ogni anno elabora una classifica delle prime 500 imprese di software commerciali per entrate, si può notare che tra le compagnie più lucrative si trovano entrambi i tipi di imprese descritti.

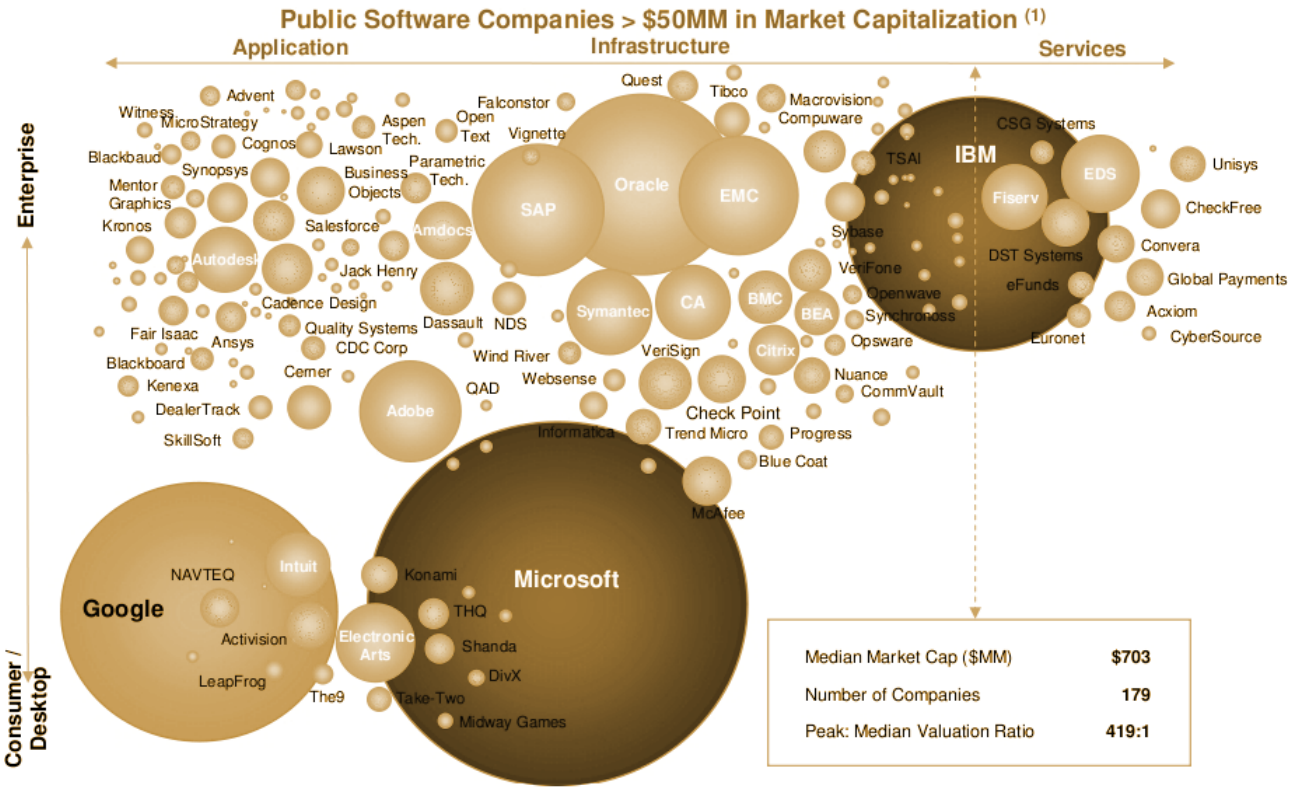
Tuttavia, delle prime venti solo quattro hanno un focus marcato sul prodotto, con meno del 30% della loro attività dedicata ai servizi: Microsoft Corporation, Oracle, SAP, e Symantec, che rappresentano prodotti leader nei rispettivi settori, orientati a clienti imprese ed a mercati di massa (sistemi operativi di desktop, basi di dati, ERP e sicurezza, rispettivamente).

Due imprese hanno un bilancio equilibrato al 50% tra prodotti e servizi, Lockheed Martin Corporation e EMC Corporation. Delle restanti, dieci dichiarano di avere come settore di business principale l'integrazione, la consulenza ed i servizi di subappalto, mentre il resto, nonostante si occupi dello sviluppo di prodotti specifici, ottiene le sue entrate direttamente dalla prestazione di servizi legati ad essi.

	Company	Website	Revenue from software /services ($ million)	Revenue growth (%)	Services as a %	No. employees	Software sector
1	IBM	www.ibm.com	$66,451.00	3.0%	72.6%	394,540	Middleware/Application server/Web server
2	Microsoft Corporation	www.microsoft.com	$39,317.00	9.0%	NA	71,000	Operating systems
3	EDS	www.eds.com	$21,268.00	8.0%	100%	118,500	Outsourcing services
4	Hewlett-Packard Company	www.hp.com	$16,918.00	2.0%	92.3%	156,000	Systems integration services/ IT consulting
5	Accenture	www.accenture.com	$16,646.40	7.0%	100.%	140,000	Systems integration services/ IT consulting
6	Computer Sciences Corporation	www.csc.com	$14,615.60	4.0%	NA	79,000	Systems integration services/ IT consulting
7	Oracle Corporation	www.oracle.com	$14,380.00	22.0%	19.7%	56,133	Databases
8	SAP	www.sap.com	$12,309.70	23.0%	29.2%	39,355	ERP (Enterprise Resource Planning)
9	Cap Gemini	www.capgemini.com	$10,158.60	23.0%	NA	67,889	Systems integration services/ IT consulting
10	Hitachi	www.hitachi.com	$9,019.20	5.0%	85.4%	356,000	Storage management
11	Lockheed Martin Corporation	www.lockheedmartin.com	$8,992.00	10.0%	51.2%	140,000	Industrial vertical applications
12	Science Applications International Corporation (SAIC)	www.saic.com	$7,775.00	8.0%	NA	43,600	Systems integration services/ IT consulting
13	NTT Data Corporation	www.nttdata.co.jp	$6,685.80	4.0%	7.9%	21,308	Systems integration services/IT consulting
14	EMC Corporation	www.emc.com	$6,014.50	16.0%	51.2%	31,100	Middleware/Application server/Web server
15	Affiliated Computer Services, Inc.	www.acs-inc.com	$5,353.70	23.0%	NA	58,000	Outsourcing services
16	LogicaCMG plc	www.logicacmg.com	$5,221.40	65.0%	NA	40,483	Systems integration services/ IT consulting
17	Unisys Corporation	www.unisys.com	$4,917.20	3.0%	NA	31,500	Systems integration services/ IT consulting
18	Sun Microsystems, Inc.	www.sun.com	$4,697.00	19.0%	100.%	34,400	Middleware/Application Server/Web Server
19	SunGard Data Systems, Inc. Pvt	www.sungard.com	$4,212.00	8.0%	91.9%	16,600	Financial applications
20	Symantec Corporation	www.symantec.com	$4,143.40	60.0%	3.3%	17,396	Security tools/ Systems

Prime 20 imprese nel settore del software e loro attività principale (elaborata a partire dallo sutdio "Software 500" del 2007. http://www.softwaremag.com/SW500/)

Nell'immagine che segue possiamo vedere un quadro del posizionamento attuale di queste e di altre imprese di software, quanto al loro focus (applicazione, infrastruttura, servizi) ed al tipo di clienti cui si rivolgono (imprese o consumatori individuali).



Posizionamento delle principali imprese di software (con capitalizzazione di mercato superiore a 50 milioni di dollari e quotazione in borsa). John Prendergast (2008). "Can Xensource, MySQL or Jboss tell you anything about your company's prospects?". *Open Source Business Conference.* Disponibile su: *http://http://akamai.infoworld.com/event/osbc/08/docs/CEO-CMO-Prendergast.pdf*)

3. Marketing nell'impresa: a chi vendere?

Finora abbiamo esaminato vari aspetti della natura principale di un'impresa di software e la definizione delle sue attività principali. Tuttavia, un altro aspetto fondamentale che non va tralasciato in qualunque tipo di impresa è che cosa vendere e a chi rivolgersi.

3.1. Mercati di nicchia e mercati di massa

Per qualsiasi impresa che gode di grandi economie di scala, come nel caso delle imprese di prodotti di software, quanto più ampia è la sua base di utenti, tanto maggiore sarà il suo margine di profitto. Pertanto, la situazione in apparenza più redditizia sarebbe quella di rivolgere i suoi prodotti ai mercati di massa.

Tuttavia, una strategia come questa può presentare diverse difficoltà: il mercato di massa sarà più studiato, controllato e saturato dalle grandi imprese. Per una compagnia che si sta facendo largo da poco sarà estremamente difficile arrivare a fare concorrenza alle imprese già mature e che dominano il settore, che oltretutto possiedono buone e consolidate capacità nel marketing e nella diffusione.

Sembra più facile rispondere ai bisogni dei **mercati di nicchia**, che, date le loro dimensioni, non fanno parte del target delle grandi imprese. Per le grandi compagnie, i ricavi potenziali provenienti da questi mercati sono troppo bassi, visto il numero di clienti piuttosto ridotto, ma per le piccole imprese sono più che sufficienti. Il numero di nicchie possibili è enorme, sfaccettato in molte sfumature in base alle quali si può segmentare un mercato. La domanda chiave in questo senso sarà quanti consumatori sono presenti in una determinata nicchia, e la risposta permetterà di calcolare il potenziale volume del business, e quindi il volume di spese che l'impresa può permettersi di avere.

Il mondo del software offre delle opportunità più interessanti rispetto ad altri prodotti tangibili presenti nei mercati di nicchia, data l'assenza di barriere geografiche che scompaiono grazie ad internet. Una nicchia scoperta in una data area geografica potrà essere esportata con uno sforzo relativo ad altre zone che abbiano necessità simili, o si amplierà addirittura da sola, senza necessità di una regia da parte dell'impresa.

Nel momento in cui si arriva a dover creare i prodotti per i mercati di nicchia, è fondamentale conoscere approfonditamente la nicchia stessa. Oltre a poter contare su conoscenze tecniche, è necessario che l'impresa possieda una buona panoramica delle attività, delle priorità e della maniera in cui funziona la nicchia particolare che vuole attaccare. Secondo la regola di Eric Raymond, 'ogni buon lavoro nel campo del software comincia a partire dai bisogni personali del programmatore' ('*Every good work of software starts by scratching a developer's personal itch*'), è utile partire da una nicchia della quale si fa parte, per poter comprendere meglio quali necessità e quali problematiche vi esistono.

Conoscenza dell'ambiente

E' il caso della nicchia degli sviluppatori di software: è stato sfruttato in lungo e in largo, dato che tutti i programmatori sono allo stesso tempo creatori ed utenti, e potendo quindi fare affidamento su una conoscenza profonda dei bisogni e dei problemi del settore.

Un altro fattore importante da tenere in debita considerazione è se si decide di vendere il prodotto a grandi imprese, a piccole imprese o a persone individuali.

Le imprese di servizi dovranno puntare a grandi imprese, amministrazioni pubbliche o altre organizzazioni, dato che i consumatori privati molto difficilmente saranno disposti a pagare per servizi orientati al software. Le imprese di prodotti, invece, potranno scegliere, in base alle caratteristiche dei loro prodotti ed alla loro strategia imprenditoriale, quali clienti far rientrare nel loro target. Le grandi imprese possono sembrare più appetibili, dal momento che saranno più orientate a spendere per un prodotto di software, ed inoltre contribuiranno a generare entrate attraverso i servizi.

In una grande impresa, si pagherà per un prodotto di software, ma anche per l'assistenza relativa al prodotto, per la formazione, per l'installazione e per l'integrazione col resto dei sistemi. Le imprese che comprano del software in genere pagano tra il 15 e il 25% del prezzo della licenza attraverso la manutenzione annuale (Dan Woods, Gautam Giuliani, 'Open source for the enterprise'). Spesso richiedono anche degli sviluppi realizzati su misura, per adattare il prodotto alle loro necessità. In questo modo, la grande impresa aiuterà l'impresa di software a generare entrate dal lato dei servizi, dandole una certa garanzia di continuità. Queste entrate, però, saranno dovute ad un'intensità maggiore di lavoro di manodopera, e sarà necessario, quindi, gestire l'impresa in maniera attenta, per poter tenere sotto controllo il fatto che i costi della prestazione del servizio non superino le entrate derivate dal servizio stesso.

Lettura consigliata

D. Woods; G. Guliani (2005). *Open source for the enterprise: managing risks, reaping rewards*. O'Reilly Media, Inc.

Da parte loro, i servizi di assistenza vengono prestati relativamente a versioni specifiche del prodotto, per cui operare in questo senso può aiutare anche a generare entrate sotto forma di licenze per versioni successive: se anche il cliente non avesse interesse ad acquisire la nuova versione, si vedrà obbligato a comprarla, dato che l'assistenza sulla vecchia non viene più realizzata.

Uno svantaggio possibile è che una grande impresa di un certo peso non sarà esattamente propensa ad acquistare servizi da una impresa piccola e nuova. Uno dei fattori essenziali quando si vende un servizio di questo tipo è la repu-

tazione, seguita dalla fiducia che ispira la compagnia che fornisce il servizio, e per questo le imprese minori o di recente creazione reperiranno clienti più facilmente nel proprio ambiente, cioè tra le imprese piccole e medie.

3.2. I patroni dell'adozione tecnologica e l' 'abisso'

Individuare una nicchia di mercato ed elaborare un buon prodotto che soddisfi i bisogni del potenziale gruppo di utenti non è sufficiente per ottenere che sia accettato. Per riuscire ad introdurre nel mercato un nuovo prodotto o un nuovo servizio, sarà fondamentale tener conto di come sono i patroni dell'adozione tecnologica all'interno di un gruppo di persone.

I libri di marketing riportano tradizionalmente un modello di adozione costruito sulla base di una curva di Gauss con quattro gruppi di utenti:

- **Innovatori e primi a provare (*early adopters*):** a loro piacciono la tecnologia e l'innovazione. Adotteranno un certo prodotto spesso solo perché è nuovo.

- **Maggioranze attente**: adotteranno una tecnologia solo se le aiuta a risolvere problemi concreti.

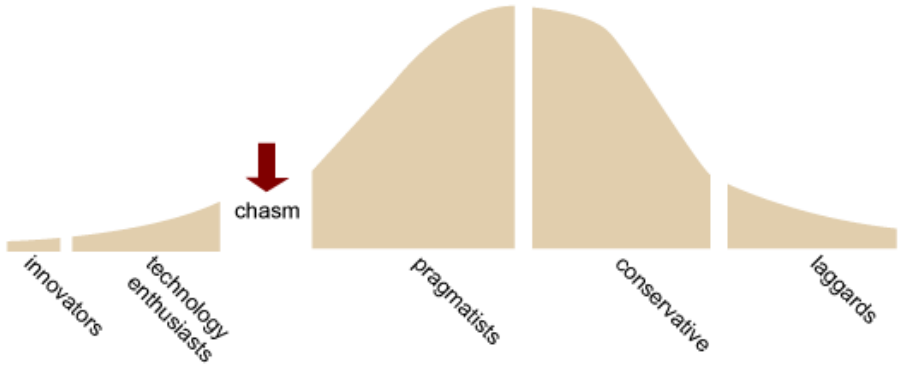
- **Maggioranze distratte**: cercano di evitare le nuove tecnologie.

- **Ritardatari**: saranno gli ultimi a provare un prodotto nuovo, e può anche essere che non arrivino mai a provarlo.

La curva rappresenta due idee chiave: la prima è che le due categorie intermedie includono la grande maggioranza dei **clienti potenziali**, la seconda è che si può riuscire ad attrarre i gruppi di clienti solo in ordine, da sinistra a destra (i 'primi a provare' adottano un prodotto solo dopo che l'avranno fatto gli innovatori, le maggioranze attente se lo avranno fatto 'i primi a provare', le maggioranze distratte solo se lo avranno fatto quelle attente, e i ritardatari solo dopo che l'avranno fatto le maggioranze distratte).

Geoffrey Moore, nel suo libro *Crossing the Chasm* ,rinomina i gruppi, riferendosi a loro come **entusiasti tecnologici**, **visionari**, **pragmatici**, **conservatori** e **scettici**, ed argomenta che questa teoria ha una falla, dal momento che la transizione tra gli entusiasti e le maggioranze pragmatiche non è senza soluzione di continuità, e per questo non è facile da raggiungere. Le maggioranze attente non adotteranno soluzioni che non siano state abbondantemente verificate, ed implementeranno solo quelle sulle quali possono ottenere buone referenze da parte di altri pragmatici. Per questo a volte attrarre queste mag-

gioranze può sembrare una missione impossibile. Per Moore esiste un abisso (*chasm*) tra i due gruppi, ed in base a questa considerazione ridisegnò la curva in questo modo:



Curva dell'adozione tecnologica secondo Moore

Gli innovatori e gli entusiasti tecnologici possiedono un'**alta propensione al rischio** ed un'alta tolleranza rispetto ai difetti delle nuove tecnologie, dal momento che possono fare affidamento su una provata abilità tecnica. Questi utenti adotteranno una tecnologia specifica sulla base delle funzionalità per le quali stanno cercando delle novità. Le maggioranze attente e quelle distratte (pragmatici e conservatori) hanno una **bassa propensione al rischio**, e saranno sì interessate a comprare un prodotto che permetta loro di aumentare la propria produttività, ma pretendono allo stesso tempo che il prodotto sia già maturo e testato.

In questo modo un prodotto innovativo potrà avere un grande successo tra gli innovatori e gli entusiasti tecnologici, ma se la compagnia fornitrice vuole ampliare la sua base di clienti dovrà mettere in moto una campagna di marketing diversa, dando enfasi non più alle funzionalità concrete e alle migliorie che offre il prodotto, ma bensì al tentativo di trasmettere fiducia quanto al prodotto, enumerando casi di successo e numero di utenti che l'hanno già adottato.

Attrarre i primi clienti nel gruppo dei pragmatici e renderli felici diventa fondamentale, ma risulta molto difficile a causa del circolo vizioso che si viene a creare: nessuno dei pragmatici adotterà una soluzione che altri pragmatici non abbiano già provato.

Si può lavorare sul concetto di fiducia offrendo soluzioni complete, che includano la manutenzione, l'assistenza e la formazione, di modo da riuscire ad attrarre quei clienti che sono più sensibili alla maturità ed alla facilità di uso del prodotto. I primi clienti di questo gruppo dovranno essere trattati con molta attenzione, senza lesinare su tempi e denari spesi, dal momento che saranno il punto di riferimento per tutti gli altri. Una volta che si sia riusciti ad attrarre alcuni pragmatici (di riferimento), attrarre anche il resto sarà molto più facile, e, quando i pragmatici avranno adottato la soluzione, anche i conservatori ne

seguiranno l'esempio senza che vi sia bisogno di particolari campagne di marketing. Concentrarsi su innovatori ed entusiasti -supponendo che nonostante sia un mercato potenzialmente ridotto, sarà comunque sufficiente per una piccola impresa- può essere rischioso, dato che il gruppo che li include, per sua natura, non è un gruppo stabile, e non esiterà ad abbandonare il prodotto non appena smetta di essere la novità del momento.

Questa curva di adozione segnerà anche il ciclo di vita del prodotto, le sue dinamiche di sviluppo e le sue pratiche di marketing. L'impresa che commercializza deve aver chiara la fase nella quale si trova, e quali sono i suoi clienti in ogni determinato momento, dal momento che ogni gruppo si sente attratto da fattori molto diversi. Mentre includere molte funzioni nuove e realizzare prodotti molto innovativi costituiranno la massima attrazione per gli innovatori, i conservatori avranno bisogno che il prodotto, molto più semplicemente, funzioni in alcuni contesti specifici, e che lo faccia sempre alla stessa maniera. Ogni cambiamento implica una difficoltà che saranno disposti a superare solo se porta con sé la soluzione a qualche problema nel quale si sono imbattuti.

4. Funzione del prodotto: cosa vendere?

Considerare con attenzione la natura del prodotto da sviluppare è molto importante. Una delle domande cui bisognerà dar risposta è se si desidera che il prodotto sia il leader del settore, un prodotto che insegue o un prodotto complementare.

Anche se in teoria essere il leader del settore può sembrare l'obiettivo più appetibile, può non essere il più efficiente. Quando individua l'assenza di una funzione in un prodotto di larga adozione, un'impresa si trova di fronte a due possibilità: sviluppare la propria versione, includendo la funzione che manca e cercando così di competere con il leader, o realizzare un complemento al leader che ne completi le possibilità.

La prima opzione risulterà assai complicata, e può finire in un niente di fatto molto facilmente, dal momento che necessita di un investimento considerevole non solo per il nuovo sviluppo, ma anche per la campagna di marketing e per quella successiva di vendita. Nella seconda, oltre a poter sviluppare il prodotto in meno tempo, il lavoro di marketing sarà già stato fatto, in larga misura, dal leader, per cui sarà molto più facile che il complemento trovi degli acquirenti. Inoltre, gli acquirenti conservatori (la maggioranza) saranno molto più disposti ad includere un complemento ad una soluzione già nota e testata piuttosto che a cambiare la propria tecnologia ed il proprio fornitore. Un pericolo comune sarà che l'impresa leader decida di dotarsi della funzione sviluppata, eliminando di fatto la necessità per il cliente di comprare il complemento. Sotto questo aspetto, la relazione intrattenuta con l'impresa che sviluppa il prodotto di base sarà fondamentale.

È importante, pertanto, definire bene il ruolo che avranno le altre imprese attive nel settore. Quali si comporteranno come concorrenti dirette, quali come collaboratrici e quali, benché siano presenti nello stesso settore, non entreranno in concorrenza con il nostro prodotto, in quanto in possesso di una specializzazione ben definita. Segmentando le nicchie ed offrendo una buona differenziazione, si riesce ad evitare la concorrenza diretta di imprese forti, e l'esistenza di imprese che elaborano prodotti o servizi complementari può essere un importante fattore di successo.

Quando si va a decidere il posizionamento di un prodotto, sarà importante tener conto anche della piattaforma per la quale si sviluppa il prodotto, ovvero, che insieme di software di base sono necessari per il funzionamento del prodotto. Pensiamo, ad esempio, alla scelta del sistema operativo e della tecnologia attraverso i quali l'applicazione opera. Questa decisione influirà molto

sulla definizione della nicchia di mercato nella quale si farà ingresso, e su che tipo di clienti si orienterà, ma sarà importante in ugual misura per la definizione della relazione con alleati e concorrenza.

Un'applicazione disegnata per funzionare in una determinata piattaforma sarà un'applicazione complementare alla suddetta piattaforma. Se si tratta di un insieme di software già radicato nel mercato e molto diffuso, si allarga anche il mercato potenziale di clienti, ma si riducono le possibilità di trovare alleati tra gli sviluppatori della piattaforma. Il valore di queste piattaforme sarà dato soprattutto dal numero e dalla varietà delle applicazioni che le percorrono, per cui un'impresa che cerchi di prendere il posto di leader della piattaforma sarà molto interessata allo sviluppo di applicazioni complementari, e sarà, pertanto, un'alleata più fedele.

Tuttavia, benché più difficile, può essere più interessante collocarsi nel ruolo di leader di un determinato settore. La domanda, in questo caso, sarà se si desidera creare una categoria di prodotto nuova per una nicchia che non viene ancora sfruttata, o se si vuole andare a scalzare un altro prodotto già esistente.

La segmentazione ed i potenziali clienti

Per un'impresa modesta l'unica possibilità è quella di segmentare il mercato, fino a trovare una nicchia specifica nella quale posizionarsi. Può essere difficile arrivare ad essere il leader delle applicazioni usate per pianificare le risorse imprenditoriali (ERP, *enterprise resource planning*), ma può essere semplice se si sviluppa un ERP per le piccole e medie imprese. Certamente, quando aumenta la segmentazione, diminuisce la concorrenza, anche se diminuisce pure la base dei potenziali clienti.

Essere i primi in un determinato mercato darà senza dubbio dei vantaggi quando ci si proponga di diventare leader del settore, così come quando si definiranno gli standard di una tecnologia, ma non dà nessuna garanzia. La prima impresa che sviluppa una tecnologia non sempre diventa il leader del settore. A volte arrivare primi e 'catturare' gli entusiasti della tecnologia può essere una falsa spia di successo, dal momento che il prodotto dovrà essere adottato dalla maggioranza per potersi definire leader del settore. Le decisioni strategiche e tecniche successive saranno decisive per determinare se l'impresa sarà capace di fare tesoro delle economie di scala della domanda e posizionare il suo prodotto come il numero uno del settore.

Per riuscire a penetrare in un mercato che ha già un leader, ci sarà bisogno di campagne di marketing e di vendita che spesso non sono possibili per imprese di creazione recente. Tuttavia, l'uso di un prodotto di software libero, che entri sul mercato a prezzo zero, può essere un agente di rottura sufficientemente potente. Nei prossimi moduli vedremo questa e le altre strategie che il software libero offre per concorrere in mercati diversi.

Riepilogo

I bisogni relativi al software generano molteplici opportunità di business nel corso del suo ciclo di vita, dallo sviluppo vero e proprio ai servizi connessi, come l'installazione, la migrazione o la formazione degli utenti.

Il posizionamento dell'impresa è un fattore chiave per determinare le possibilità di successo del business:

- La preferenza accordata alla prestazione di servizi dà luogo ad una cornice economica più stabile nel tempo.

- L'orientamento verso lo sviluppo dei prodotti dà vita ad un'economia del prodotto, più complessa da mantenere in periodi lunghi.

- I modelli ibridi cercano di offrire garanzie di equilibrio ai due modelli precedenti.

- L'irruzione del software come servizio rappresenta una minaccia per i modelli più tradizionali, perché offre una variazione più versatile per i potenziali clienti.

D'altra parte, lo sfruttamento di filoni di mercato che sono vicini e noti può aiutare la strategia imprenditoriale di un business nuovo, così come l'adeguamento del prodotto ai patroni di adozione tecnologica del mercato target.

Infine, sarà necessario stabilire chiaramente la relazione del business con i suoi concorrenti, così come quella del prodotto con quella dei suoi concorrenti. Queste relazioni possono anche favorire l'introduzione del prodotto nel mercato obiettivo.

Bibliografia

Christensen, C. M. (1997). *The innovator's dilemma*. Harvard University Press <http://books.google.es/books?id=SIexi_qgq2gC> [Data di consultazione: aprile 2009]

Christensen, C. M.; Raynor, M. E. (2003). *The innovator's solution*. Harvard University Press. <http://books.google.es/books?id=ZUsn9uIgkAUC> [Data di consultazione: aprile 2009]

Cusumano, M. (2004). *The Business of Software* Free Press. Cambridge: Cambridge University Press. <http://books.google.com/books?id=7KAW-ToDnBAC&dq=the+business+of+software&hl=es> [Consultazione: febbraio 2009]

Daffara, C. (marzo, 2006). "Sustainability of FLOSS-based economic models". *II Open Source World Conference*. Malaga. <http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf> [Data di consultazione: aprile 2009]

McKenna, R.; Moore, G. (2006). *Crossing the chasm* Capstone. <http://books.google.com/books?id=GTwFAQAACAAJ&dq=crossing+the+chasm&hl=es> [Consultazione: febbraio 2009]

Sink, E. (2006). *Eric Sink on the Business of Software* Apress. New Jersey: Princeton University Press <http://books.google.com/books?id=h5IQuengOGIC&dq=eric+sink+business+of+software> [Consultazione: febbraio 2009]

Sviluppare software libero in un'impresa

Amadeu Albós Raya

PID_00145047

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

Attraverso questo modulo ci addentriamo nel mondo della produzione di software libero e nelle sue peculiarità più rilevanti riguardo al prodotto, all'impresa ed alla comunità di utenti.

In un primo momento, analizzeremo lo sviluppo di software libero dal punto di vista del progetto, considerando gli aspetti principali legati alla popolazione ed alla gestione del progetto, così come la partecipazione della comunità di utenti in diversi ambiti.

Mediante il progetto di software libero si realizza la relazione tra impresa e comunità di utenti. Il bilanciamento dei tratti particolari di questa relazione è fondamentale per il conseguimento degli obiettivi del progetto.

Successivamente, descriveremo le caratteristiche che rendono particolare la comunità degli utenti del software libero e la loro gestione da parte dell'impresa. Questa gestione accompagna, essendone complemento, la metodologia di produzione, ed implementa la strategia relazionale già citata.

Alla fine, il modulo si concluderà con la presentazione del caso di studio di un'impresa reale che produce software libero.

Questo modulo si sviluppa come una guida attraverso testi esistenti, il cui obiettivo è di approfondire in dettaglio le peculiarità dei diversi aspetti che si presentano e che sono rilevanti per la produzione d'impresa del software libero.

Obiettivi

Al termine di questo modulo, si dovrà essere in grado di:

1. avere familiarità con la metodologia usata nella produzione di software libero.

2. comprendere l'importanza che riveste la comunità degli utenti nello sviluppo di prodotti basati sul software libero.

3. identificare ed analizzare i fattori rilevanti che influiscono sul successo della produzione di software libero.

4. capire l'importanza che gioca la formulazione precisa di una metodologia che permetta di rendere complementari gli sforzi dell'impresa e della comunità degli utenti.

5. approfondire le implicazioni dirette ed indirette del portare a compimento un progetto di sviluppo basato sul software libero.

1. La produzione di software libero

In questa prima sezione ci concentreremo sulla produzione di software libero dal punto di vista del suo sviluppo e della sua elaborazione, indipendentemente da eventuali modelli di business che lo sfruttino a scopo di lucro.

In diverse materie del master, soprattutto in quelle dedicate alla produzione del software (apice), viene analizzato in dettaglio il processo tecnologico che caratterizza l'elaborazione di software libero.

Questo processo tecnologico è complementare alle metodologie che ci consentono di formulare un progetto di cooperazione sostenibile e duraturo nel tempo. In questo senso, la collaborazione della comunità degli utenti nel progetto di software libero è fondamentale per ottenere una massa critica di utenti che aiuti a realizzare la realizzabilità del progetto.

Di conseguenza, una buona parte di queste metodologie e delle azioni conseguenti sono dirette ad offrire supporto e garanzie alle relazioni tra il progetto e la comunità di utenti. Per renderci conto dell'importanza di questa relazione, è sufficiente dare un'occhiata alle risorse che i progetti di software libero più popolari offrono alle loro comunità di utenti.

Progetti popolari

Ad esempio, OpenOffice.org (http://contributing.openoffice.org/) e Mozilla (http://www.mozilla.org/contribute/).

Per sviluppare questi concetti, nei prossimi paragrafi descriveremo tre punti di vista complementari. Inizieremo presentando alcune idee di base sulla produzione del software libero. Poi forniremo brevemente i principali dettagli sui passi da compiere per mettere in moto un progetto basato sul software libero. Infine, approfondiremo i principali aspetti che caratterizzano la gestione del progetto di software libero.

1.1. La produzione di software libero

La produzione di software libero, allo stesso modo che per qualsiasi altro software, risponde alla necessità di risolvere una problematica tecnologica concreta. concreta[1].

(1)Ad esempio, aggiungere delle funzioni ad un'applicazioneo risolvere errori di funzionamento.

Anche se il processo tecnologico di raffinare e apportare evoluzioni ad un'applicazione di software libero può presentare molte somiglianze rispetto ad un'applicazione basata sul software proprietario, la differenza che le dà

l'apertura del modello le conferisce un funzionamento particolare. Il carattere aperto e di cooperazione della sua produzione incide sulla struttura evolutiva quantitativa e qualitativa, di versione in versione.

Sono molti gli autori che hanno scritto testi sulle particolarità che implica produrre software libero. Dal momento che questo modulo non è destinato a descrivere in maniera esaustiva le suddette particolarità, già ampiamente trattate in altre materie, ci concentreremo solo su alcune, le più rilevanti per il nostro caso.

Per questo motivo, prenderemo spunto da alcuni dei concetti presenti nel saggio La catedral y el bazar, di Eric S. Raymond, nel quale viene fatta un'analisi delle particolarità del software libero, e viene trattato soprattutto il caso GNU/Linux.

Web consigliato

E. Raymod (1997). *The cathedral and the bazaar* (http://www.catb.org/~esr/writings/cathedral-bazaar/).

- **L'origine della produzione**
  A grandi linee, la produzione del software libero nasce a partire dalle necessità proprie dell'utente o dello sviluppatore nella sua attività quotidiana. Questo significa che la collaborazione nello sviluppo del software comincia tramite la ricerca di un problema la cui soluzione ci sembri interessante, rilevante o necessaria.

Origini della produzione

Una buona parte dei fondamenti del software libero si basa sulla pubblicazione degli aggiustamenti o degli sviluppi concreti che realizzano dei lavoratori nell'esercizio quotidiano delle loro attività lavorative.

- **La comunità di utenti**
  La comunità di utenti del software libero, che racchiude tanto utenti finali quanto sviluppatori e programmatori, è la base che dà un senso alla definizione di sviluppo di software libero.
  Trattare gli utenti come dei collaboratori nel progetto di produzione è la maniera più facile per migliorare e perfezionare rapidamente il codice (sempre che la base di collaboratori sia sufficiente).
  Per questo, i collaboratori rappresentano una delle risorse più preziose nello sviluppo dell'applicazione, e pertanto è prezioso anche riconoscere le buone idee e le soluzioni che propongono.

- **Le versioni dell'applicazione**
  Una delle caratteristiche della produzione di software libero è la riutilizzazione e la riscrittura del codice originale, che porta ad ottenere un codice nuovo, senza errori, con nuove funzionalità o con un miglior rendimento (tra le altre cose).
  Dall'altra parte, nei progetti di sviluppo di software libero si promuove la liberazione rapida e frequente del codice, così da rendere l'attività del progetto dinamica ed incessante.

- **Il coordinamento della produzione**
  La persona (o le persone) che coordina il progetto deve saper gestire il potenziale globale della comunità di utenti, dirigendo l'evoluzione del progetto senza coercizioni, ma sfruttando i mezzi e le sinergie che offre una rete come Internet.

L'eredità del codice dell'applicazione e della sua gestione del coordinamento è importante per il futuro del progetto di sviluppo del software libero. La scelta del successore chiamato a controllare ed a gestire la produzione non deve essere lasciata al caso.

1.2. Il progetto di software libero

A complemento delle considerazioni tecnologiche e funzionali delle applicazioni basate sul software libero, uno degli obiettivi primordiali di ogni progetto di software libero è la diffusione dell'applicazione o l'ottenimento di una massa critica di utenti.

È poco utile per il futuro del progetto, quindi, che il codice generato, anche se può risolvere problemi o carenze concrete, non sia conosciuto ed utilizzato dagli utenti potenziali. Inoltre costituisce un obiettivo necessario per il suo mantenimento ulteriore e per la sua evoluzione nel tempo. Nel caso del software libero, il compimento di questo aspetto è fondamentale per la creazione di una comunità di utenti stabile e duratura.

Esistono diverse guide che, in maggior o minor misura, apportano i concetti necessari alla creazione ed alla gestione di progetti basati sul software libero. In questa parte svilupperemo la questione attraverso l'articolo Free Software Project Management HOW TO di Benjamin Mako, che passa in rassegna le principali peculiarità del progetto dal punto di vista pratico.

Web consigliato

B. Mako (2001). *Free Software Project Management HOW TO* (http://mako.cc/projects/howto).

Messa in moto

Prima di lanciare il progetto basato sul software libero, è molto importante disegnare una struttura solida che formi una base per poter poi sopportare il processo di sviluppo con garanzie sufficienti.

In generale, la struttura basica del progetto deve dare risposta ai seguenti aspetti:

- La necessità di creare un progetto nuovo, a causa di idee e obiettivi propri o dell'esistenza di progetti simili.
- La definizione delle principali caratteristiche dell'applicazione (funzionalità, licenza, numerazione, ecc.).
- L'infrastruttura di base a supporto della diffusione del nuovo progetto e della collaborazione nel suo processo di sviluppo (pagina web, posta, ecc.).

Gli sviluppatori

Una volta messo in moto il progetto, il nostro secondo obiettivo sarà l'integrazione, il coinvolgimento e il consolidamento degli utenti e degli sviluppatori dell'applicazione. Quanto a questi ultimi, dovremo essere in grado di mettere in atto politiche e strategie che ci consentano di definire e di strutturare la loro collaborazione.

Le politiche di cooperazione devono dare soluzione principalmente a due obiettivi concreti:

- Il coordinamento della produzione interna e di quella esterna, che include la delega delle responsabilità e i protocolli di accettazione dei contributi.

- La gestione della produzione, come, ad esempio, la struttura dei rami dell'attività di sviluppo e dei magazzini associati.

Gli utenti

Nei prodotti basati sul software libero, spesso gli utenti sono anche sviluppatori (e viceversa). Uno dei principali obiettivi che dobbiamo tenere in considerazione sono le prove o i test dell'applicazione, siano essi funzionali, operativi, di qualità, ecc.

L'infrastruttura di supporto

L'attività quotidiana del progetto fondato sul software libero non si potrebbe realizzare senza un'infrastruttura di supporto adeguata agli obiettivi di cooperazione dello stesso.

Nella messa in moto del progetto si sono già compiute le prime azioni per realizzare questo sistema di infrastrutture, ma una volta che il sistema si attiva, è necessario adeguare, migliorare e rinsaldare le risorse esistenti d'accordo con l'evoluzione del progetto.

Risorse abituali

Alcune delle risorse abituali nei progetti di software libero sono: documentazione, mailing list, sistemi di controllo di errori e di versioni, forum, *chat*, *wiki*, ecc.

L'applicazione

Senza dubbio, la componente più rilevante del progetto e quella da cui dipende il resto degli aspetti considerati è l'applicazione. Una delle caratteristiche più rilevanti che deve contenere l'applicazione è che l'utente abbia sufficienti garanzie sul funzionamento di ognuna della versioni che vengono lanciate sul mercato.

Il lancio delle versioni è un tema delicato su cui conviene soffermarsi. A grandi linee, terremo a mente questi aspetti:

- Il controllo delle revisioni in termini di funzionalità e di correzione degli errori (versioni alfa, beta, distribuzione candidata, ecc.).

- Quando lanciare la versione completa, ovvero quando il codice sarà pronto per offrire le garanzie che ci aspettiamo e che si aspettano gli utenti.

- Come lanciare la versione (incartamento, codice fonte, formato binario, ecc.).

Diffusione del progetto

Infine, e come avevamo già anticipato, diffondere l'esistenza del progetto è importante per il progetto stesso, ma questo compito deve essere riproposto di continuo, nel tempo, se si vuole che le fondamenta si consolidino.

Ad ogni passo avanti del progetto, dovremo pensare a far risaltare il progetto attraverso liste di posta relazionate con il software libero o attraverso Usenet, inserire il progetto in altri portali pubblici (come Freshmeat o Sourceforge), o anche ad annunciare le nuove versioni dell'applicazione nelle liste di posta del progetto stesso.

1.3. La gestione del progetto

In questa sezione approfondiremo gli aspetti della gestione del progetto che, in quanto promotori dello stesso, dovremo tenere in considerazione per dotarlo di garanzie di successo.

I concetti che presentiamo in questa sezione sono complementari a quelli esposti nelle sezioni precedenti, dato che rendono possibile operare in concreto e migliorare le diverse azioni che abbiamo descritto. Per questo è possibile che vi siano delle coincidenze dirette o indirette con questi temi.

Per fare una relazione degli aspetti di base della gestione del progetto basato sul software libero, terremo conto delle considerazioni espresse in *Producing Open Source Software* di Karl Fogel, e nello specifico del quinto capitolo, intitolato 'Money'.

Web consigliato

K. Fogel (2005). *Producing Open Source Sotware: How to Run a Successful Free Software Project* (capítulo 5 "Money"). (http://producingoss.com/en/money.html).

Finanziamento

Le caratteristiche peculiari dei progetti di software libero fanno sì che molti dei contributi siano sovvenzioni informali (ad esempio, quando il lavoratore di un'impresa pubblica gli aggiustamenti che ha apportato al codice nell'esercizio delle sue funzioni).

Nonostante questo, vengono fatte anche donazioni e sovvenzioni che consentono di ottenere entrate dirette per il funzionamento del progetto, ma bisogna tener conto della gestione di questi fondi, dato che una buona parte dell'appoggio che riceve un progetto di software libero si basa sulla credibilità dei suoi partecipanti.

Tipi di partecipazione

Esistono molti tipi (e molte possibili combinazioni) di partecipazione in forma finanziaria ad un progetto di software libero. Inoltre, in questo modello di finanziamento influiscono anche aspetti che esulano dal progetto stesso, ma dipendono dal contorno e dal contesto della messa in opera.

Sommariamente, si può dire che la partecipazione in un progetto di software libero ha una relazione con la collaborazione dei suoi partecipanti, con il modello di business che utilizza l'impresa che lo promuove (nel caso che esista), con le azioni di marketing promosse, e con le licenze dei prodotti coinvolti e le donazioni effettuate.

Contratti indefiniti

Il team di sviluppatori dell'applicazione è molto importante per lo sviluppo del progetto e per la sua evoluzione futura. La stabilità e la permanenza dei partecipanti nei loro ruoli di responsabilità consente di rinsaldare le basi e la credibilità del progetto di fronte alla comunità degli utenti.

Un progetto stabile

La credibilità è fondamentale per tutti gli attori direttamente o indirettamente legati al progetto, dato che non è possibile trasferirla agli eventuali sostituti, e che la sua perdita può compromettere il futuro dell'applicazione e del progetto stesso; pertanto, dobbiamo prendere le giuste iniziative per controllare e gestire attivamente il progetto.

Decentramento

Una delle caratteristiche più rilevanti (e desiderabili) delle comunità di utenti di software libero è la distribuzione e quindi il decentramento delle decisioni che vengono prese all'interno del processo.

L'organizzazione del progetto, quindi, deve tener conto di questa struttura, per poterla sfruttare per stimolare la motivazione e rinforzare la comunità degli utenti dell'applicazione, di modo da far sì che il consenso sorga dalla stessa interazione tra i membri.

Trasparenza

L'aspetto precedente, legato al decentramento, ci dà un'idea della trasparenza che deve ergersi sovrana nella relazione tra progetto e comunità.

Tanto gli obiettivi del progetto quanto le linee di evoluzione dell'applicazione devono essere chiari e ben noti a tutti coloro che sono coinvolti nei processi. Per questo, l'influenza del promotore sull'evoluzione futura deve manifestare i tratti di un comportamento onesto e trasparente, tali da garantire la credibilità del progetto credibilidad del proyecto[2].

(2)Ad esempio, il manifesto di Openbravo (http://www.openbravo.com/es/about-us/openbravo-manifesto/).

Credibilità

La credibilità del progetto (insieme con la credibilità dei suoi membri) è comparsa spesso in molti degli aspetti che abbiamo considerato finora. La sua rilevanza è molto legata alla comunità degli utenti del software libero e dà vita ad una condizione importante per il mantenimento nel tempo.

Il denaro o la posizione gerarchica non sono in grado di infondere la credibilità necessaria alle azioni di ogni membro in ogni momento. Pertanto la metodologia, i procedimenti o i protocolli stabiliti, o il funzionamento o la gestione operativa devono essere gli stessi per tutti senza eccezioni.

Contratti

.L'assunzione dei lavoratori è un aspetto che va curato nei minimi particolari nei progetti di software libero, a causa delle sue ripercussioni sulla struttura e sul funzionamento. Bisogna stare attenti a che tutti i dettagli ed i processi relazionati con le assunzioni restino aperti e trasparenti.

Per questo, è importante discutere questi cambiamenti con la collaborazione della comunità degli utenti, fino al punto che in alcuni casi può essere preferibile assumere direttamente sviluppatori della comunità con permessi di scrittura nel magazzino ufficiale (*committers*).

Risorse

Il progetto di software libero non si basa solo sull'evoluzione e sul mantenimento del codice di un'applicazione basata sul software libero, ma deve comprendere anche altri aspetti di supporto complementari

Risorse complementari

È questo il caso della gestione della qualità del codice prodotto, della protezione legale dei contributi, della documentazione e dell'utilità dell'applicazione e della consegna di risorse infrastrutturali per la comunità del software libero (pagine web, sistemi di controllo delle versioni, ecc.).

Queste risorse possono essere motivo di differenze significative nella diffusione e nell'acquisizione di popolarità tanto dell'applicazione quanto del progetto, nella comunità degli utenti del software libero.

Marketing

Infine, anche se si tratta di un progetto basato sul software libero, devono essere applicate misure di marketing per la sua diffusione e l'acquisizione di popolarità tanto dell'applicazione quanto del progetto in generale.

Per questo è importante ricordare che l'intero funzionamento del progetto si trova esposto al pubblico, e che ognuna della affermazioni che si fanno può essere facilmente dimostrata o smentita. Stabilire delle misure di controllo dell'immagine e del funzionamento del progetto permette di guadagnare in credibilità, trasparenza e verificabilità.

In mezzo a queste misure, va sottolineata l'importanza che assume il mantenere una politica aperta, onesta ed oggettiva rispetto ai progetti dei concorrenti. Da un lato perché riafferma un valore stabile per la comunità degli utenti, e dall'altro perché favorisce lo sviluppo di strategie di cooperazione tra progetti connessi.

2. La comunità di utenti

Come avevamo messo in rilievo fin dalla prima parte di questo modulo, il ruolo che gioca la comunità degli utenti del software libero è molto importante nel paradigma di sviluppo del software libero.

Tanto gli utenti come gli sviluppatori che sono parte della comunità collaborano al mantenimento, al supporto ed all'evoluzione dell'applicazione nel corso del tempo, assicurando così la coesione e la stabilità del progetto.

Di conseguenza, la loro partecipazione è fondamentale per dotare di garanzie valide gli obiettivi del progetto, e deve essere considerata come tale da qualunque organizzazione a scopo di lucro che desideri cogliere l'opportunità di un business basato sulla produzione del software libero.

Per questo motivo la relazione tra comunità ed impresa deve fondarsi sulla credibilità e sulla trasparenza di tutte le azioni e di tutte le decisioni che vengono prese, cosicché entrambe le parti possano trarre profitto da questa relazione. Non è invano che si richiede che il posizionamento dell'impresa quanto ai prodotti basati sul software libero sia ben definito e ben strutturato, ciò serve infatti a favorire la formazione di una rete di collaboratori intorno all'impresa.

Bisogna tener conto del fatto che la comunità di utenti è un'organizzazione dinamica e che evolve nel tempo, per cui sarà necessario stabilire delle metodologie di gestione per mantenere sempre la relazione ad un ottimo livello. Questa gestione comprende la fissazione di procedimenti per identificare lo stato attuale della comunità, valutare la qualità dei contributi al progetto da parte dei membri e definire gli aspetti legali legati a questi contributi.

Nei prossimi paragrafi prenderemo in esame ognuno di questi aspetti singolarmente.

2.1. Gestione della comunità

Per raggiungere gli obiettivi del progetto, l'impresa che intraprende un progetto di sviluppo di software libero deve strutturare minuziosamente la sua relazione con la comunità di utenti.

Nella prima parte di questo modulo abbiamo già menzionato i principali aspetti sui quali edificare un progetto di software libero. Nel caso in cui un'impresa faccia da promotrice del progetto, sarà necessario stabilire ed organizzare una strategia adeguata agli obiettivi dell'impresa, ma sempre tenendo in debita considerazione che deve offrire contropartite per la collaborazione che si aspetta di ottenere dalla comunità degli utenti.

In questo senso, e così come in qualsiasi altro progetto di software libero, aspetti come la credibilità o la trasparenza - tra gli altri ? giocheranno un ruolo molto importante nella creazione di una comunità di utenti intorno al progetto.

Ben Collins-Sussman e Brian Fitzpatrick hanno identificato e classificato le diverse *Open Source* che può adottare un'impresa basata sullo sviluppo di un software libero nella conferenza intitolata 'What's in it for me'? di OSCON 2007.

Web consigliato

B. Collins-Sussman; B. Fitzpatrick (2007). "What's in it for me?"
(http://www.youtube.com/watch?v=ZtYJoatnHb8).

Questa classificazione tiene conto delle due componenti principali della relazione tra impresa e comunità:

- Da un lato, l'orientamento, la struttura ed il funzionamento generale del progetto, così come le responsabilità dell'impresa in esso.

- Dall'altro, i benefici e gli inconvenienti per l'impresa e per la comunità di utenti che scaturiscono dalla scelta di una strategia concreta per portare il progetto a termine.

In effetti, il lavoro di Collins-Sussman e di Fitzpatrick si avvicina molto ad essere un manuale di buone maniere atte ad instaurare una relazione sana e soddisfacente tra impresa e comunità di utenti.

Di seguito presenteremo brevemente i tratti principali di questa classificazione di strategie *Open Source*.

Fake Open Source

Questa strategia si fonda sulla concessione del codice fonte dell'applicazione con una licenza non approvata dall'OSI.

Web consigliato

Open Source Initiative (http://www.opensource.org/).

A dire il vero, non si tratta di una strategia *Open Source* vera e propria, dato che non solo ne vengono persi i benefici, ma alcuni membri della comunità potrebbero perfino arrivare a boicottare il progetto.

Nonostante questo, però, il progetto può avere una copertura mediatica e così catturare l'attenzione con un costo o uno sforzo relativamente bassi.

Throw code over the wall

Questa strategia è simile alla precedente, ma questa volta l'impresa concede il codice con una licenza approvata dall'OSI, anche se continua a non preoccuparsi sul futuro del progetto.

Quanto sopra significa che se concede il codice e poi se ne dimentica, l'impresa dà di sé un'immagine di scarsa credibilità, dal momento che lancia un'applicazione senza che esista una comunità di utenti che permetta di mantenere il progetto attivo. Un possibile inconveniente è infatti che si creino comunità alternative che sviluppano il software indipendentemente dagli obiettivi dell'impresa.

Develop internally, post externally

Questa strategia si basa sullo sviluppo dell'applicazione all'interno dell'impresa, per poi pubblicarne gli aggiornamenti in un'area pubblica.

In questo modo l'impresa migliora tanto le relazioni pubbliche con la comunità degli utenti quanto la credibilità all'interno del mondo del software libero. Da parte sua, la comunità potrebbe collaborare attivamente al progetto. Nonostante questo, il fatto che lo sviluppo avvenga totalmente all'interno può dare impulso alla nascita di comunità parallele ed indipendenti dalle strategie dell'impresa (che genera un certo grado di sfiducia).

Open monarchy

Questa strategia si sviluppa attraverso la manifestazione al pubblico tanto delle discussioni quanto del contenuto delle applicazioni, anche se gli utenti che godono di diritti su di essa sono interni all'impresa.

Con questa strategia migliorano sensibilmente la credibilità e la trasparenza delle imprese e dei contributi della comunità (cosa che ha effetti positivi sul codice), anche se è l'impresa ad avere comunque l'ultima parola su tutte le decisioni che vengono prese. Quest'ultimo aspetto costituisce un rischio per la buona conservazione della comunità nel lungo periodo, ed esiste perfino il rischio di una separazione nel progetto, che potrebbe snodarsi su due distinti cammini (*fork*)

Consensus-based development

Questa strategia si serve di praticamente tutte le possibili forme di relazione tra impresa e comunità, dato che praticamente tutto viene realizzato in forma pubblica.

Il progetto, in questo caso, si articola tanto attraverso la presa decentralizzata di decisioni, quanto attraverso sistemi di funzionamento di tipo meritocratico tra i collaboratori.

Queste caratteristiche si riversano a formare un modello sostenibile nel lungo periodo con volontari di alto profilo, dato che l'impresa guadagna in credibilità, trasparenza e fiducia di fronte alla comunità ed alle altre imprese basate sul software libero.

Di contro, nel breve periodo i benefici sono scarsi ed il volume di lavoro è significativo. Pertanto il ruolo dei leader del progetto è rilevante per il funzionamento strategico dell'intera organizzazione.

2.2. Caratteristiche della comunità

La comunità di utenti del software libero è un'organizzazione dinamica ed in evoluzione, nel senso che esistono diversi fattori che influiscono sulla sua situazione e ne modellano in maggior o minor misura le tendenze.

Se si fa una riflessione su un progetto di software libero, si scopre che è preferibile creare quanto prima una solida comunità di utenti attorno all'applicazione, dato che una buona parte del suo successo e del conseguimento dei suoi obiettivi emergono proprio grazie alla comunità.

Una volta creatasi la comunità, è importante fare un piano di strategie che permettano non solo di mantenerla solida, ma anche di ampliarla e di farla evolvere per lo meno allo stesso ritmo del prodotto stesso. Uno step precedente a qualunque tipo di azione intesa in questo senso è di conoscere con precisione ed in ogni momento lo stato della comunità e la sua tendenza evolutiva più recente ed aggiornata, in relazione al progetto.

Cogliere appieno ed in ogni momento lo stato della comunità di utenti può essere relativamente complicato, sul piano pratico, soprattutto per le sue caratteristiche intrinseche di distribuzione e decentramento.

Nonostante ciò, si può comunque individuare una serie di indicatori che agevolino le decisioni, capaci come sono di fornire un quadro della situazione abbastanza realistico.

Nell'articolo di Crowston e Howison 'Assessing the Health of a FLOSS Community' viene proposta una guida semplice, ma efficace, per identificare e soppesare compiutamente lo stato di una comunità di utenti di software libero. Questa guida tiene in considerazione i principali indicatori cui prestare attenzione quando si voglia fare una valutazione dello stato di salute della comunità e, pertanto, del progetto basato sul software libero.

Web consigliato

K. Crowston; J. Howison (2006). "Assessing the health of a FLOSS Community" (http://floss.syr.edu/publications/Crowston2006Assessing_the_health_of_open_source_communities.pdf)

Nei prossimi paragrafi forniremo alcune delle conclusioni tratte nell'articolo.

Ciclo di vita e motivazioni

Vari autori sono d'accordo nel considerare che i progetti si iniziano in seno a un ristretto gruppo di promotori, per poi trovare una loro struttura e svilupparsi in forma pubblica.

Una volta messo in moto il progetto, va iniziata una seconda fase relativa al perfezionamento progressivo del concetto di base. Questo implica la condivisione di idee, suggerimenti e conoscenze, la cui raccolta consente di migliorare il concetto originario. Questo processo non può essere realizzato senza la cooperazione della comunità del software libero.

D'altra parte, la motivazione che spinge i membri della comunità a partecipare attivamente al progetto si fonda principalmente sullo sviluppo intellettuale, sulla condivisione delle conoscenze, sull'interesse per l'applicazione, sulla filosofia che sta dietro al progetto o al software libero in generale e sulla reputazione pubblica, così come sugli obblighi interni alla comunità.

Struttura e dimensione della comunità

La comunità degli utenti di un'applicazione basata sul software libero può trovare la sua struttura più appropriata attraverso diverse modalità, che si differenziano in base alle azioni ed alle decisioni dei promotori del progetto, così come in base alle caratteristiche dell'applicazione e della sua produzione.

In generale, si può dire che la comunità di utenti di un'applicazione è in salute se presenta una struttura gerarchica funzionale adeguata agli obiettivi, attorno ad un nucleo attivo di sviluppatori.

Struttura gerarchica

Questo concetto può essere paragonato alla struttura di una cipolla (*onion-shaped* in inglese): al centro si mettono i membri più attivi e collegati al progetto, e nella parte più esterna quelli che partecipano di meno.

A grandi linee, possiamo classificare le seguenti tipologie di membri all'interno di un progetto:

- Sviluppatori del nucleo dell'applicazione, con diritti di scrittura sul contenitore e con una storia pregressa di contributivi significativi al progetto.

- Leader del progetto, che forniscano le motivazioni e portino il progetto e la sua comunità ad un livello di maturità e di stabilità.

- Sviluppatori in generale, che contribuiscono al codice ma non hanno diritti di scrittura sul contenitore. Sono spesso incaricati di compiti di revisione.

- Utenti attivi che provano l'applicazione, segnalano errori, forniscono una documentazione ed agiscono da tessuto connettivo tra il progetto e gli utenti passivi (oltre ad altri compiti).

Nota

Questa prima classificazione di tipologie non è una struttura chiusa, dato che ogni progetto la adatta alle sue particolari caratteristiche.

Processi di sviluppo

Il processo di sviluppo del software libero risulta essere spesso poco strutturato all'interno dei progetti, e questo è dovuto basicamente all'assenza di un piano di orientamento, all'imprecisa assegnazione dei compiti specifici ed alla mancanza di un ordine di priorità nelle funzioni dell'applicazione.

L'organizzazione del progetto è una caratteristica rilevante per il funzionamento ed il coordinamento della produzione, anche se un certo grado di duplicazione degli sforzi può essere visto come un sintomo positivo circa la relazione ed il coinvolgimento della comunità nel progetto.

Alla stessa maniera, il ciclo di valutazione e di posteriore accettazione dei contributi al progetto da parte dei membri della comunità permette di disegnare un quadro preciso sulla salute del suo funzionamento. Ad esempio, il rifiuto di un contributo può essere rappresentativo di una visione coesa e qualitativa del progetto a lungo termine.

2.3. Gestione della qualità

In alcune occasioni, la qualità del software libero è stato oggetto di dibattito tra promotori e detrattori. Nel corso di esso viene posto l'accento su aspetti come l'apertura del modello di sviluppo o la formazione dei collaboratori che contribuiscono al progetto, ad esempio.

Come in qualsiasi progetto di software, la produzione di software libero deve porsi delle regole di controllo di qualità nel corso del suo ciclo di vita. La qualità, cioè, deve poter essere misurata e confrontata con i valori attesi in ogni tappa della produzione del software e del suo utilizzo, così come da ogni punto di osservazione (promotore, utente o comunità).

In questo senso, anche se l'apertura ed il decentramento del modello di sviluppo del software libero favoriscono i meccanismi di controllo e di gestione della qualità, non costituiscono altresì una soluzione di per se stesse, ed è importante sottolinearlo perché non si deve smettere di cercare nuove soluzioni di verifica della qualità.

Per sviluppare gli aspetti relativi alla qualità della produzione del software libero, sfrutteremo l'articolo 'Managing Quality in Open Source Software' di Dhruv Mohindra, che realizza uno studio attento sul controllo di qualità negli aspetti riconducibili al software libero. Nei prossimi paragrafi vedremo i principali concetti offerti dall'autore.

Web consigliato

D. Mohindra (2008). "Managing Quality in Open Source Software"
(http://www1.webng.com/dhruv/material/managing_quality_in_oo.pdf).

La qualità nel software libero

In generale, la qualità di una soluzione di software può essere valutata tanto in base alla sua architettura o al suo disegno interno, quanto in base alle funzioni che fornisce all'utente.

Le caratteristiche di apertura e decentramento proprie del modello di sviluppo del software libero costituiscono una infrastruttura sulla quale si possono poggiare politiche di gestione della qualità basate sull'individuazione e sulla risoluzione di alcune problematiche (tra vari aspetti). Pur essendo vero questo, a volte la mancanza di chiarezza e/o di una corretta strutturazione dei processi di produzione può generare risultati che non sono quelli attesi.

Misurazione della qualità

Esistono diversi modi per valutare la qualità funzionale di un'applicazione. La funzionalità di metodologie quantitative dipende in gran parte dalla tipologia stessa del software, e pertanto vanno scelte in funzione delle caratteristiche e degli obiettivi dell'applicazione.

Quanto al discorso relativo alla qualità 'non quantificabile', bisogna porre in risalto il ruolo che riveste la comunità del software libero. Da un lato i test che realizza il team sulla qualità, dall'altro l'attività propria degli utenti dell'applicazione, che segnalano l'evidenza di errori di funzionamento o di elementi migliorabili del prodotto.

In quest'ottica, lo stesso funzionamento distribuito e decentralizzato della comunità di utenti è importante per dare più garanzie di qualità al processo di produzione.

Controllo e revisione

Un fattore importante per la qualità del prodotto finale è il controllo, abbinato alla revisione, di tutto il processo di sviluppo. In generale, i progetti di produzione di software libero utilizzano sistemi di controllo delle versioni per supportare con efficacia ed efficienza l'evoluzione delle diverse componenti del progetto.

Ci sono più maniere di organizzare il controllo e la revisione dell'evoluzione del software, così come dei rami di sviluppo e del contenitore (tra gli altri). Ad ogni modo conviene adattare la metodologia della produzione ed i sistemi di controllo e revisione dell'evoluzione alle peculiarità del progetto e del prodotto che si sta realizzando.

Leggende sul software libero

Nonostante la sua costante diffusione, il software libero ancora oggi porta con sé alcuni miti -tanto positivi quanto negativi- che possono influire sulla considerazione che gli viene data.

Questi miti non hanno nessuna base salda attraverso la quale si possa ideare ed implementare una gestione coerente e sostenibile della qualità, e pertanto si deve analizzare ognuno di essi singolarmente.

A continuazione elenchiamo alcune leggende abituali legate alla qualità del software libero.

- Il fatto che il codice fonte sia pubblico non è garanzia di sicurezza e qualità, dato che in ogni caso dipende dall'interesse e dalla revisione della comunità.

- Il congelamento delle funzioni non aumenta la stabilità dell'applicazione di per se stessa, dal momento che la cosa importante è che il codice sia scritto bene fin dall'inizio.

- Il miglior modo di comprendere un progetto non è quello di correggere i suoi eventuali errori, dato che la documentazione è di gran lunga migliore per questo obiettivo.

- In generale, gli utenti non dispongono dell'ultima versione del contenitore con le correzioni degli errori aggiornate ogni giorno.

A grandi linee, i processi di verifica e di revisione, così come le discussioni pubbliche e la cultura *hacker* proprie della comunità di utenti, devono essere accompagnati da una pianificazione e da una gestione attiva della qualità della produzione.

Questa gestione deve essere diretta a coprire eventuali lacune in uno o più aspetti del prodotto, ad esempio la pianificazione della produzione, lo sviluppo delle funzioni o la documentazione dell'applicazione.

Considerazioni aggiuntive rispetto alla qualità

In generale, tanto la pubblicazione del codice fonte quanto l'inclusione di sistemi di gestione dell'errore, ed il fatto di ripartire le responsabilità relative al prodotto tra tutti coloro che vi sono implicati, sono aspetti chiave per la gestione della qualità.

In questo senso, è molto importante per la qualità generale del progetto considerare anche la trasparenza intrinseca in tutte le azioni, fidarsi del team di sviluppo, verificare e testare tutte le parti del codice fonte, e promuovere tanto la filosofia quanto l'importanza di farlo bene fin dall'inizio.

2.4. Legalità e contributi

In un progetto basato sul software libero con la partecipazione della comunità degli utenti, assume particolare rilevanza la gestione legale dei contributi di ogni membro coinvolto.

Questa gestione è importante per i promotori del progetto tanto quanto per i membri della comunità, dato che maneggia le caratteristiche di paternità e di titolarità dei diritti del codice. La sua importanza è data anche dalle implicazioni che può creare la combinazione di codici di differenti autori in uno stesso prodotto.

Per sviluppare questi concetti prenderemo spunto dalla lettura della sezione 2.4 'Autori e titolari del diritto', presa dai materiali didattici del corso *Aspetti legali e dello sfruttamento del software libero*.

Web consigliato

M. Bain ed altri (2007). *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya (http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/aspectes-legals-i-dexplotacio/materials/).

L'autore

L'autore di un'opera è la persona fisica o giuridica che realizza l'opera, e questo implica inevitabilmente che gli appartiene la paternità della sua creazione originale.

Le opere realizzate da più persone si collocano in diverse fattispecie:

- Un'opera in collaborazione è il risultato unitario di una composizione di diverse parti che possono essere utilizzate e sfruttate indipendentemente.

- Un'opera collettiva è l'unione di vari contributi che non si possono utilizzare e sfruttare in modo indipendente.

- In un'opera realizzata su commissione (o con contropartite economiche), la paternità ricade sulla persona fisica o giuridica che realizza il prodotto.

Nel mondo del software libero, la paternità dipende in larga misura dalle considerazioni di cui sopra, tenendo sempre a mente che a volte il trasferimento della paternità può risultare utile e pratico.

D'altro canto, le condizioni attraverso le quali si realizzano le opere derivate (preesistenza del contenuto) possono cambiare anche sostanzialmente a causa dell'autore o dell'opera stessa. In ogni caso, le licenze libere devono specificare le condizioni di derivazione e di redistribuzione delle opere.

Il titolare originario e il titolare derivato

Il titolare originario dell'opera è sempre il suo autore. Nonostante questo, alcuni diritti sull'opera possono essere ceduti ad altre persone, fisiche o giuridiche.

In questo caso, la persona che riceve la cessione di parte dei diritti relativi ad un'opera diventa il titolare derivato della stessa. È necessario sottolineare che solo il titolare di un diritto concreto può concedere licenze sul suo diritto.

Identificazione del titolare

Per poter esercitare i diritti di cui sopra deve essere possibile identificare l'autore di ogni opera. Questo può essere complicato nel mondo del software libero, dal momento che coloro che contribuiscono al progetto possono essere molteplici.

Per attenuare questo tipo di problemi, i progetti basati sul software libero conservano sempre le liste di coloro che vi hanno preso parte. A volte questi progetti richiedono la cessione dell'intero pacchetto, o solo di alcuni dei diritti, prima di accettarne il contributo.

3. Caso di studio

Nelle sezioni precedenti sono stati esaminati tanto il progetto di software libero quanto la gestione della comunità degli utenti. Entrambe le sezioni riportano i principali aspetti legati alla produzione del software libero dal punto di vista della gestione del progetto.

Per concludere questo modulo, dedicheremo questo ultimo capitolo a rendere più concreta una buona fetta delle idee e delle proposte presentate, attraverso lo studio di un caso concreto di un'impresa basata sul software libero.

Le prossime parti hanno l'obiettivo di fornire una guida per rendere più chiaro come un'impresa basata sulla produzione di software libero implementa la sua propria metodologia, crea a gestisce la relazione con la comunità degli utenti, e come affronta molte delle decisioni che deve prendere con il passare del tempo.

Ci concentreremo sul caso della Openbravo, S.L.

3.1. L'impresa

Openbravo, S.L. è un'impresa che si dedica a sviluppare soluzioni professionali basate sul software libero per le imprese.

Nota

Tutta l'informazione presentata in questa sezione è stata tratta dal suo sito web (http://www.openbravo.com/).

Modello di business

Il modello di business che utilizza l'impresa si basa sulla prestazione di servizi relativi ai prodotti che sviluppa. Come è già stato detto, la sua strategia di business è basata sull'associazione e sulla collaborazione tra imprese per sfruttare la stessa possibilità di business.

Strategia dell'impresa

Il modello di business viene portato avanti mediante dei *partner* che offrono servizi ai clienti finali (ad esempio la personalizzazione ed il supporto). In un certo qual modo, questo particolare tipo di gerarchia tra produttore, distributore (o *partner*) e cliente stabilisce un'atmosfera di cooperazione con obiettivi comuni.

Per portare a termine la sua strategia, l'impresa pubblica un manifesto come fosse una dichiarazione di intenti, che include tanto aspetti legati al software libero (come la trasparenza, l'apertura e la collaborazione), quanto gli obblighi dell'impresa contratti rispetto a terzi (come l'accesso libero e la gestione dei contributi).

Gestione e direzione

La gestione dell'impresa combina compiti interni ed esterni all'organizzazione, tanto nella squadra di direzione quanto nel Consiglio di Amministrazione, prodotto dell'investimento esterno che ha ricevuto l'impresa, così come della speciale metodologia basata sul software libero.

3.2. I prodotti

Openbravo realizza due soluzioni basate sul software libero, che possono funzionare l'una indipendentemente dall'altra o in combinazione. Entrambi i prodotti sono distribuiti con licenza libera e con download diretto da internet.

I prodotti che offre Openbravo sono i seguenti:

- **Openbravo ERP**
  Openbravo ERP (*Enterprise Resource Planning*) è un sistema di gestione imprenditoriale in un ambiente web, che integra in una struttura organizzata a moduli diverse funzioni di gestione, quali gli approvvigionamenti, il magazzino, la produzione o la contabilità.
  Il prodotto è dato con licenza MPL 1.1 e può funzionare in diversi ambienti ed in diversi sistemi di base dei dati, così come può integrarsi con Openbravo POS.

  Tra le molte informazioni sul prodotto che vengono fornite spicca la mappa del piano di sviluppo del progetto.

Web consigliato

Mozilla Public License 1.1 (http://www.mozilla.org/MPL/MPL-1.1.html).

Caratteristiche principali di Openbravo ERP

http://sourceforge.net/projects/openbravo/.

- **Openbravo POS**
  Openbravo POS (*Point Of Sales*) è un sistema di terminali di punti di vendita che può essere integrato con Openbravo ERP.
  Il produtto è dato con licenza GNU/GPL e può funzionare in differenti contesti e con diversi sistemi di base di dati. È realizzato in particolare per terminali tattili.

  Tra le informazioni che vengono fornite, anche qui spicca la mappa del piano di sviluppo del progetto.

Web consigliato

GNU General Public License (http://www.gnu.org/licenses/gpl.html).

Caratteristiche prinicipali di Openbravo POS

http://sourceforge.net/projects/openbravopos/.

3.3. La comunità degli utenti

La comunità degli utenti del software libero riveste un ruolo rilevante nella strategia imprenditoriale di Openbravo. Qui sotto ne menzioneremo gli aspetti principali.

Strategia *Open Source*

Per identificare la strategia *Open Source* di Openbravo dobbiamo tenere a mente le peculiarità della metodologia di sviluppo dei prodotti e la struttura di business che utilizzano.

Il nucleo di entrambi i prodotti viene sviluppato per lo più internamente all'impresa, mantenendo però contenitori pubblici ed una comunità di utenti attiva al suo fianco. D'altra parte, nello sviluppo dei complementi, delle personalizzazioni e delle estensioni rispetto al prodotto originale entrano in gioco tanto la comunità degli utenti quanto i *partner*.

Questo ultimo caso va analizzato separatamente, dal momento che corrisponde all'utilizzo di un'opportunità da parte di un'organizzazione differente.

Detto questo, Openbravo realizza una strategia *Open Source* che combina diversi orientamenti:

- Per lo sviluppo e la revisione dei prodotti, la strategia si avvicina all'*Open Monarchy*, principalmente a causa dello sviluppo interno del nucleo dei prodotti, dei contenitori pubblici del codice fonte, dell'approvazione finale dei cambi relativi al nucleo a carico dell'impresa e della progettazione dello sviluppo dei progetti (ad esempio, le direzioni intraprese).

- Per lo sviluppo dei complementi (ad esempio, la documentazione), la strategia è simile al *Consensus-based development*, visto che lo sviluppo avviene all'interno di una comunità di utenti.

- Infine, la strategia per lo sviluppo delle estensioni e delle personalizzazioni dipende dallo sviluppatore che la pianifica. Se sono progetti realizzati in seno alla comunità (attraverso le risorse offerte da Openbravo), si avvicinano più propriamente al modello del *Consensus-based development*, mentre se sono sviluppate dai *partner* dipenderanno tanto dalla loro particolare strategia quanto dalle caratteristiche dello sviluppo.

Strategia del *partner*

Nel caso in cui il *partner* sviluppi delle estensioni del prodotto originale, la strategia *Open Source* dipenderà tanto dalla sua filosofia imprenditoriale quanto dalle caratteristiche del prodotto (ad esempio, la licenza MPL è più flessibile con i moduli proprietari rispetto alla GPL).

Struttura della comunità

La comunità degli utenti di Openbravo ERP è definita e strutturata secondo un sistema meritocratico: esistono vari livelli di collaborazione ed ognuno di essi si definisce a partire dalle conoscenze necessarie per il livello, dalla quantità di contributi, dalle responsabilità e dai privilegi.

Nel caso di Openbravo ERP esistono tre diversi profili di collaborazione (sviluppatori, esperti funzionali e tester), mentre nel caso di Openbravo POS esiste solo il profilo dello sviluppatore. I membri della comunità di utenti si organizzano e si distribuiscono a seconda dei progetti attivi nella comunità.

Risorse a disposizione della comunità

Openbravo dispone di diverse risorse (alcune della quali in più di una lingua) sia per la comunità sia per i *partner* e per gli utenti in generale, tra le quali troviamo:

Web consigliato

Dal portale web dell'impresa si può accedere a tutte le risorse menzionate (http://www.openbravo.com/).

- Web corporativa
- Area dei *partner*
- Wiki del progetto
- Partale della comunità degli utenti di Openbravo
- *Blog* dei lavoratori
- Modellazione dei prodotti (Openbravo ERP e Openbravo POS)
- *Bug tracker*
- Università
- Liste di distribuzione della posta
- Contenitore del codice di Openbravo
- Servizio di notizie di Openbravo

In generale, la comunità degli utenti ha a disposizione una guida specifica disponibile nel wiki che descrive come collaborare al progetto. Dispone anche di una lista esauriente dei canali di comunicazione ai quali può accedere. In più, le strategie di orientamento di ogni prodotto sono l'ultima componente presente nella guida alle risorse per la comunità degli utenti.

3.4. Posizionamento ed evoluzione

L'impresa è nata nel 2001 con il nome Tecnicia. Nel 2006 ha ottenuto dei finanziamenti per più di sei milioni di dollari e ha cambiato nome, passando a Openbravo. Nello stesso anno ha liberato il codice fonte dei prodotti che sviluppa con licenze libere.

Nel maggio del 2008, i flussi di finanziamenti sono saliti a più di dodici milioni di dollari, e tra i suoi investitori appaiono Sodena, GIMV, Adara e Amadeus Capital Partners.

Negli anni, Openbravo ha ricevuto diversi premi legati al mondo dell'impresa e del software libero, e allo stesso modo ha ricevuto sovvenzioni nell'ambito del programma di impulso alla innovazione tecnologica (PROFIT) da parte del Ministero spagnolo per l'Industria, il Turismo ed il Commercio.

Sia l'impresa sia la comunità degli utenti hanno una tendenza evolutiva positiva, se pensiamo che il progetto si situa attualmente tra i venticinque più attivi di SourceForge, con più di un milione di download accumulati agli inizi del 2009.

Web consigliato

Sui progetti più attivi di SourceForge:
http://sourceforge.net/top/mostactive.php?type=week.

Riepilogo

In questo modulo abbiamo esposto le principali caratteristiche legate alla creazione, alla gestione ed al mantenimento del progetto di sviluppo del software libero, facendo particolare attenzione alla partecipazione della comunità degli utenti.

In una certa maniera, i fondamenti della produzione di software libero non sono troppo diversi dalle metodologie di sviluppo dei software più tradizionali. Nonostante questo, le peculiarità dell'apertura del codice e dell'esistenza di una comunità di utenti ne modellano il funzionamento e lo rendono particolare sotto molti punti di vista.

Per quello che riguarda il progetto in sé, bisogna sottolineare l'importanza che rivestono l'identificare, definire e strutturare gli aspetti funzionali del progetto (ad esempio, l'infrastruttura, la gestione delle versioni o le misure di coordinamento) e quelli legati al software libero (ad esempio, la credibilità, la trasparenza o le tipologie di partecipazione).

A questi aspetti vanno sommati i fattori legati alla comunità del software libero, come la strategia di gestione della comunità da parte dell'impresa (strategia *Open Source*), la metodologia ed il ciclo di vita del prodotto, la gestione della qualità e gli aspetti legali relativi ai contributi apportati dagli utenti.

Da ultimo, abbiamo presentato un caso di studio rappresentativo di una buona parte degli aspetti esaminati nel corso del modulo.

Bibliografia

Bain, M. y otros (2007). *Aspectes legals i d'explotació del programari lliure*. Universitat Oberta de Catalunya <http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/aspectes-legals-i-dexplotacio/Course_listing> [Consultazione: febbraio 2009].

Collins-Sussman, B.; Fitzpatrick B. (2007). *What's in it for me? How your company can benefit from open sourcing code*. OSCON: 27 luglio 2007 <http://www.youtube.com/watch?v=ZtYJoatnHb8> e diapositive <http://www.red-bean.com/fitz/presentations/2007-07-27-OSCON-whats-in-it-for-me.pdf> [Consultazione: febbraio 2009].

Crowston, K.; Howison, J. (mayo, 2006). *Assessing the Health of a FLOSS Community*. IT Systems perspectivas (pág. 113-115). <http://floss.syr.edu/publications/Crowston2006Assessing_the_health_of_open_source_communities.pdf> [Consultazione: febbraio 2009].

Fogel, K. (2005). *Producing Open Source Software: How to Run a Successful Free Software Project.* <http://producingoss.com> [Consultazione: febbraio 2009].

Mako, B. (2001). *Free Software Project Management HOW TO*. <http://mako.cc/projects/howto> [Consultazione: febbraio 2009].

Mohindra, D. (2008). *Managing Quality in Open Source Software*. <http://www1.webng.com/dhruv/material/managing_quality_in_oo.pdf> [Consultazione: febbraio 2009].

Openbravo <http://www.openbravo.com/> [Consultazione: febbraio 2009].

Raymod, E. (1997). *The cathedral and the bazaar* <http://www.catb.org/~esr/writings/cathedral-bazaar/> [Consultazione: febbraio 2009].

Tawileh, A. ed altri (agosto 2006). *Managing Quality in the Free and Open Source Software Community* (pags. 4-6). Proceedings of the Twelfth Americas Conference on Information Systems. México: Acapulco. <http://www.tawileh.net/anas//files/downloads/papers/FOSS-QA.pdf?download> [Consultazione: febbraio 2009].

Strategie del software libero come business

Amadeu Albós Raya

PID_00145046

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

Nel business del software libero, così come in qualunque altro basato sulla tecnologia in generale, subentra una molteplicità di fattori che influiscono in minor o maggior misura sul suo successo. Una buona parte di questi fattori viene trattata nei successivi moduli, come ad esempio le caratteristiche del mercato del software, i modelli di business o le particolari specifiche esistenti nella produzione del software libero.

In questo senso, deve essere minuziosamente messo a punto l'insieme di azioni che permettono di stabilire un'opportunità di business efficace e sostenibile, per raggiungere gli obiettivi che si sono fissati. Nello specifico, è di fondamentale importanza trasferire le caratteristiche del software libero inteso come business al mercato-obiettivo reale, così da stabilire una strategia d'impresa concreta e adeguata che permetta di godere dei vantaggi che offre il software libero e di mantenere sotto controllo gli inconvenienti che dovessero presentarsi.

Questa strategia deve riflettere la realtà del contesto e dell'ambiente d'impresa, attraverso l'identificazione e l'analisi delle prospettive di ognuno degli attori del mercato, così da massimizzare il più possibile le probabilità di successo.

Nel corso di questo modulo presenteremo le principali caratteristiche che giocano un ruolo nel business delle imprese che si occupano di software libero, quelle che rendono il business peculiare, assegnando ad ognuno degli elementi la qualifica di vantaggio o di inconveniente derivante per il business in oggetto.

Obiettivi

Al termine di questo modulo, si dovrà essere in grado di:

1. comprendere l'importanza della strategia nel business basato sul software libero.

2. identificare e dare il giusto peso ai vantaggi derivanti dal business del software libero.

3. identificare e dare il giusto peso agli inconvenienti legati al business del software libero.

4. isolare ed approfondire le strategie dei modelli di business del software libero.

1. La competitività del software libero

Il software libero costituisce oggi una alternativa efficace e sostenibile al software di proprietà. Caratteristiche quali l'adattabilità del suo sviluppo e della sua installazione, l'operatività basata su misure standard e la costante evoluzione delle applicazioni sono ragioni sufficienti a sostegno dell'effettiva competitività del software libero.

A dispetto di quanto riporato sopra, questa competitività può non essere sufficiente per il business del software libero se le caratteristiche presentate non vengono adeguatamente indirizzate. Per fare affidamento su un progetto stabile e credibile nel tempo, infatti, è imperativo definire una strategia di business che coordini ed armonizzi i vantaggi che offre, e nello stesso tempo che sappia gestire e ridurre i suoi inconvenienti.

In questo primo paragrafo facciamo un breve ripasso delle principali caratteristiche che fanno del software libero un'alternativa competitiva rispetto al software di proprietà.

Web consigliato

M. Boyer; J. Robert (2006). *The economics of Free and Open Source Software: Contributions to a Government Policy on Open Source Software* (cap. 3, "Advantages and disadvantages of FOSS").

<http://www.cirano.qc.ca/pdf/publication/2006RP-03.pdf>

Costo

In genere, le applicazioni basate sul software libero si possono ottenere liberamente e gratis su Internet. Questa filosofia di distribuzione si colloca agli antipodi rispetto al modello di proprietà, che impone il pagamento di una somma per l'utilizzo limitato del formato binario dell'applicazione.

Di conseguenza, il costo rappresenta un vantaggio competitivo importante per la loro adozione rispetto alle alternative di proprietà, dato che possono ridurre significativamente l'investimento necessario alla loro installazione (che si tratti di una novità o di un aggiornamento radicale del sistema).

D'altro canto, la riduzione del costo può essere significativa anche nei casi di miglioria o di specializzazione di un'applicazione concreta, e questo in quanto, mentre il software libero garantisce la possibilità di adeguare l'applicazione

alle più disparate necessità degli individui attraverso il libero accesso al codice fonte, per il software di proprietà può essere necessario uno sviluppo del tutto nuovo.

Sviluppo, flessibilità ed adattabilità

Anche se a volte lo sviluppo di una tecnologia basata sul software libero può non essere molto diverso da quello dell'equivalente tecnologia di proprietà, la metodologia che sfrutta la collaborazione e l'aggiornamento congiunto di impresa e comunità di utenti offre il vantaggio della cooperazione di scala.

Queste peculiarità offrono un ventaglio di possibilità che vanno dalle economie di scala e la creazione di mercati segmentati alla flessibilità ed all'adattabilità che permettono di migliorare tanto l'interazione e l'integrazione tra applicazioni, quanto la loro estensione ed il loro sviluppo. In definitiva, sono tutte caratteristiche che contribuiscono alla creazione di concrete opportunità di business.

Rischio tecnologico

Detto a grandi linee, i rischi legati all'adozione della tecnologia colpiscono allo stesso modo il software libero e quello di proprietà, almeno da un punto di vista strettamente tecnologico.

Fatta questa premessa, e nel caso di applicazioni o soluzioni concrete, il rischio ha un nesso maggiore con le qualità e le caratteristiche specifiche di ognuna di esse, che non con la tecnologia o la metodologia utilizzata per il loro sviluppo.

Sicurezza, affidabilità e ciclo di vita

Col passare del tempo l'evoluzione delle metodologie di sviluppo del software ha consentito di tenere un maggior ed un miglior controllo sulla qualità del software prodotto, soprattutto su aspetti come la correzione degli errori.

In questo caso, l'apertura del processo di sviluppo del software libero e la collaborazione della comunità degli utenti conferiscono al processo stesso una differenza sostanziale rispetto al modello di proprietà. Sembra difficile, infatti, che una impresa che produce software di proprietà possa arrivare ad impiegare lo stesso numero di risorse umane e temporali impiegate nei progetti di software libero.

Questa peculiarità del software libero favorisce la competitività e l'affidabilità delle soluzioni, per le imprese così come per i loro clienti.

Supporto e documentazione

In alcune occasioni, le applicazioni che operano mediante il software libero sono carenti sul lato dell'incartamento che le accompagna. Spesso infatti non sono accompagnate dallo stesso pacchetto che solitamente correda le applicazioni equivalenti ma di proprietà. Dal punto di vista commerciale, questa situazione può rappresentare una fonte di opportunità di business a diversi livelli, con il vantaggio addizionale che possono rappresentare la specializzazione e la prossimità al cliente.

Gestione del cambiamento

Ristrutturazione dei valori

Il software libero offre indipendenza, libertà, costi ridotti ed efficienza degli investimenti; molti di questi aspetti si presentano mitigati nel business del software tradizionale.

Il software libero aiuta a restituire alcuni valori al mercato tradizionale: offre indipendenza, libertà, costi ridotti ed efficacia degli investimenti; molti di questi aspetti non sono totalmente garantiti dal business del software tradizionale.

Oltretutto, il software libero permette alle imprese di sistemare la struttura dei costi e di implementare strategie di collaborazione tra fornitori simili o complementari. Questa condizione è più vantaggiosa, più competitiva, meno rischiosa e più efficiente per coloro che partecipano al software libero che per coloro che si affidano al modello di proprietà.

2. La prospettiva del cliente

Per il cliente che utilizza prodotti e servizi basati sul software libero è molto importante saper identificare vantaggi ed inconvenienti legati al modello di software libero piuttosto che a quello di proprietà, soprattutto quando il software libero si inserisce nel contesto di un mercato tradizionale già molto radicato.

Dal punto di vista del cliente di prodotti di software, gli argomenti economici possono sembrare più importanti che non la differenza tecnologica nell'architettura del prodotto. Le imprese devono tenere in stretta considerazione questo aspetto quando elaborano le strategie che dovrebbero consentire loro di entrare nel mercato con garanzie di successo.

Nei prossimi paragrafi approfondiremo vantaggi ed inconvenienti implicati dal business del software libero dal punto di vista del cliente.

2.1. Vantaggi

I vantaggi del software libero per il cliente costituiscono una fetta importante della possibilità di creare un buon business per l'impresa, dal momento che influiscono sul suo posizionamento nel mercato.

Effetti economici

Il software libero assicura al cliente indipendenza dai fornitori della tecnologia, alternative ai prodotti ed ai servizi di proprietà (o perfino altre soluzioni libere), possibilità di sfruttare un'offerta in costante crescita di software legati a soluzioni standard e di un ampio corollario di soluzioni, e la possibilità di godere di situazioni di software intercambiabile (*commoditization*).

Costi

L'aumento dell'efficienza e dell'efficacia nella gestione dei costi della tecnologia può essere assai rilevante per il cliente finale, che sia un privato o un'impresa di qualunque dimensione.

Il software libero permette al cliente di aggiustare la struttura dei costi e degli investimenti in tecnologia mediante una gestione più efficiente ed efficace degli stessi.

Cambiamenti nei costi

Si possono ridurre i costi dell'impianto utilizzando il software libero che viene distribuito gratuitamente oppure attraverso la diminuzione della quantità di aggiornamenti obbligatori degli strumenti in intervalli di tempo brevi. Oltretutto questo risparmio può essere utilizzato per finanziare servizi o investimenti in tecnologia di lungo periodo (ad esempio, per ottenere minori costi di mantenimento del sistema).

Web consigliato

J. García; A. Romeo; C. Prieto (2003). *Análisis Financiero del Software Libre* (cap. 7) <http://www.lapastillaroja.net/capitulos_liberados_pdf/la_pastilla_roja_capitulo_7.pdf>

D'altro canto, il libero accesso al codice fonte favorisce la specializzazione e l'estensione delle applicazioni del software libero per mano del cliente stesso (o per mano di qualche impresa specializzata).

Valori etici

Per alcuni clienti i valori etici legati al software libero, come trasparenza, indipendenza, uguaglianza e cooperazione, possono essere motivo di preferenza, in quanto più adeguati ai loro obiettivi (o all'immagine che vogliono dare di sé).

2.2. Inconvenienti

Nonostante l'adozione del software libero offra evidenti benefici al cliente, lo stesso software presenta anche alcuni inconvenienti che vanno attenuati e tenuti sotto controllo, se l'impresa vuole davvero trarre il massimo vantaggio dalle opportunità di business che questo offre.

Effetti economici

Il cliente può essere reticente ad adottare il software libero a causa dei costi legati al cambio o dei problemi di compatibilità che potrebbero nascere con le soluzioni che già adotta. In alcuni casi alle alternative non si riconosce il valore che meriterebbero, per la ricerca del risultato o del profitto a breve termine, per i miti associati al software libero, o per il vincolo che storicamente lega il cliente al software già in uso.

Gestione del rischio

Ogni innovazione tecnologica all'interno di un'organizzazione ha un certo rischio associato (compreso il cliente privato), così per il software libero come per il software di proprietà. Per il cliente, le varianti che esistono tra le due soluzioni possono essere discriminanti in determinate occasioni, come succede per esempio nel caso di un cliente che ha tentato diverse volte in passato di passare da un software all'altro ed ha sempre trovato ostacoli sul suo cammino.

A volte il cliente può non essere disposto a rischiare con novità che possono compromettere il funzionamento abituale dei suoi processi. Così facendo, preferisce scansare la necessità di apportare degli aggiustamenti in determinate aree della sua attività, anche se queste potrebbero migliorarne l'efficienza, in termini di organizzazione, come sarebbe per il caso di una profonda moder-

nizzazione tecnologica. Se non adeguatamente pianificata, del resto, questa modernizzazione potrebbe essere causa di problemi nelle aree del funzionamento e dell'operatività.

Gestione dei costi

Una parte dei costi di installazione può essere comune al software libero così come a quello di proprietà. A volte il cliente può pensare che dei cambiamenti nella strumentazione utilizzata portino irrimediabilmente con sé dei costi maggiori, legati alla formazione, al supporto e alla motivazione del personale, o ancora alla perdita di produttività dell'organizzazione, ad esempio. Può essere complicato ribattere ad argomentazioni di questo tipo, soprattutto per la difficoltà di darne una corretta quantificazione economica.

3. La strategia d'impresa

Conoscere la prospettiva del cliente, e pertanto quella del mercato obiettivo, è fondamentale per la definizione di una strategia di business solida. Tuttavia, l'impresa deve completare la sua strategia tenendo conto dei vantaggi e degli inconvenienti che le procura il modello del software libero e, più in concreto, il modello di business specifico che utilizza.

L'impresa che decide di sfruttare commercialmente il software libero deve essere consapevole dell'ambiente nel quale opera. Tutte le caratteristiche proprie del software libero, del cliente e del modello di business che utilizza devono essere identificate e analizzate per strutturare al meglio una strategia realista e adeguata al conseguimento degli obiettivi fissati.

In questa parte ci dedichiamo in un primo momento ai vantaggi ed agli inconvenienti del modello del software libero per l'impresa, per poi analizzare le strategie legate ai modelli di business basati sul software libero.

3.1. Il modello di software libero

Come nel caso del cliente, le peculiarità del modello del software libero influiscono tanto sulla definizione del business come sulle possibilità di posizionarsi nel mercato e sulle possibilità di sviluppo dell'impresa nel lungo periodo.

Vantaggi

I principali vantaggi per il fornitore o per la impresa che utilizza con scopo di lucro il software libero sono elencati di seguito.

- **Posizionamento e differenziazione**

il software libero permette di collocare l'impresa che lo utilizza in una posizione favorevole per la pubblicità e il marketing positivo nel mercato, nel senso che la diffusione può essere coerente con obiettivi di consolidamento, fiducia, sostenibilità e popolarità dell'impresa.

- **Mercato**

Nel mercato del software tradizionale può essere difficile identificare e saper sfruttare nuove opportunità di business, a causa degli effetti economici provocati dalle politiche di business tradizionali. In questo ambito, e come abbiamo

già descritto più volte, il software libero favorisce l'introduzione di tecnologie innovative (dirompenti) che consentono di ottenere un carattere di novità che può essere sfruttato per nuove opportunità di business.

Di conseguenza, il software libero agevola la penetrazione di nuove imprese nel mercato tradizionale demolendo gli effetti economici che rendono immobili gli attori presenti nel mercato.

- **Sviluppo e distribuzione**

La libertà, la facilità e il basso costo della distribuzione del software libero (di norma attraverso il download diretto e gratuito su internet), così come la partecipazione, il coinvolgimento e la motivazione della comunità degli utenti nel suo processo di sviluppo, favoriscono la diffusione e l'adozione delle applicazioni. Tanto la metodologia che porta allo sviluppo quanto le peculiarità nella distribuzione delle soluzioni favoriscono l'efficienza e l'efficacia del progetto.

- **Costi e rischi**

La mole e la struttura dei costi e dei rischi per le imprese basate sul software libero sembrano essere più vantaggiose e più competitive che nei modelli basati sul software di proprietà, a causa della distribuzione e della decentralizzazione di parte dei loro processi tra i differenti attori implicati.

- **Commercializzazione**

La commercializzazione del software è una situazione globalmente vantaggiosa per l'insieme degli attori, dal momento che diminuisce le barriere all'entrata per i nuovi produttori di software, aumenta la competitività del settore e, pertanto, gli stessi beni sono prodotti con maggiore efficienza. Oltre a cercare la specializzazione così da poter sfruttare le possibilità di business, è anche possibile muoversi in un mercato totalmente commercializzato.

- **Innovazione e creazione di valore**

Le metodologie di produzione e di sviluppo aperte e cooperative risultano finalmente in una maggiore efficacia ed efficienza, tanto nel processo creativo di innovazione, quanto nella creazione e nel conseguimento di valore da parte dell'impresa. Per meglio dire, con l'apertura dei processi di produzione, l'impresa smette di dipendere esclusivamente dal personale interno per l'innovazione (il quale è soggetto alle limitazioni di tempo e di obiettivi da raggiungere), e comincia invece a beneficiare delle intuizioni e delle prospettive di volontari, utenti e clienti (la cui flessibilità e la cui grande motivazione favoriscono l'ideazione di innovazioni interessanti).

Lettura consigliata

L. Morgan; P. Finnegan (2008). *Deciding on open innovation: an exploration of how firms create and capture value with open source software* (vol. 287, pag. 229-246). IFIP 2008.

In questo modo si chiude il circolo di mutua alimentazione tra impresa e clienti o utenti (trattari come co-sviluppatori), così riducendo il rischio del progetto e massimizzando le garanzie di successo.

Inconvenienti

Presentiamo ora alcuni dei problemi che possono sorgere per le imprese basate sul software libero.

- **Effetti economici**

Limitazioni

Ad esempio, la prigionia dei clienti e l'economia delle idee impediscono che l'impresa possa situarsi in una posizione preponderante sul mercato, come invece potrebbe accadere in alcuni mercati fondati sulle soluzioni proprietarie.

Alcuni degli effetti economici che agevolano l'ingresso di una nuova impresa nel mercato possono essere anche la causa della limitazione nella quantità e nella qualità delle sue operazioni.

- **Risultati**

Una delle conseguenze del punto precedente è che può essere difficile ottenere grandi benefici (per lo meno se si guarda alla misura di quelli ottenuti fino ad oggi dalle corporazioni di software di proprietà) o benefici durevoli nel lungo periodo.

- **Commercializzazione**

La commercializzazione del software può avere anche un impatto negativo sulle imprese basate sul software libero, se queste non identificano e pianificano al meglio la specificità da dare ai loro prodotti e ai loro servizi, o se non adottano una politica di marketing adeguata. Il concetto che vogliamo spiegare è che in una situazione di beni sostituti, se al prodotto non vengono apportati tratti di differenziazione ben marcati, con l'andare del tempo, questo può ripercuotersi sulla composizione e sulla distribuzione del mercato.

Da un altro punto di vista, operare in un mercato commercializzato rende impossibile ottenere grossi margini di profitto, a causa della relativa facilità di cui il cliente dispone per cambiare fornitore di tecnologia. L'impresa deve cioè essere davvero migliore o almeno alla pari della concorrenza del settore per poter mantenere la sua posizione, ad esempio agendo sulla velocità di risposta e sulla capacità di adeguarsi.

- **Mitologia**

Nonostante molto tempo sia passato, è possibile che in alcuni mercati resistano ancora dei miti relativi al software libero che rendono complicata la sua adozione e la sua diffusione. Combattere questi miti può essere più o meno

difficile a seconda delle caratteristiche stesse del mercato, come per esempio il grado di radicazione del software di proprietà o l'eventuale numero di tentativi falliti di adozione del software libero.

3.2. Produzione del software libero

In generale, se risulta che l'applicazione sviluppata ha successo tra i potenziali fruitori, si possono ottenere vantaggi relazionati con l'attrazione di migliorie e accessori, la simpatia di pubblico e comunità, e minori costi di mantenimento grazie alla partecipazione della comunità.

In senso opposto, lo sviluppo di software libero può presentare delle difficoltà nel recuperare l'investimento iniziale, che in alcune occasioni può essere piuttosto significativo. Sebbene sia un problema comune al mondo del software (libero o di proprietà), risulta più difficile vendere copie di software libero che non di altri modelli.

Modelli misti

La presenza di una doppia anima nei modelli misti (generalmente, quella pubblica e quella commerciale) da un lato favorisce l'adozione e la diffusione dell'applicazione, dall'altro porta con sé alcuni inconvenienti, come la poca partecipazione della comunità agli obiettivi d'impresa o la necessità di mantenere interessante un prodotto commerciale nonostante il passare del tempo.

Quest'ultimo aspetto può implicare altri problemi se la gestione della comunità degli utenti da parte dell'impresa non è adeguata, come succederebbe ad esempio se la comunità sviluppasse autonomamente (e in forma pubblica) le estensioni di proprietà della versione commerciale.

Software e servizi

Nel caso della prestazione di servizi associati ad un'applicazione libera, è possibile portare a termine strategie di collaborazione per ampliare il mercato obiettivo, e per poi segmentarlo mediante la differenziazione. Nel caso in cui non sia possibile attuare strategie di collaborazione, il modello presenta poche barriere all'entrata per la concorrenza, la quale, disponendo del codice fonte, può dotarsi della struttura necessaria per competere come in un mercato tradizionale.

D'altro canto, ottenere abbondanti ingressi unicamente a partire dai servizi associati può essere difficile in mercati con forte presenza di innovatori ed entusiasti della tecnologia.

3.3. Prestazione di servizi legati al software libero

La prestazione di servizi presenta alcuni vantaggi rispetto al software di proprietà, come l'assenza di grossi costi derivati dalle licenze, dalla qualità del prodotto e dall'accesso al codice fonte. Queste caratteristiche contribuiscono a fornire servizi in maniera efficiente ed efficace, cosa che finisce per essere un valore aggiunto importante per il cliente.

Nonostante questo, può essere difficile conservare i clienti nel lungo periodo, a causa della facilità ad entrare nel mercato e alla difficoltà di rendere diversi i servizi che sperimentano i fornitori.

Piccole e medie imprese

Le principali occasioni di business provengono dall'esiguo incartamento e dalla distribuzione delle applicazioni basate sul software libero (come l'installazione, il supporto, la personalizzazione o la formazione), inserendosi in nicchie di mercato concrete.

Tuttavia, eventuali sviluppi su misura di un'applicazione concreta possono dover affrontare delle difficoltà sul versante dell'integrazione e della compatibilità con versioni posteriori. Allo stesso modo, può essere anche problematica la comparsa di imprese concorrenti nello stesso settore, a causa del margine ridotto per la collaborazione.

Grandi imprese

Per le grandi imprese, la partecipazione a progetti basati sul software libero può essere relativamente facile grazie alle loro infrastrutture ed alla loro organizzazione. Il software libero consente inoltre di risparmiare sui costi, migliorare l'immagine della marca su caratteristiche come la fiducia, la solidità, la stabilità o il supporto professionale.

Detto quanto sopra, non è comunque facile dare una forma all'immagine di una marca in poco tempo. La predominanza di grandi corporazioni di software proprietario nel mercato complica il posizionamento, ed anche il rischio per progetti di grande dimensione è maggiore.

3.4. Mercati ausiliari

Di norma, i modelli di business associati a mercati ausiliari servono da complemento alle strategie principali. Tuttavia, possono essere adeguati e sostenibili anche come strategia principale in mercati con scarsa concorrenza o con requisiti di differenziazione e specializzazione.

Hardware

Il mercato ausiliare dell'hardware può essere utile per operare in mercati che richiedono specializzazione nei prodotti, come nell'esempio dei servizi integrati, di alto rendimento e di minor costo per il cliente. I mercati di cui parliamo sono quelli per i quali i sistemi di proprietà non provano interesse e così il software libero può rappresentare una differenziazione significativa per il cliente.

I principali inconvenienti provengono dalla capacità di assorbire i costi di produzione e di sviluppo quando il mercato è di piccole dimensioni o esiste una forte concorrenza di prezzo. In alcune occasioni, la difficoltà a recuperare l'investimento iniziale può far sì che sia poco indicato per piccole e medie imprese.

Altri mercati

I mercati ausiliari come quello della vendita di libri o del merchandising possono essere equiparati ai loro omologhi basati sul software di proprietà, tenendo conto delle peculiarità legate al software libero, come per esempio gli aspetti di complementarietà rispetto al prodotto originale o alla diffusione di valori etici.

Riepilogo

Il modello di software libero costituisce un'alternativa valida e solida al software di proprietà, dal momento che basa i suoi meccanismi di concorrenza su caratteristiche molto diverse dal secondo, quali il costo e la flessibilità.

Queste caratteristiche presentano vantaggi ed inconvenienti per i principali attori dediti al mercato del software. A volte, quegli aspetti che sono un vantaggio per alcuni rappresentano un ostacolo per altri, e questo è indice della grande importanza che riveste la necessità di realizzare una strategia di business realista che permetta di raggiungere gli obiettivi con efficienza ed efficacia.

Per sviluppare la strategia, l'impresa basata sul software libero deve tener conto delle implicazioni del modello del software libero sull'atteggiamento del cliente, così come sul suo stesso funzionamento:

- Per il cliente, il software libero è uno strumento che lo aiuta a combattere gli effetti economici di un mercato tradizionale, e a gestire meglio i costi di adozione, a costo di affrontare dei possibili rischi.

- Per l'impresa rappresenta una possibilità di fare business basata sulla differenziazione e sulla gestione efficiente di costi e rischi, pur con la possibilità di vedere ridotti la sua fetta di mercato e i risultati che può ottenere.

La pianificazione della strategia d'impresa permette di godere più e meglio dei vantaggi del software libero nel contesto di un'attività d'impresa, dal momento che tiene sotto controllo e limita gli inconvenienti che possono diminuire le sue possibilità di successo.

Bibliografia

Boyer, M.; Robert, J. (2006). *The economics of Free and Open Source Software: Contributions to a Government Policy on Open Source Software*. Centre Interuniversitaire de recherche en analyse des organisations (CIRANO), 2006RP-03 <http://www.cirano.qc.ca/pdf/publication/2006RP-03.pdf> [Consultazione: marzo 2009].

García, J.; Romeo, A.; Prieto, C. (2003). *Análisis Financiero del Software Libre*. La Pastilla Roja, capitolo 7.

<http://www.lapastillaroja.net/capitulos_liberados_pdf/la_pastilla_roja_capitulo_7.pdf> [Consultazione: marzo 2009]

Ghosh, R. A. UNU-MERIT, N. L. (2006). *Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies sector in the EU*

<http://ec.europa.eu/enterprise/ict/policy/doc/2006-11-20-flossimpact.pdf> [Consultazione: marzo 2009].

Iansiti, M.; Richards, G. L. (2006). *The Business of Free Software: Enterprise Incentives, Investment, and Motivation in the Open Source Community.*

<http://www.hbs.edu/research/pdf/07-028.pdf> [Consultazione: marzo 2009].

Morgan, L.; Finnegan, P. (2008). "Deciding on open innovation: an exploration of how firms create and capture value with open source software". In: G. León; A. Bernardos; J. Casar; K. Kautz; J. de Gross (eds). "International Federation for Information Processing". *Open IT-Based Innovation: Moving Towards Cooperative IT Transfer and Knowledge Diffusion* (vol. 287, pags. 229-246). Boston: Springer.

West, J.; Gallagher, S. (2006). "Patterns of Open Innovation in Open Source Software". En: Henry Chesbrough; Wim Vanhaverbeke; Joel West (eds.). *Open Innovation: Researching a New Paradigm* (pag. 82-106). Oxford: Oxford University Press.

<http://www.openinnovation.net/Book/NewParadigm/Chapters/index.html> [Consultazione: giugno 2008].

Wheeler, D. (2007). *Why Open Source Software?*

<http://www.dwheeler.com/oss_fs_why.html> [Consultazione: marzo 2009]

Il software libero: un nuovo modello economico?

Amadeu Albós Raya

PID_00145045

Universitat Oberta de Catalunya
www.uoc.edu

Indice

Introduzione

In questo modulo esamineremo il concetto di software libero considerandolo come modello economico. Studieremo, per meglio dire, la definizione e la sostenibilità del software libero in qualità di modello di funzionamento economico sostenibile nel lungo periodo.

Nello studio del software libero come modello economico saremo limitati dalla relativa novità del business basato sul software libero. Nonostante ciò, tenendo in considerazione che in generale le regole economiche del mercato non sono state modificate, ci baseremo, nello studio, sulla differenza che distingue il business del software libero dai mercati tradizionali. Questo punto di osservazione ci permetterà di ottenere una prima approssimazione realistica circa le potenzialità del software libero come modello economico.

Inizialmente passeremo in rassegna le basi sostegno del concetto di software libero, il suo funzionamento e le sue possibilità. Ci soffermeremo, cioè, su quelle caratteristiche concettuali legate alla filosofia operativa e sottostante il modello, così come, ad esempio, la produzione sociale.

In seguito analizzeremo le conseguenze del modello basato sul software libero da diverse angolazioni, tenendo conto delle differenze che presenta rispetto ai modelli tradizionali di produzione di software e a quelli di business. La presentazione di questi concetti ci sarà di aiuto per comprendere meglio come potrà inserirsi il modello del software libero nel mercato in un futuro prossimo.

Termineremo poi con uno studio sulla relazione tra il modello del software libero e solidità e forza delle imprese che vi fanno affidamento, dando rilievo all'importanza di coniugare strategia ed opportunità.

Obiettivi

Al termine di questo modulo lo studente dovrà essere in grado di:

1. Avere familiarità con gli aspetti economici del modello legato al software libero.

2. Conoscere i concetti cardine e le implicazioni del modello di software libero rispetto al modello tradizionale.

3. Comprendere la peculiare diversità del modello di software libero e saper valutare la sua capacità di creare valore per il mercato.

4. Saper approfondire la validità e la sostenibilità del modello di software libero ed i modelli di business utilizzabili.

1. Le basi del modello

Del software libero conosciamo molte delle caratteristiche tecnologiche che, in minor od in maggior misura, possono risultare simili a quelle di un qualsiasi software di proprietà. Ovverossia, qualora vi siano, le differenze fondamentali tra il software libero ed il software di proprietà non si riscontrano negli aspetti interni od esterni del prodotto.

A grandi linee, la tecnologia applicata ad un prodotto (ad esempio il disegno, l'architettura o la realizzazione particolare) non giustifica di per sé una differenza sostanziale tra modelli liberi e modelli di proprietà, al meno dal ristretto punto di vista del prodotto finito.

Le principali differenze tra il software libero ed altri modelli di produzione di software (su tutti, quello di proprietà) si rilevano nelle particolarità del progetto di sviluppo, nella comunità degli utenti e nella differenza di valore aggiunto del prodotto.

Queste differenze non si basano su aspetti tecnologici propri dell'applicazione o del software, ma nelle caratteristiche e nelle implicazioni sottostanti la sua produzione. Concentra, cioè, un orientamento particolare alla creazione di valore in prodotti e servizi che differiscono dal punto di vista tradizionale.

Come spiegato nei moduli precedenti, nel corso degli ultimi anni si sono perfezionati modelli di business che sfruttano queste caratteristiche differenziali in un mercato tradizionale. Ad ogni modo, il valore maggiore non ricade nel software in sé, bensì nel capitale che si acquisisce quando se ne adotta l'utilizzo.

Questo capitale caratterizza stabilmente i fondamenti del software libero. Questo significa che il software libero trova il suo fondamento nella produzione sociale e nella cultura in rete, che non solo lo rendono possibile, ma ne potenziano anche le capacità e gli effetti.

Nelle prossime sezioni svilupperemo brevemente entrambi i concetti. Prima esamineremo le principali caratteristiche della produzione sociale e poi ci occuperemo della cultura in rete e della sua influenza sull'economia che sostiene il software libero.

1.1. La produzione sociale

Probabilmente, tanto i progressi nella comunicazione globale quanto la diffusione della tecnologia degli ultimi decenni hanno influito, seppure in diversi modi, su quello che oggi chiamiamo software libero.

Basti osservare la facilità d'accesso all'informazione e la volontà di cooperazione, che non sono caratteristiche del software libero solamente, ma bensì creano la base per lo sviluppo di alternative valide e praticabili in una moltitudine di campi.

Anche se al giorno d'oggi esistono molteplici iniziative più o meno connesse alla produzione sociale, le organizzazioni imprenditoriali ritrovano in questo modello uno strumento per incentivare la creazione e la cattura di valore per i loro modelli di business.

Un esempio di produzione sociale

Wikipedia (http://www.wikipedia.org/).

Yochai Benkler, nel suo libro *The Wealth of Networks*, studia approfonditamente questa questione. Di seguito ci addentreremo in alcuni degli aspetti più rilevanti che caratterizzano la produzione sociale.

Lettura consigliata

L. Morgan; P. Finnegan (2008). *Deciding on open innovation: an exploration of how firms create and capture value with open source software* (vol. 287, pag. 229-246). IFIP.

L'economia dell'informazione

L'informazione è un bene pubblico che ha implicazioni economiche a vari livelli grazie alle tecnologie dell'informazione.

Web consigliato

Y. Benkler (2006). *The Wealth of Networks: How social production transforms markets and freedom.* (http://www.benkler.org/Benkler_Wealth_Of_Networks.pdf).

L'innovazione, intesa come creazione di nuova informazione, può essere difficile in situazioni di limitazione o di controllo, mentre è spesso facilitata dall'apertura e dalla collaborazione nella produzione di informazione, conoscenza e cultura.

In questo senso, la produzione o innovazione verso reti di collaborazione o da pari a pari genera una spirale di opportunità caratterizzate da motivazione ed efficienza, con il supporto della tecnologia.

Reti di collaborazione

In inglese, *Peer-to-Peer*. In questo caso, la parola fa riferimento al funzionamento della comunità e non all'assistenza tecnologica che sostiene la comunicazione.

Lo sviluppo e la diffusione dell'informazione

Lo sviluppo e la diffusione dell'informazione possono seguire cammini diversi a seconda della distribuzione di libertà esistente tra produttori e consumatori. In generale, quanta più libertà viene affidata al produttore, tanta meno ne ottiene il consumatore.

I canali di diffusione dell'informazione influiscono sulla maniera di diffonderla. Anche il verso direzionale della trasmissione e gli obiettivi della stessa influiscono su come viene diffusa l'informazione.

In ogni caso, le licenze ed i brevetti possono restringere il flusso dell'informazione, mentre non si vede perché la crescita della rete dovrebbe renderla frammentaria o limitarla.

Le implicazioni della produzione sociale

Benkler afferma che il nostro modo di percepire la struttura di funzionamento del mondo che ci sta attorno è in piena trasformazione, soprattutto quanto al modo di collaborare ed interagire tra tutti nel processo di condivisione di idee e di conoscenze per creare nuova conoscenza.

1.2. Economia e cultura in rete

Le implicazioni della produzione sociale ultimamente sono diventate più chiare in una moltitudine di campi, soprattutto in quello del software libero. La condivisione e l'interazione di conoscenze e il perfezionamento di idee sono, al giorno d'oggi, una buona maniera per approfondire lo sviluppo di un concetto.

Questa visione della produzione come collaborazione per il raggiungimento di un obiettivo concreto contrasta con la visione più tradizionale del mercato circa idee e conoscenze. In esso l'importanza risiede più nella realizzazione finale del prodotto che nel consenso, nel dialogo o nella qualità.

Nella pubblicazione di David Bollier *When Push Comes to Pull: The New Economy and Culture of Networking Technology* viene esaminato in dettaglio come l'evoluzione della tecnologia dell'informazione ha permesso di creare un nuovo punto di vista in aperto contrasto con la tendenza all'accentramento e la gerarchia del modello tradizionale.

Web consigliato

D. Bollier (2006). *When Push Comes to Pull: The New Economy and Culture of Networking Technology.* (http://www.aspeninstitute.org/atf/cf/%7bDEB6F227-659B-4EC8-8F84-8DF23CA704F5%7d/2005InfoTechText.pdf).

Nei successivi paragrafi esamineremo brevemente le principali caratteristiche economiche e culturali della cultura in rete che considera Bollier.

Il modello *push* e il modello *pull*

Il modello *push* si basa sulla produzione di massa. Si anticipa la domanda dei consumatori e si gestiscono in maniera dinamica i tempi e la logistica delle risorse della produzione.

Il modeilo *pull* si basa sull'apertura e sulla flessibilità dei programmi di produzione che vengono utilizzati come risorse. Non anticipa la domanda dei consumatori, ma personalizza i prodotti in funzione della domanda mediante processi rapidi e dinamici.

Reti di creazione di valore

Nei modelli *pull*, il fatto di condividere tanto la informazione quanto le buone procedure migliora in maniera sostanziale il corpus delle conoscenze di tutti i membri della rete.

Questa rete stimola e rende coesi modelli di business aperti, strutturati sulla creazione di valore e sulla personalizzazione o differenziazione dei prodotti.

In questo contesto, i programmi del modello *pull* plasmano, migliorano e danno flessibilità all'innovazione e all'evoluzione attraverso la comunità, senza dover sostenere i costi che, a condizioni simili, sarebbero inevitabili in un modello di tipo *push*.

Il mercato obiettivo

I modelli *push* hanno successo nei settori in cui i consumatori non hanno molto chiaro quello che cercano e preferiscono affidarsi a tipologie predefinite.

Per contro, nei modelli *pull*, i consumatori vogliono avere voce nel processo di elaborazione e selezione, nel senso che magari non sanno esattamente cosa vogliono, però sono certi di voler partecipare nel processo.

La produzione

I modelli *push* tendono a cercare alternative di produzione che possano risultare economicamente più competitive (ad esempio, minimizzando i costi di produzione), mentre i modelli *pull* tendono piuttosto alla ricerca delle soluzioni più atte ad apportare valore alla rete di produzione.

Questo specifico orientamento dei modelli *pull* favorisce la possibilità di risalire nella scala della rete di produzione, così come anche il confronto tra i migliori partecipanti, per ottenere la migliore specializzazione della produzione.

La cooperazione

Nei modelli *pull* si favorisce la creazione di una rete di relazioni basate sulla fiducia, sulla condivisione delle conoscenze e sulla collaborazione tra i membri della rete per il beneficio di tutti.

In svariate occasioni, questa particolare filosofia sfocia in un regime di governo collettivo, allo scopo di gestire con giustizia ed allo stesso tempo efficacia le risorse comuni.

Per questo, le imprese che operano attraverso modelli *pull* devono garantire il riconoscimento di tutti i membri della rete, dato che il modello si fonda su fiducia e creazione di valore.

L'educazione

I modelli *push* permettono agli studenti di concentrare i loro sforzi sulla costruzione di conoscenze statiche, che servano ad abituarsi alla successiva società gerarchizzata.

I modelli *pull* promuovono forme di educazione alternative, in quanto le tecnologie dell'informazione rendono possibile che gli studenti entrino in un vortice di attività dinamiche e possano avere accesso ad un massiccio insieme di risorse indipendenti tale da potersi creare il proprio corpus di conoscenze (e di metterlo a loro volta a disposizione della condivisione).

2. Le caratteristiche del modello del software libero

I fondamenti sui quali poggia il software libero danno forma ad una struttura nella quale la collaborazione e la condivisione di conoscenza tra i suoi membri permettono di innovare, modificare e far progredire la conoscenza globale.

Senza alcun dubbio, la creazione di valore è un obiettivo importante per tutti i membri della comunità (che siano utenti, sviluppatori, ecc.) e per il modello in se stesso. Per questo, la decentralizzazione, la libertà e l'indipendenza che reggono la comunità offrono solide garanzie per consolidare e compattare tanto la produzione quanto il capitale sociale.

Il modello del software libero trova il suo fondamento nella differenza con i valori che reggono il mercato tradizionale, dal punto di vista dello sviluppo del software così come da quello della valutazione del valore creato.

Se pure è vero che da un punto di vista classico alcune caratteristiche del modello del software libero sono applicabili anche ad altri schemi di sviluppo e di creazione di valore, il modello del software libero introduce delle novità sostanziali nella percezione e nella valutazione dei valori connessi al mercato tradizionale.

In questa parte del modulo esamineremo le caratteristiche del modello del software libero mettendole a confronto con quelle di un modello tradizionale, allo scopo di dare risalto alla piena novità che propone il modello nel suo svolgimento quotidiano.

Nel primo paragrafo esamineremo il modello nell'ottica dello sviluppo del software, per poi passare, nel secondo, all'analisi delle implicazioni della sua novità, intesa come schema basato sulla produzione sociale.

2.1. Lo sviluppo del software

La metodologia utilizzata nel lavoro di sviluppo del software libero è probabilmente uno dei fattori universalmente ritenuti caratterizzanti la differenza con gli altri schemi di sviluppo di software, ad esempio con il modello di proprietà. Ma è davvero così?

Dal punto di vista di produzione di software, lo sviluppo di software libero ha importanti punti di contatto con altri modelli di sviluppo, ad esempio con il modello di proprietà, dato che i metodi di produzione hanno una certa indipendenza relativamente alle singole realizzazioni.

Il fatto però che la produzione di software possa risultare più o meno coincidente con altri modelli, o che alcuni dei requisiti di libertà rispetto al codice siano più o meno necessari nella pratica, non implica che non possa esistere una differenza significativa sotto altri aspetti, la quale permetta di attribuire all'insieme un carattere di novità.

A proposito di questo contesto, l'articolo di Fuggetta intitolato *Software libre y de código abierto: ¿un nuevo modelo para el desarrollo de software?* analizza in dettaglio questo ed altri aspetti che differenziano il modello di sviluppo del software libero dal modello di sviluppo del software di proprietà. Nei seguenti paragrafi si espongono brevemente alcune delle sue conclusioni.

Web consigliato

A. Fuggetta (2004). *Software libre y de código abierto: ¿un nuevo modelo para el desarrollo de software?* (http://alarcos.inf-cr.uclm.es/doc/ig1/doc/temas/4/IG1-t4slibreabierto.pdf)

Il contesto

Il successo del software libero può essere attribuito ad una complessa varietà di aspetti tecnologici ed economici connessi all'innovazione ed alla produzione del modello stesso.

Le caratteristiche di decentralizzazione, collaborazione e libertà di uso e di sfruttamento trasformano il software libero nel cavallo di battaglia di una nuova filosofia che si avvicina a problematiche di diversa natura e tenta di risolverle.

Secondo Fuggeta, molte delle convinzioni relative al software libero possono essere attribuite anche al software di proprietà, ed è pertanto conveniente realizzare uno studio esaustivo della tematica.

Il processo di sviluppo

Sotto il profilo tecnologico, lo sviluppo del software libero non implica un nuovo schema, dato che la maggioranza dei progetti sono maneggiati da un numero limitato di collaboratori, mentre le metodologie di sviluppo incrementale e di sviluppo evolutivo non sono esclusive del software libero.

Per contro, il software libero è riuscito a motivare tanto gli sviluppatori quanto gli utenti ad inserirsi nel progetto, riuscendo a far coincidere lo sviluppo e l'evoluzione del software con le necessità della comunità.

La protezione dei diritti dei clienti

I problemi relativi alla protezione dei clienti compaiono principalmente quando si ha a che fare con un pacchetto di software, dato che negli sviluppi fatti su misura il cliente è già proprietario del codice.

Nel caso di pacchetti di software potrebbe essere sufficiente avere accesso al codice fonte senza poterlo modificare o passarlo ad altri. D'altra parte, il supporto all'utente da parte dell'impresa dovrebbe reggersi su norme che facilitino la consegna del codice nel caso in cui l'impresa non si possa fare carico del suo mantenimento.

La diffusione della conoscenza

La diffusione della conoscenza mediante l'accesso al codice fonte è insufficiente, se è vero, come è vero, che le materie relazionate all'ingegneria del software dimostrano che è necessario disporre di documenti che descrivano l'architettura del software.

D'altro canto, se fosse possibile diffondere questa conoscenza, sarebbe sufficiente la pubblicazione del suo codice fonte (tolto il diritto di copiare e ridistribuire il software).

Il costo

Il fatto che il software sia divulgato con una licenza libera non implica che non possa essere commercializzato o che il suo sviluppo non abbia dei costi associati (anche se non ne conosciamo l'entità).

D'altra parte, che non si possano quantificare i costi e che non si possano assegnare ad un unico centro di spesa, questo non significa che gli stessi non siano distribuiti tra i collaboratori, tenendo in considerazione anche l'ipotesi di distribuzione per via indiretta ad imprese che poco o niente hanno a che fare con il mondo del software.

La validità del modello di business

I principali modelli di business che sfruttano realmente il software libero sono quelli che si dedicano allo sviluppo ed alla distribuzione di pacchetti puri con codice aperto, così come quelli che sfruttano le piattaforme di software libero e le piattaforme di proprietà. Altri approcci al business si combinano più o meno bene tanto al software libero come a quello di proprietà.

D'altra parte, fino ad oggi non abbiamo nulla che indichi che un'impresa basata unicamente sui servizi possa essere redditizia in maniera duratura nel tempo.

Sulla redditività

Nel libro *The Business of Software*, Michael Cusumano afferma che le compagnie di software dipenderanno sempre più dalla combinazione delle entrate tra licenze e servizi.

L'industria del software

L'Europa non dispone di una strategia industriale che permetta l'azione coordinata delle diverse imprese coinvolte. In questo senso, sostenere il software libero non è una strategia, rispetto alla creazione di prodotti innovativi.

2.2. Il paradigma cooperativo

Anche se alcune delle caratteristiche del modello di software libero non sono innovative secondo la prospettiva classica, come abbiamo visto, lo sono invece quelle che causano esattamente un cambiamento nella prospettiva del mercato.

Per assegnare il giusto valore all'originalità del modello di software libero rispetto agli altri modelli tradizionali, bisogna prendere in considerazione gli aspetti della produzione e della creazione di valore, oltre che quelli delle conoscenze sulle quali si fonda il modello.

Nell'articolo *Open Source Paradigm Shift*, Tim O'Reilly analizza queste ed altre caratteristiche del software libero, considerandole come innovatrici e capaci di creare un vantaggio competitivo che può essere sfruttato a scopo di lucro. Qui di seguito passeremo in rassegna alcune delle sue conclusioni.

Web consigliato

T. O'Reilly (2004). *Open Source Paradigm Shift*.
(http://www.oreillynet.com/pub/a/oreilly/tim/articles/paradigmshift_0504.html).

Il cambiamento

Il software libero suppone un cambiamento profondo nella struttura del mercato di riferimento, che, in svariate occasioni, ha implicazioni e conseguenze che vanno oltre quelle immaginate dai suoi creatori.

I cambiamenti si manifestano nella qualità del prodotto, nell'abbassamento dei costi di produzione e nello sfruttamento dei modelli standard, così come nelle novità nelle aree marketing, distribuzione e logistica.

Il software come *commodity*

Se consideriamo un contesto in cui la comunicazione permanente è standard, come quello in cui viviamo, tutte le applicazioni che veicolano la comunicazione sono intercambiabili tra loro (ad esempio, un navigatore web). Questo per dire che lo sfruttamento di modelli standard fa sì che il software può essere considerato come una *commodity*.

Per questo, quando la capacità di un'applicazione di generare profitti si esaurisce a causa del processo di *comoditization*, in quel momento compare un nuovo mercato per i prodotti di proprietà, soprattutto quando questi utilizzano la rete di comunicazione globale.

Nonostante questo, il software libero continua ad essere un modello affidabile per le imprese di servizi, anche se non ci si può aspettare un margine di guadagni pari a quello delle grandi compagnie di software.

Collaborazione in rete

La cultura dello scambio del software è cresciuta fin dai suoi inizi allo stesso ritmo di Internet, che sperimenta ogni giorno di più una maggiore partecipazione in praticamente tutte le sue funzioni.

Il software libero costituisce il linguaggio naturale della comunità in rete, dando luogo ad uno stile di collaborazione e di partecipazione da parte di ognuno dei suoi membri. Questa collaborazione è fondamentale per il successo e per gli aggiornamenti delle applicazioni leader di Internet, dato che ha dato risalto all'importanza del trattare gli utenti come co-sviluppatori del software.

Personalizzazione e *software-as-service*

Al giorno d'oggi siamo ormai abituati a considerare un'applicazione non come un processo, ma piuttosto come un dato immobile. I programmi hanno bisogno di ingegneria per essere creati, ma i linguaggi dinamici che permettono la coesione delle componenti (ad esempio, degli *scripts* di gestione dei dati) forniscono la prospettiva di un processo dinamico ed in evoluzione per l'applicazione.

Buona parte dei servizi che vengono offerti su Internet (ad esempio, un motore di ricerca) ha bisogno di revisioni ed aggiornamenti costanti per poter assolvere correttamente il suo compito. Questa situazione genera un nuovo modello di business legato al mondo dei computer e della tecnologia dell'informazione, in generale, ed in particolar modo all'utilizzo del software come servizio.

Il sistema operativo Internet

Possiamo trattare Internet come se fosse un grande ed unico computer virtuale che costituisce il sistema operativo partendo dall'unione di tanti piccoli pezzi e che consente la partecipazione di chiunque nella creazione di valore.

I valori della comunità degli utenti del software libero sono importanti per il modello, in quanto promotori di uno spirito di ricerca e di condivisione delle scoperte.

Il processo di *commodization* della tecnologia è parte del processo che permette all'industria di progredire nella direzione di creare un maggior valore per tutti. Per l'industria è essenziale trovare l'equilibrio che le permetta di creare un valore maggiore di quello che si otterrebbe se la partecipazione fosse più semplicemente individuale.

3. L'efficacia e la validità del modello di software libero

Nei sottocapitoli precedenti abbiamo esaminato tanto i fondamenti che stanno alla base del modello del software libero, quanto le caratteristiche che lo rendono diverso dai modelli più tradizionali.

Per assegnare un valore alla sostenibilità nel lungo periodo del modello del software libero ci servono molti più dati di quelli che sono ad oggi disponibili; ci manca, cioè, la possibilità di osservare uno spazio temporale molto più ampio, che ci dia maggiori strumenti per poter confrontare il modello con quelli più tradizionali in modo più preciso.

Il passare del tempo sarà il discrimine che determinerà se il software libero costituisce realmente un nuovo modello economico, e quali saranno le caratteristiche e le condizioni del modello che lo renderanno possibile.

Facciamo in merito delle considerazioni. A dispetto del fatto che il business basato sul software libero sia al momento della stesura di questo testo relativamente recente, abbiamo cercato di dare risalto alle differenze che permettono di avere una nuova prospettiva del business basata principalmente sul potenziamento della produzione in forma di cooperazione della conoscenza.

L'applicazione basata sul software libero

La produzione su base allargata di una applicazione o di una soluzione concreta favorisce la creazione di valore senza alcun impatto negativo per i costi di produzione, cosa che assegna un vantaggio competitivo rispetto ad altre alternative del mercato.

Le applicazioni basate sul software libero, assieme con l'apertura dei modelli standard, possono limitare alcuni degli effetti economici che rafforzano i prodotti basati sul modello tradizionale. In questo modo, oltre ad indurre una differenza sostanziale rispetto alle applicazioni tradizionali, rendono possibili strategie e politiche di concorrenza tra imprese secondo un profilo vincitore-vincitore.

Il mercato

La produzione sociale ha riempito Internet di iniziative alternative ai modelli tradizionali. Il capitale sociale è diventato, con il passare del tempo, un valore importante per l'innovazione e lo sviluppo in ambienti aperti. Oggi esistono modelli di business lucrativi che remunerano la produzione di conoscenza.

Il business della conoscenza

Innocentive (http://www.innocentive.com/), tra gli altri, è un portale web dedicato a ricompensare le idee che diano una soluzione a problemi concreti. Per meglio dire, vi partecipano utenti che propongono problemi (*seekers*)) ed altri che li risolvono (*solvers*) in cambio di un riconoscimento economico

Questo ed altri esempi hanno dato avvio alla creazione di una nuova logica di mercato, chiamata in alcuni contesti wikinomìa e *crowdsourcing*. Questa logica si basa sul modello *pull* che abbiamo precedentemente spiegato, ovvero, sulla attrazione di idee e di sforzi messa a confronto con il tradizionale modello *push*.

Col tempo scopriremo se questa prospettiva di mercato consentirà l'evoluzione dei protettori del mercato tradizionale verso una nuova situazione, nel mondo della tecnologia.

Il business

La nuova prospettiva di mercato può schiudere nuove opportunità di business legate all'utilizzo a scopo di lucro di idee, concetti e conoscenze, senza che si debba esserne i proprietari. Questo significa che il valore dell'applicazione basata sul software libero non risiede nella soluzione in se stessa, ma bensì nel capitale che si acquisisce ed in quello che si può generare attraverso l'utilizzo del software libero.

Nonostante questo, l'efficacia e l'affidabilità del software libero come modello traggono sostentamento anche dalla particolare concezione dell'impresa che lo utilizza. Sembra fondamentale concepire e vivere l'impresa come una opportunità di business solida e duratura.

I rischi

Senza dubbio, i principali rischi per il modello basato sul software libero sono: ottenere una massa di utenti che renda sostenibile il progetto, e gettare le fondamenta di un modello solido ed affidabile nel tempo. Bisogna poi tener conto del rapporto tra investimento iniziale e benefici attesi.

Lo studio di sostenibilità dell'impresa

L'analizzare, disegnare e dare forma all'impresa in maniera completa ed esaustiva ci permette di accrescere le garanzie di successo del business basato sul software libero. Per massimizzare queste garanzie, la sostenibilità dell'impresa va progettata e studiata prima della sua costituzione, e va precisata in un piano di azione dell'impresa.

Guarda anche

Nel terzo modulo di questo corso si è già fatta una prima approssimazione delle principali caratteristiche legate alla sostenibilità imprenditoriale del business classico del software, ad esempio gli aspetti di commercializzazione e di marketing, così come i prodotti ed i servizi oggetto del business imprenditoriale.

Nell'impresa basata sul software libero dobbiamo completare gli aspetti precedenti con le caratteristiche dei modelli di business basati sul software libero che vedremo nel quarto modulo, di modo che la loro combinazione dia luogo alla possibilità di realizzare una base solida sulla quale edificare un business sostenibile.

L'impresa di software libero

Come in qualunque altro modello aziendale, anche l'impresa basata sul software libero richiede un disegno ed una progettazione dettagliata prima dell'avvio della sua attività. Nei paragrafi precedenti abbiamo sottolineato l'importanza dell'analizzare con cura i fondamenti commerciali dell'impresa come condizione per valutare la sua validità e la sua sostenibilità.

Tanto i fondamenti del software libero come le implicazioni che abbiamo visto in dettaglio in questo modulo possono esercitare forze differenti in funzione della tipologia dell'opportunità di business e del contesto nel quale si vuole lavorare.

Di conseguenza, la strategia dell'impresa basata sul software libero può e deve modificare le sue azioni considerando l'originalità del business, gli effetti economici sul suo ambiente, il capitale e la produzione sociale, e la concorrenza.

Riepilogo

In questo modulo abbiamo esaminato le caratteristiche del software libero come modello economico, sempre tenendo in considerazione la scarsità di dati disponibili, dovuta alla relativa novità dei modelli di business nati sullo schema del software libero.

Da un lato, i fondamenti del capitale sociale e la produzione collettiva di idee e conoscenze non sono esclusivi del software libero. Ai giorni nostri esistono diverse iniziative che dimostrano quanto possano risultare fattibili la cooperazione e la collaborazione nell'innovazione e nella produzione di conoscenza.

Questi fondamenti fanno notare l'importanza della rete di collaboratori, la sua implicazione e la sua spinta verso un progresso globale ed individuale dei membri della comunità, e rappresentano un'alternativa credibile ai modelli tradizionali di produzione.

Per altro verso, le implicazioni della filosofia della produzione sociale possono essere posti sotto diverse lenti di ingrandimento. Se anche è vero che certe caratteristiche del software libero non rappresentano una differenza significativa rispetto ad altri modelli, esistono altre caratteristiche che sono fortemente innovative.

Se si considera il business del software libero, risulta di primaria importanza rinforzare e servirsi degli aspetti originali del software libero per offrire alternative valide ed affidabili rispetto ai modelli tradizionali. Queste azioni devono essere necessariamente accompagnate dallo studio e dalla progettazione dettagliata del business, perché ne venga garantita la sostenibilità presente e futura.

Bibliografia

Benkler, Y. (2006). *The wealth of networks: How social production reforms markets and freedom*. New Haven: Yale University Press. <http://www.benkler.org/Benkler_Wealth_Of_Networks.pdf; WIKI:http://cyber.law.harvard.edu/wealth_of_networks/Main_Page> [Consultazione: marzo 2009]

Bollier, D. (2006). *When Push Comes to Pull: The New Economy and Culture of Networking Technology.* <http://www.aspeninstitute.org/atf/cf/%7bDEB6F227-659B-4EC8-8F84-8DF23CA704F5%7d/2005InfoTechText.pdf>

Fogel, K. (2004). *The Promise of the Post-Copyright World* <http://www.questioncopyright.org/promise> [Consultazione: marzo 2009]

Fuggetta, A. (set.-ott., 2004): *Open Source and Free Software: A New Model for the Software Development Process?* (num. 171). Novática – Upgrade: Monografia del processo del software, inglese <http://www.upgrade-cepis.org/issues/2004/5/up5-5Fuggetta.pdf) | spagnolo: (http://alarcos.inf-cr.uclm.es/doc/ig1/doc/temas/4/IG1-t4slibreabierto.pdf> [Consultazione: febbraio 2009]

Goldhaber, M. (giugno, 2006). *The Value of Openness in an Attention Economy* (vol. 11, num. 6). <http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1334/1254> [Consultazione: marzo 2009]

Moglen, E. (1999). *Anarchism Triumphant and the Death of Copyright*. <http://www.uic.edu/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/684/594> [Consultazione: marzo 2009]

Morgan, L.; Finnegan, P. (2008). *Deciding on open innovation: an exploration of how firms create and capture value with open source software*. En: G. León; A. Bernardos; J. Casar; K. Kautz; J. DeGross (ed.). International Federation for Information Processing. Open IT-Based Innovation: Moving Towards Cooperative IT Transfer and Knowledge Diffusion (vol. 287, pag. 229-246). Boston: Springer.

O'Reilly, T. (2004). *Open Source Paradigm Shift.* <http://tim.oreilly.com/articles/paradigmshift_0504.html> [Consultazione: febbraio 2009]

Appendice

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in per-

forming those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered

work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of

the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license

document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>
```

```
    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type 'show c' for details.
```

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

WITH SUPPORT FROM THE
Education and Culture DG
Lifelong Learning Programme

IL TESTO ASPETTI ECONOMICI E MODELLI DI BUSINESS DEL SOFTWARE LIBERO SI PROPONE L'OBIETTIVO DI FORNIRE LE CONOSCENZE NECESSARIE A COMPRENDERE E A METTERE IN PRATICA L'ECONOMIA DEL SOFTWARE LIBERO MEDIANTE LO STUDIO E L'ANALISI DEGLI ASPETTI ECONOMICI E DEI MODELLI DI BUSINESS AD ESSO LEGATI, COSÌ COME MEDIANTE LO STUDIO E L'ANALISI DELLE OPPORTUNITÀ CHE OFFRE QUESTO NUOVO MERCATO.

freeknowledgeinstitute
UOC Universitat Oberta de Catalunya www.uoc.edu
www.ou.nl
UNIVERSITETET I AGDER