

# Cosa è Web 2.0

## Design Patterns e Modelli di Business per la Prossima Generazione di Software

[articolo originale di Tim O'Reilly](#)

Lo scoppio della bolla dot-com nell'autunno del 2001 ha segnato un punto di svolta per la rete. Molte persone sono giunte alla conclusione che la rete fosse assolutamente sopravvalutata, quando, invece, [le bolle e le conseguenti crisi sembrano essere una caratteristica comune di tutte le rivoluzioni tecnologiche](#). Le crisi normalmente segnano il punto in cui una tecnologia in crescita è pronta a prendere il posto che le spetta, al centro del palcoscenico. I simulatori vengono eliminati, le storie di effettivo successo mostrano la loro forza e qui si inizia a comprendere cosa separa le une dalle altre.

Il concetto di "Web 2.0" ebbe inizio con una sessione di brainstorming durante una conferenza tra O'Reilly e MediaLive International. Dale Dougherty, pioniere del web e Vice-Presidente di O'Reilly, fece notare che, tutt'altro che "crollata", la rete era più importante che mai, con nuove interessanti applicazioni e siti nascenti con una sorprendente regolarità. Inoltre, le società che erano sopravvissute al collasso, sembravano avere alcune caratteristiche in comune. Poteva essere che il collasso delle dot-com avesse segnato per la rete un punto di svolta tale che un richiamo all'azione definito come "Web 2.0" potesse avere senso? Concordammo con questa analisi e così nacque la [Conferenza Web 2.0](#).

Nell'anno e mezzo trascorso da allora, il termine "Web 2.0" ha decisamente preso piede, con oltre 9,5 milioni di citazioni in Google. Ma c'è ancora un [grande disaccordo circa il significato di Web 2.0](#), alcuni lo denigrano, considerandolo un termine di marketing, alla moda, ma insignificante, mentre altri lo accettano come il nuovo standard convenzionale.

Questo articolo ha lo scopo di tentare di fare chiarezza su cosa noi intendiamo con Web 2.0.

Nel nostro brainstorming iniziale, abbiamo formulato il significato che per noi ha il concetto di Web 2.0 attraverso degli esempi:

<b>Web 1.0</b>		<b>Web 2.0</b>
DoubleClick	-->	Google AdSense
Ofoto	-->	Flickr
Akamai	-->	BitTorrent
mp3.com	-->	Napster
Britannica Online	-->	Wikipedia
Siti personali	-->	blogging
evite	-->	upcoming.org e EVDB
Ricerca nomi dominio	-->	Ottimizzazione dei motori di ricerca
page views	-->	cost per click
screen scraping	-->	web services
pubblicazione	-->	partecipazione
sistemi di gestione dei contenuti	-->	wikis
directories (tassonomia)	-->	tagging ("folksonomia")
stickiness	-->	syndication

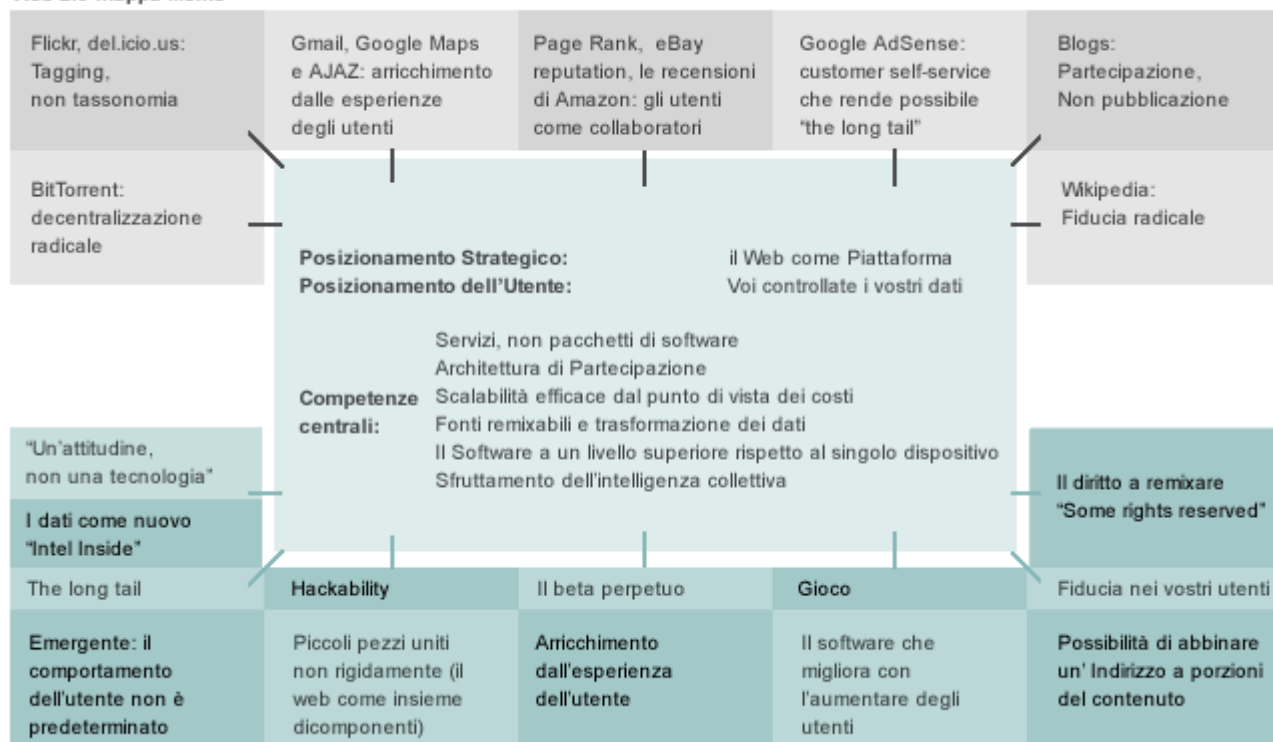
La lista continuava a lungo. Ma cosa ci ha portato a identificare e a definire un'applicazione "Web

1.0" e l'altra "Web 2.0"? (Porsi la questione è particolarmente urgente perchè il termine Web 2.0 è diventato così diffuso che le società ora lo stanno utilizzando un po' a sproposito, senza comprenderne effettivamente il significato. La domanda è particolarmente difficile in quanto molti di queste startup che si circondano di paroloni, sono decisamente *non* Web 2.0, mentre alcune delle applicazioni che abbiamo identificato come Web 2.0, quali Napster e BitTorrent, non sono propriamente applicazioni per il web!). Abbiamo provato a sbrogliare i principi che si sono dimostrati alla base, in un modo o in un altro, delle storie di successo del web 1.0 e delle più interessanti tra le nuove applicazioni.

## Il Web Come Piattaforma

Come molti concetti importanti, il Web 2.0 non ha confini rigidi, ma un'anima gravitazionale. Potete visualizzare il Web 2.0 come un insieme di principi e di procedure che collegano un autentico sistema solare di siti che dimostrano in toto o in parte di questi principi, a una distanza variabile dal centro stesso.

Web 2.0 mappa Meme



La **Figura 1** mostra una “mappa meme” del Web 2.0 sviluppata nel corso di una sessione di brainstorming durante un FOO Camp, una conferenza tenutasi presso O'Reilly Media. Si tratta, per molti versi, di un work in progress, ma mostra le molte idee che scaturiscono dal cuore del Web 2.0.

Per esempio, nella prima conferenza sul tema Web 2.0, nell'ottobre 2004, John Battelle e io, nel nostro discorso di apertura, abbiamo elencato una serie di principi preliminari. Il primo di questi principi fu "Il web come piattaforma". Però, quello era anche il grido di guerra dell'eroe del Web 1.0, Netscape, che crollò in fiamme dopo una battaglia sanguinosa con Microsoft. Inoltre, due dei nostri esemplari iniziali del Web 1.0, DoubleClick e Akamai, erano entrambi dei pionieri nel trattare il web come una piattaforma. La gente spesso non pensa ad esso come “web service”, ma in effetti, l'ad serving fu il primo web service ampiamente diffuso e il primo "mashup" (per usare un altro termine molto utilizzato negli ultimi periodi). Ogni banner pubblicitario è erogato in maniera trasparente da due siti web, che forniscono un'unica pagina all'utente. Anche Akamai tratta il network come una piattaforma e, a un livello più profondo dello stack, costruisce una cache

trasparente e una rete di distribuzione dei contenuti che semplifica le dinamiche di congestione della banda.

Nonostante tutto, questi pionieri hanno fornito utili contrasti, grazie ai quali chi è arrivato successivamente, hanno sviluppato la loro soluzione allo stesso problema, comprendendo qualcosa di più profondo circa la natura della nuova piattaforma. Sia DoubleClick, sia Akamai furono pionieri del Web 2.0, e quindi possiamo anche vedere come possa essere possibile realizzare la maggior parte delle possibilità abbracciando gli ulteriori design patterns del Web 2.0.

Approfondiamo per un momento ognuno di questi tre casi, cercando di analizzare alcuni degli elementi essenziali di differenza.

## **Netscape vs. Google**

Se Netscape era l'archetipo del Web 1.0, Google è certamente l'archetipo del Web 2.0, non fosse altro che per il fatto che le loro rispettive IPO hanno definito gli eventi di ciascuna era. Quindi, iniziamo una comparazione tra queste due società e il loro posizionamento.

Netscape diede forma al "web come piattaforma" nei termini del vecchio paradigma dei software: tra i prodotti, il loro fiore all'occhiello era il browser web, un'applicazione desktop, e la loro strategia era quella di usare il loro predominio nel mercato dei browser per stabilire un mercato di prodotti server di fascia alta. Il controllo sugli standard relativi alla visualizzazione dei contenuti e delle applicazioni nel browser avrebbero dovuto, in teoria, dare a Netscape il tipo di potere di mercato goduto da Microsoft nel mercato dei PC. Così come "carrozza senza cavalli" identificava l'automobile per estensione del concetto più familiare di carrozza, Netscape promosse un "webtop" in sostituzione del desktop, e programmò di popolare quel webtop con aggiornamenti di informazioni e applet, inseriti sul webtop dai fornitori di informazioni che avrebbero acquistato i server di Netscape.

Alla fine, sia i web browser, sia i web server si dimostrarono commodities, mentre crebbe il valore dei servizi forniti su piattaforma web.

Google, invece, iniziò la sua vita come un'applicazione web nativa, mai concepita come un pacchetto in vendita, ma fornita come servizio, con i clienti che pagavano, direttamente o indirettamente, per l'uso di quel servizio. Nessuna delle infrastrutture della vecchia industria del software era presente. Nessuna release di software programmata, ma solo miglioramenti continui. Nessuna licenza o vendita, solo utilizzo. Nessun porting su piattaforme diverse affinché i clienti potessero utilizzare il software sulle proprie macchine, solo un'enorme collezione scalabile di PC, ormai commodity, funzionante con sistemi operativi open source oltre ad applicazioni e utility locali che nessuno al di fuori della società potrà mai vedere.

Google richiede una competenza che Netscape non ha mai richiesto: la gestione dei database. Google non è semplicemente un insieme di strumenti di software, ma è un database specializzato. Senza i dati, gli strumenti sono inutili; senza il software, i dati non possono essere gestiti. Le licenze software e il controllo delle API—la leva di potere nell'era precedente—sono irrilevanti perché il software non avrà mai bisogno di essere distribuito, ma solo utilizzato anche perché senza l'abilità di raccogliere e gestire i dati, il software serve a poco. Infatti, il valore del software è proporzionale alla scala e al dinamismo dei dati che esso aiuta a gestire.

Il servizio di Google non è un server—sebbene sia erogato tramite un'immensa rete di server internet—né un browser—sebbene sia percepito dall'utilizzatore all'interno del browser. Né il suo eccellente servizio di ricerca ospita i contenuti che consente di trovare ai suoi utenti. Analogamente a una telefonata, che non avviene solo ai capi dei due telefoni coinvolti, ma anche sulla rete tra i punti, Google si trova nello spazio tra il browser e il motore di ricerca e il server di destinazione dei

contenuti, come uno strumento o un intermediario tra l'utente e la sua esperienza online.

Sebbene sia Netscape che Google possano essere descritte come società di software, è chiaro che Netscape apparteneva allo stesso mondo del software cui appartengono Lotus, Microsoft, Oracle, SAP, e altre società che iniziarono la loro attività nel corso della rivoluzione del software degli anni 80, mentre Google è più simile alle applicazioni internet quali eBay, Amazon, Napster, e sì, DoubleClick e Akamai.

## **DoubleClick vs. Overture e AdSense**

Come Google, DoubleClick è un vero figlio dell'era internet. DoubleClick tratta il software come servizio, ha una competenza di base nella gestione dei dati e, come indicato in precedenza, è stato un pioniere nei web services molto prima che avessero un nome. Però, DoubleClick fu alla fine limitata dal suo modello di business. Condivise la convinzione degli anni 90 secondo la quale il web significava pubblicare e non partecipare, che gli inserzionisti e non i consumatori dovessero comandare; che le dimensioni contassero e che internet fosse sempre più dominato dai siti principali secondo le valutazioni di MediaMetrix e di altre società che misuravano l'advertising online.

Come risultato, DoubleClick orgogliosamente cita sul suo sito "oltre 2000 implementazioni di successo" del suo software. Yahoo! Search Marketing (originariamente Overture) e Google AdSense, invece, servono già centinaia di migliaia di inserzionisti.

Il successo di Overture e di Google deriva dalla comprensione di quello che Chris Anderson definisce long tail ("lunga coda"), il potere collettivo dei piccoli siti che costituiscono il grosso del contenuto del web. L'offerta di DoubleClick richiedeva un formale contratto di vendita, limitando il proprio mercato a poche migliaia di siti tra i più grandi. Overture e Google hanno compreso come consentire il posizionamento delle pubblicità su, virtualmente, qualsiasi pagina web. Inoltre, hanno evitato i formati pubblicitari tipici delle agenzie pubblicitarie quali banner e popup, preferendo testi pubblicitari minimamente invadenti, sensibili al contesto, e consumer-friendly.

La lezione di Web 2.0: far leva sul customer-self service e sulla gestione di dati algoritmici per raggiungere l'intero web, le periferie, non solo il centro, la "lunga coda", non solo la testa.

## **Una Piattaforma Batte Sempre un'Applicazione**

In ogni confronto del passato con i propri rivali, Microsoft ha giocato con successo la carta della piattaforma, sconfiggendo anche le applicazioni più importanti. Windows ha consentito a Microsoft di sostituire Lotus 1-2-3 con Excel, WordPerfect con Word, e Netscape Navigator con Internet Explorer.

Questa volta, però, lo scontro non è tra una piattaforma e un'applicazione, ma tra due piattaforme, ognuna delle quali con un modello di business radicalmente diverso. Da una parte, un singolo fornitore di software, la cui base installata in modo massiccio e il sistema operativo e le API strettamente integrati forniscono il controllo sul paradigma di programmazione; dall'altra, un sistema, senza un proprietario, tenuto insieme da una serie di protocolli, standard aperti e accordi di co-operazione.

Windows rappresenta il culmine del controllo di software proprietario tramite l'uso delle API. Netscape ha provato a strappare il controllo a Microsoft utilizzando le stesse tecniche che Microsoft stessa aveva usato contro i suoi rivali, ma ha fallito. Apache, che aveva scelto gli standard aperti del web, ha avuto successo. La battaglia non è più iniqua, una piattaforma contro una singola applicazione, bensì una piattaforma contro un'altra piattaforma; la domanda quindi è quale piattaforma, e più specificatamente, quale architettura e quale modello di business è più idoneo per approfittare delle opportunità future.

Windows era una soluzione brillante a questi problemi all'inizio dell'era dei PC. Ha livellato il campo di gioco per gli sviluppatori di applicazioni, risolvendo un buon numero di problemi che in precedenza aveva reso difficile la vita a questa industria. Ma un singolo approccio monolitico, controllato da un solo venditore, non è più una soluzione, è un problema. I sistemi communication oriented, come è sicuramente la piattaforma internet, richiedono interoperabilità. [A meno che un venditore possa controllare entrambe le estremità di ogni interazione](#), le possibilità di vincolo dell'utente via API software sono limitate.

Qualsiasi fornitore di applicazioni Web 2.0 che cercasse di vincolare i profitti al controllo della piattaforma, per definizione, non giocherebbe più sulla potenza data dalla piattaforma stessa.

Questo non per dire che non ci siano opportunità di creare vincoli e vantaggi competitivi, ma noi crediamo che questi non dovrebbero essere ricercati attraverso il controllo delle API software e dei protocolli. È in corso un nuovo gioco. Le società che avranno successo nell'era del Web 2.0 saranno quelle in grado di comprendere le regole del gioco, piuttosto che cercare di tornare alle regole dell'era del software per PC.

Non sorprende che altre storie di successo del web 2.0 dimostrino questo stesso comportamento. eBay ha abilitato transazioni occasionali anche di soli pochi dollari tra singoli individui, agendo come un intermediario automatizzato. Napster (sebbene chiuso per ragioni legali), ha costruito la sua rete non costruendo un database di canzoni centralizzato, ma architettando un sistema in modo tale che chiunque scarichi un pezzo, diventi esso stesso un server, facendo quindi crescere il network.

## **Akamai vs. BitTorrent**

Come DoubleClick, Akamai è ottimizzato per fare business con la testa e non con la coda, con il centro e non con le periferie. Sebbene sia a vantaggio degli individui delle periferie del web, facilitando il loro accesso ai siti high-demand al centro, raccoglie i propri guadagni da proprio questi siti centrali.

BitTorrent, come altri pionieri nel movimento P2P, assume un approccio radicale verso la decentralizzazione di internet. Ogni client è anche un server; i file sono frazionati in parti che possono essere serviti da postazioni multiple, concorrenti alla creazione di una rete trasparente di downloader che forniscono sia banda, che dati agli altri downloader. Più il file è popolare, infatti, più velocemente può essere disponibile, in quanto ci sarà un numero maggiore di downloader che mettono a disposizione i diversi frammenti del file completo.

BitTorrent dimostra così un principio chiave del Web 2.0: il servizio migliora automaticamente con l'aumentare degli utenti. Mentre Akamai deve aggiungere server per migliorare il servizio, ogni consumatore di BitTorrent porta le proprie risorse al gruppo. C'è un'implicita "architettura della partecipazione", un'etica incorporata di co-operazione, nella quale il servizio funziona principalmente come un broker intelligente, collegando le periferie una con l'altra e sfruttando la potenza degli utenti stessi.

## **Sfruttare l'Intelligenza Collettiva**

Il principio centrale che sta dietro il successo dei giganti nati nell'era del Web 1.0 che sono sopravvissuti per guidare l'era del Web 2.0 sembra essere questo: essi hanno abbracciato la potenza del web per sfruttare l'intelligenza collettiva:

- L'hyperlinking è il fondamento del web. Quando gli utenti aggiungono nuovi concetti e nuovi siti, questi vengono integrati alla struttura del web dagli altri utenti che ne scoprono il contenuto e creano link. Così come le sinapsi si formano nel cervello, con le associazioni che

diventano più forti attraverso la ripetizione o l'intensità, le connessioni nel web crescono organicamente come risultato dell'attività collettiva di tutti gli utenti del web.

- Yahoo!, la prima grande storia di successo in internet, nacque come un catalogo, o una directory di link, un'aggregazione del lavoro migliore di migliaia e poi milioni di utilizzatori del web. Sebbene Yahoo! da allora abbia modificato il proprio business, creando contenuti di vario tipo, il suo ruolo come portale per il lavoro collettivo degli utenti della rete rimane il centro del suo valore.
- Il fattore di successo di Google nel campo delle ricerche, che rapidamente l'ha trasformato nel leader di mercato indiscusso, è stato il PageRank, un metodo che utilizza la struttura dei link anziché semplicemente le caratteristiche di una pagina web, per fornire risultati di ricerca migliori.
- Il prodotto di eBay è l'attività collettiva di tutti i suoi utenti; come il web stesso, eBay cresce organicamente in risposta all'attività degli utenti. Il ruolo della società è quello di mettere a disposizione un contesto in cui l'attività degli utenti possa aver luogo. Inoltre, il vantaggio competitivo di eBay viene quasi interamente dalla massa critica di acquirenti e venditori che rendono qualsiasi nuovo attore, con un servizio simile, significativamente meno interessante.
- Amazon vende gli stessi prodotti che vendono i suoi concorrenti, quali Barnesandnoble.com, e riceve le stesse descrizioni del prodotto, le stesse immagini di copertina e gli stessi contenuti editoriali dai suoi fornitori. Ma Amazon ha fatto della partecipazione degli utenti una scienza. Conta su un numero sempre maggiore di recensioni da parte degli utenti, invita a partecipare in vari modi su virtualmente qualsiasi pagina—e ancora più importante, usa l'attività degli utenti per produrre risultati di ricerca migliori. Mentre una ricerca sul sito Barnesandnoble.com molto probabilmente porterà ai prodotti della società o ai risultati sponsorizzati, Amazon porta sempre al "più popolare", un calcolo in tempo reale basato non solo sulle vendite, ma anche su altri fattori che gli *insider* di Amazon chiamano il "flusso" intorno ai prodotti. Contando su una partecipazione sempre maggiore, non sorprende che le vendite di Amazon superino quelle dei concorrenti.

Ora, le società innovative che seguono queste intuizioni e che forse le estenderanno ulteriormente, stanno lasciando il segno nel web:

- Wikipedia, un'enciclopedia online basata sull'inverosimile idea che ciascuna voce possa essere aggiunta da qualsiasi utente web o editata da qualcun altro, è un esperimento radicale di fiducia, che applica alla creazione di contenuti il detto di Eric Raymond (coniato originariamente nel contesto del [software open source](#)) che sostiene che "con molti occhi puntati addosso, ogni bug diventa una bazzecola". Wikipedia è già nei 100 top siti web, e molti ritengono che sarà tra i primi dieci a breve. Questo rappresenta un cambiamento profondo nelle dinamiche della creazione di contenuti!
- Siti come del.icio.us e [Flickr](#), due società che hanno ottenuto grande attenzione negli ultimi tempi, hanno fatto da pionieri per un concetto che alcuni definiscono "[folksonomia](#)" (in contrasto con la tassonomia), uno stile di categorizzazione collaborativa dei siti che utilizza parole chiave liberamente scelte, che spesso sono definite tag. Il tagging consente di ottenere quel tipo di associazione multipla e in sovrapposizione che utilizza il cervello stesso, anziché delle categorie rigide. Nell'esempio canonico, una foto Flickr di un cucciolo può essere "taggata" sia come "cucciolo", sia come "carino"—consentendo il suo reperimento lungo gli assi naturali generati dall'attività degli utenti.
- I prodotti per filtrare lo spam in modo collaborativo come Cloudmark, aggregano le decisioni individuali degli utenti che leggono le proprie e-mail circa cosa è e cosa non è spam, superando i sistemi che si basano sull'analisi dei messaggi stessi.
- È una verità lapalissiana che le società che vantano i più grandi successi in internet non pubblicizzano i propri prodotti. La loro adozione è guidata dal "marketing virale"—cioè dalle raccomandazioni che passano direttamente da un utente a un altro. Potete quasi arrivare alla conclusione che se un sito o un prodotto si basa sulla pubblicità per farsi conoscere, non è Web 2.0.
- Anche buona parte dell'infrastruttura del web—inclusi il codice di Linux, Apache, MySQL, e

Perl, PHP, o Python usati in molti server web—si affidano ai metodi di [peer-production](#) dell'open source, in essi stessi un esempio di intelligenza collettiva, creata dalla rete. Ci sono più di 100.000 progetti di software open source elencati su [SourceForge.net](#). Chiunque può aggiungere un progetto, chiunque può scaricare e utilizzare il codice. I nuovi progetti migrano dalle periferie al centro come risultato del fatto che gli utenti li utilizzano, un processo di adozione organico del software che si affida interamente al marketing virale.

La lezione: *Gli effetti del network derivanti dai contributi degli utenti sono la chiave del predominio del mercato nell'era del Web 2.0*

## **Il Blog e la Saggezza delle Folle**

Una delle caratteristiche più pubblicizzate dell'era del Web 2.0 è la crescita dei blog. Home page personali sono state pubblicate sin dall'inizio del web e i diari personali e le colonne di opinioni quotidiane lo sono da molto prima, quindi a cosa si deve tutto questo scalpore?

Nella sua essenza, un blog non è altro che una home page personale nel formato di un diario. Ma come fa [notare](#) Rich Skrenta, l'organizzazione cronologica di un blog "sembra una differenza insignificante, ma definisce un sistema di erogazione, inserimento pubblicitario e una catena del valore completamente differente".

Una delle cose che ha fatto la differenza è una tecnologia che si chiama [RSS](#). RSS è il progresso più significativo nell'architettura fondamentale del web da quando i primi hacker hanno capito che le CGI potevano essere utilizzate per creare website basati su un database. RSS consente di collegarsi non solo a una pagina, ma di "abbonarsi" ad essa, ricevendo un avviso ogni volta che la pagina viene modificata. Skrenta lo definisce il "web incrementale". Altri lo chiamano "live web".

Ora, naturalmente, i "website dinamici" (cioè siti basati su un database con contenuti generati dinamicamente) hanno sostituito le pagine web statiche più di dieci anni fa. Quello che è dinamico nel *live web* non sono solo le pagine, ma anche i link. Un link a un weblog punta a una pagina continuamente modificata, con un "permalink" per ogni singolo inserimento e una notifica per ogni cambiamento. Un feed RSS è quindi più potente di un link, sia esso un bookmark o di un link ad una singola pagina.

RSS significa anche che il browser web non è l'unico modo per vedere una pagina web. Mentre alcuni aggregatori RSS, come Bloglines, si basano sul web, altri sono applicazioni desktop per i client e altri ancora consentono l'abbonamento ai contenuti costantemente aggiornati agli utilizzatori di dispositivi portatili.

RSS viene utilizzato per spingere non solo i nuovi inserimenti sul blog, ma anche tutti i tipi di aggiornamento dei dati, incluse le quotazioni azionarie, il meteo e la disponibilità di foto. Questo utilizzo è anche un ritorno a una delle sue radici: RSS nacque nel 1997 dalla confluenza della tecnologia "Really Simple Syndication" di Dave Winer, utilizzata per pubblicare gli aggiornamenti del blog, e "Rich Site Summary" di Netscape, che consentiva agli utenti di creare home page personali di Netscape con un flusso di dati aggiornati regolarmente. Netscape ha perso interesse e la tecnologia è stata sviluppata dalla Userland, la società di proprietà di Winer, pioniera nel settore del blogging. Nell'insieme delle attuali applicazioni, comunque, vediamo l'eredità di entrambi i genitori.

## **L'Architettura partecipativa**

Alcuni sistemi sono progettati per incoraggiare la partecipazione. Nei suoi scritti, [The Cornucopia of the Commons](#), Dan Bricklin indica che ci sono tre modi per realizzare un grande database. Il primo dimostrato da Yahoo!, è di pagare le persone perché lo facciano. Il secondo, ispirato dalle lezioni che arrivano dalla comunità degli open source, è di cercare volontari che realizzino lo stesso compito.

L'[Open Directory Project](#), un concorrente open source di Yahoo, ne è il risultato. Ma [Napster](#) ha dimostrato un terzo modo. Perché Napster ha impostato di default la possibilità di far scaricare automaticamente qualsiasi pezzo musicale, consentendo ad ogni utente di contribuire all'aumento del valore del database condiviso. Questo stesso approccio è stato seguito da tutti gli altri servizi di condivisione di file P2P.

Una delle lezioni chiave dell'era di Web 2.0 è questa: *gli utilizzatori aggiungono valore*. Ma solo una piccola percentuale di utenti si prenderà la briga di aggiungere valore alla vostra applicazione tramite metodi espliciti. Perciò, le società Web 2.0 impostano di default sistemi *per l'aggregazione dei dati degli utenti e per la costruzione di valore come effetto laterale dell'utilizzo ordinario dell'applicazione*. Come indicato in precedenza, questi realizzano sistemi che migliorano con l'aumentare del numero di utenti.

Mitch Kapor una volta affermò che “architettura è politica”. La partecipazione è intrinseca in Napster, è parte della sua architettura fondamentale.

Questa visione dell'architettura può essere considerata alla base del successo dei software open source anche più dei frequentemente citati appelli al volontariato. L'architettura di internet, e il World Wide Web, così come i progetti di software open source come Linux, Apache, e Perl, è tale che gli utenti che perseguono i propri interessi “egoistici” costruiscono un valore collettivo come un sottoprodotto automatico. Ognuno di questi progetti ha una piccola anima, meccanismi di estensione ben definiti e un approccio che consente a chiunque di aggiungere qualsiasi componente ben funzionante, facendo crescere gli strati più esterni di quello che Larry Wall, il creatore di Perl, definisce “la cipolla”. In altre parole, queste tecnologie dimostrano gli effetti della rete, semplicemente attraverso il modo in cui sono state progettate.

Questi progetti possono essere visti come aventi una naturale architettura partecipativa. Ma, come Amazon dimostra, con sforzi continui (oltre a incentivi economici quale il programma Associates), è possibile sovrapporre questa architettura a un sistema che normalmente non la possiede.

Ma l'utilizzo di RSS è solo parte di quello che rende differente un weblog da una pagina web comune. Tom Coates sottolinea [il significato del permalink](#):

Può sembrare un'insignificante funzionalità ora, ma si tratta effettivamente del dispositivo che ha trasformato i weblog da un fenomeno di “pubblicazione facile” in una confusione conversazionale di comunità che si sovrappongono. Per la prima volta è diventato relativamente semplice puntare direttamente a una sezione specifica del sito di qualcun altro e parlarne. Sono nate discussioni. Sono nate conversazioni. E –come risultato– sono nate amicizie. Il permalink è stato il primo tentativo – e quello più di successo—di costruire ponti tra i weblog.

In molti modi, la combinazione di RSS e permalink aggiunge molte delle caratteristiche di NNTP, il Network News Protocol of the Usenet, all'HTTP, il protocollo web. La “blogosfera” può essere considerata un nuovo, peer-to-peer equivalente di Usenet e dei bulletin-board, le aree conversazionali del primo internet. La gente può non solo sottoscrivere i siti degli altri, e collegarsi facilmente ai singoli commenti su una pagina, ma anche, tramite un meccanismo noto come *trackback*, può vedere quando qualcun altro si collega alle proprie pagine e può rispondere, sia con link reciproci, o aggiungendo commenti.

Stranamente, i collegamenti reciproci erano lo scopo dei primi sistemi ipertestuali, come Xanadu. I puristi dell'ipertesto hanno celebrato i trackback come un passo avanti verso i collegamenti a due vie. Ma bisogna notare che i trackback non sono propriamente sistemi a due vie--piuttosto, sono collegamenti a una via realmente (potenzialmente) simmetrici, che creano solo l'effetto di reciprocità. La differenza può sembrare sottile, ma in pratica è enorme. I sistemi di network sociale come Friendster, Orkut, e LinkedIn, che richiedono il riconoscimento da parte del ricevente al fine di

stabilire una connessione, mancano della stessa scalabilità del web. Come ha fatto notare Caterina Fake, co-fondatrice del servizio di condivisione foto di Flickr, l'attenzione è reciproca solo per coincidenza. (Flickr consente così agli utilizzatori di impostare delle watch list—qualunque utilizzatore può iscriversi al photostream di un altro utilizzatore via RSS. Chi è oggetto delle attenzioni altrui riceve una notifica, ma non deve approvare la connessione).

Se una parte essenziale del Web 2.0 riguarda lo sfruttamento dell'intelligenza collettiva, trasformando il web in una specie di cervello globale, la blogosfera è l'equivalente di un chiacchiericcio mentale costante nel proencefalo, la voce che tutti sentiamo nella nostra testa. Può non riflettere la struttura profonda del cervello, che spesso è inconscia, ma è l'equivalente dei pensieri consci. E come una riflessione del pensiero conscio e dell'attenzione, la blogosfera ha iniziato ad avere un potente effetto.

Innanzitutto, perchè i motori di ricerca utilizzano le strutture di link per restituire le pagine presumibilmente più utili: i blogger, essendo i linker più prolifici e più tempestivi, hanno un ruolo sproporzionato nel dare forma ai risultati dei motori di ricerca. Poi, perchè la comunità dei blog è profondamente auto-referenziale: i blogger che prestano attenzione agli altri blogger aumentano la loro visibilità e il loro potere. Anche la "camera dell'eco" che i critici denigrano è un amplificatore.

Se si trattasse semplicemente di un amplificatore, i blog sarebbero poco interessanti. Ma come Wikipedia, il blog sfrutta l'intelligenza collettiva come una specie di filtro. Quello che James Suriowecki chiama "[la saggezza delle folle](#)" entra in gioco e così come PageRank produce risultati migliori dell'analisi di ogni documento individuale, l'attenzione collettiva della blogosfera seleziona il valore.

Sebbene i media tradizionali possano vedere i singoli blog come concorrenti, la cosa più terrificante è che la concorrenza è in realtà con la blogosfera come insieme. Questa non è semplicemente una concorrenza tra siti, ma una concorrenza tra modelli di business. Il mondo Web 2.0 è anche il mondo di ciò che Dan Gillmor chiama "[noi, i media](#)", un mondo in cui chi in precedenza era definito "il pubblico", e non poche persone dietro le quinte, decide cosa sia importante.

## **I Dati sono il Prossimo "Intel Inside"**

Ogni significativa applicazione internet sinora è stata supportata da un database specializzato: il web crawl di Google, la directory di Yahoo!(e web crawl), il database prodotti di Amazon, il database di prodotti e venditori di eBay, il database di cartine di MapQuest, il database delle canzoni distribuite di Napster. Come ha evidenziato Hal Varian in una conversazione personale l'anno scorso, "SQL è il nuovo HTML." La gestione dei database è una competenza centrale delle società Web 2.0, al punto che spesso abbiamo definito queste applicazioni come "[infoware](#)" piuttosto che semplicemente software.

Questo fatto porta a una domanda chiave: a chi appartengono i dati?

Nell'era internet, abbiamo già avuto modo di vedere un certo numero di casi in cui il controllo sui database ha portato al controllo del mercato e a guadagni finanziari fuori misura. Il monopolio sul registro dei domini originariamente garantito per decreto dal governo alla Network Solutions (in seguito acquistata da Verisign) fu uno dei primi esempi di grandi guadagni di internet. Mentre abbiamo discusso del fatto che il vantaggio competitivo tramite il controllo delle API è molto più difficile nell'era di internet, il controllo delle fonti dei dati chiave non lo è, specialmente se la creazione di queste fonti di dati è particolarmente dispendiosa o suscettibile di aumentare i guadagni tramite gli effetti della rete.

Guardate le note sul copyright alla base di ogni mappa fornita da MapQuest, maps.yahoo.com, maps.msn.com, o maps.google.com, e vedrete la riga "Maps copyright NavTeq, TeleAtlas," o con i

nuovi servizi di immagini via satellite, "Images copyright Digital Globe." Queste società hanno fatto investimenti importanti nei loro database (NavTeq da sola ha riportato di aver investito \$750 milioni per costruire il proprio database di indirizzi di strade e di direzioni. La Digital Globe ha speso \$500 milioni per lanciare il proprio satellite per migliorare le immagini fornite dal governo). NavTeq è andata così avanti da imitare il familiare logo Intel Inside della Intel: le auto con i sistemi di navigazione portano il marchio "NavTeq Onboard." I dati corrispondono in effetti all'Intel Inside di queste applicazioni, un unico componente nei sistemi la cui infrastruttura software è ampiamente open source o altrimenti resa una commodity.

L'arena delle mappe online, ora vivamente contestata, dimostra come la scarsa comprensione dell'importanza di possedere un nucleo di dati alla base dell'applicazione potrebbe ridurre la propria capacità competitiva. MapQuest ha fatto da pioniere nella categoria delle mappe per il web nel 1995, ma quando Yahoo!, quindi Microsoft, e più recentemente Google, decisero di entrare nel mercato, questi furono in grado di offrire un'applicazione competitiva semplicemente usando in licenza gli stessi dati.

Differente è, invece, la posizione di Amazon.com. Come i concorrenti quali Barnesandnoble.com, il suo database originale deriva dal provider del registro ISBN R.R. Bowker. Ma a differenza di MapQuest, Amazon ha sistematicamente arricchito i dati in suo possesso, aggiungendo quelli forniti dagli editori, come le immagini delle copertine, le tavole dei contenuti, gli indici e materiali campione. Ancora più importante, essi hanno sfruttato i loro utenti per inserire i dati, così che, dopo dieci anni, Amazon, e non Bowker, è la fonte primaria dei dati bibliografici sui libri, una fonte di riferimento per studiosi e bibliotecari, oltre che per i consumatori. Amazon ha anche introdotto il proprio codice identificativo proprietario, l'[ASIN](#),

che quando già esiste corrisponde all'ISBN, mentre viene creato appositamente per quei prodotti che ne sono sprovvisti. Effettivamente, Amazon ha "inglobato e esteso" l'opera iniziata dai propri fornitori di dati.

Immaginate se MapQuest avesse fatto la stessa cosa, sfruttando i propri utenti per inserire mappe e direzioni, aggiungendo valore. Sarebbe stato molto più difficile per i concorrenti entrare nel mercato semplicemente usando in licenza la base di dati.

La recente introduzione di Google Maps fornisce un laboratorio vivente per la concorrenza tra i fornitori di applicazioni e i loro fornitori di dati. Il leggero modello di programmazione di Google ha portato alla creazione di numerosi servizi a valore aggiunto in forma di mashups che collegano Google Maps con altre fonti di dati accessibili via internet. L'esempio più eccellente di tali mashup è [housingmaps.com](#) di Paul Rademacher, che combina le funzionalità di Google Maps con i dati sugli acquisti e le locazioni di appartamenti forniti da [Craigslist](#), creando uno strumento interattivo per la ricerca di case.

Attualmente, questi mashups sono principalmente esperimenti innovativi, realizzati dagli hacker. Ma l'attività imprenditoriale segue da vicino. E già ora possiamo vedere che almeno per una certa classe di sviluppatori, Google ha assunto il ruolo di fonte di dati indipendentemente da Navteq e si è inserita come intermediario favorito. Ci aspettiamo di assistere, nei prossimi anni, a battaglie tra i fornitori di dati e di applicazioni, in quanto entrambi sanno quanto certe classi di dati saranno importanti per la realizzazione delle applicazioni Web 2.0.

*La corsa è per la proprietà di certe classi di dati centrali: indirizzi, identità, date di eventi pubblici, codici di identificazione di prodotti e namespace. In molti casi, quando il costo per la creazione dei dati è alto, ci può essere un'opportunità per un'azione in stile Intel Inside, con un'unica fonte di dati. In altri casi, il vincitore sarà la società che per prima raggiungerà una massa critica tramite l'aggregazione degli utenti e trasformerà questi dati aggregati in un servizio di sistema.*

Per esempio, nell'area dell'identità, PayPal, 1-click di Amazon, e i milioni di utenti dei sistemi di

comunicazione, possono essere tutti contendenti legittimi per la costruzione di un database di identità comprensivo di tutta la rete. (A questo proposito, il recente tentativo di Google di usare i numeri dei telefoni cellulari come un identificativo per i clienti Gmail può essere un passo per inglobare ed estendere ciò che è stato fatto dal sistema telefonico). Nel frattempo, i nuovi entrati come [Sxip](#) stanno esplorando il potenziale delle identità confederate, in ricerche tipo “1-click distribuito” che forniranno un ininterrotto sotto-sistema di identità Web 2.0. Nell’area della pianificazione, [EVDB](#) è un tentativo di costruire la più grande agenda virtuale condivisa al mondo tramite un’architettura partecipativa in stile wiki. Mentre la giuria è ancora impegnata a valutare il successo di ogni particolare avvio o approccio, è chiaro che gli standard e le soluzioni in queste aree, che hanno effettivamente trasformato certe classi di dati in sotto-sistemi affidabili del “sistema operativo di internet”, consentirà la realizzazione della prossima generazione di applicazioni.

Un ulteriore punto da notare riguarda i dati e cioè che gli utenti sono preoccupati per la privacy e per i loro diritti sui loro dati. In molte delle prime applicazioni web, il copyright veniva fatto rispettare in modo un po’ vago. Per esempio, Amazon rivendica i diritti su qualsiasi recensione inserita nel sito, ma in assenza di un’imposizione, le persone possono inviare la stessa recensione altrove. Comunque, poiché le società stanno iniziando a comprendere che il controllo sui dati potrebbe essere la loro fonte principale per ottenere un vantaggio competitivo, potremo notare un aumento negli sforzi per ottenerne il controllo.

Così come il nascere del software proprietario aveva portato al movimento [Free Software](#), ci aspettiamo che il nascere di database proprietari darà come risultato la nascita di un movimento Free Data entro la prossima decade. Possiamo già vedere i segni di questa tendenza nei progetti open data quali Wikipedia, Creative Commons, e nei progetti software come [Greasemonkey](#), che consentono agli utenti di controllare come i dati vengono visualizzati sui loro computer.

## Fine del Ciclo delle Release di Software

Come indicato in precedenza nella discussione di Google vs. Netscape, una delle caratteristiche che definiscono il software dell’era internet è che questo viene fornito come servizio, non come prodotto. Questo fatto porta a una serie di cambiamenti fondamentali nei modelli di business di queste società:

1. *Le operazioni devono diventare una competenza centrale.* La conoscenza di Google o di Yahoo! nello sviluppo prodotti deve misurarsi con la conoscenza nelle operazioni quotidiane. Il passaggio dal software come prodotto al software come servizio è così fondamentale che *il software cesserà di funzionare se non mantenuto quotidianamente*. Google deve tenere sotto controllo continuamente il web e aggiornare i suoi indici, deve filtrare continuamente i collegamenti spam e altri tentativi di influenzare i suoi risultati, rispondere continuamente e dinamicamente alle centinaia di milioni di ricerche asincrone degli utenti, confrontandole simultaneamente con pubblicità contestuali.

Non è un caso che l’amministrazione del sistema, il networking e le tecniche di bilanciamento del carico di Google sono segreti protetti forse più accuratamente dei loro algoritmi di ricerca. Il successo di Google nell’automatizzazione di questi processi è un fattore chiave per il suo vantaggio competitivo sui costi rispetto ai concorrenti.

Non è un caso anche che i [linguaggi di scrittura quali Perl, Python, PHP e ora Ruby, abbiano un ruolo così ampio](#) per le società web 2.0. Perl fu splendidamente descritto da Hassan Schroeder, il primo webmaster di Sun, come “il duct tape di internet”. I linguaggi dinamici (spesso definiti linguaggi di scrittura e guardati dall’alto in basso dagli ingegneri di software dell’era dei manufatti software) sono lo strumento di scelta per gli amministratori di sistemi e di reti, oltre che per gli sviluppatori di applicazioni che costruiscono sistemi dinamici che richiedono cambiamenti costanti.

2. *Gli utenti devono essere trattati come co-sviluppatori*, quale conseguenza delle pratiche di sviluppo degli open source (anche se il software in questione difficilmente sarà rilasciato con una licenza open source). Il motto open source, "rilascia presto e rilascia spesso" si è infatti trasformato in qualcosa di ancora più radicale, "il beta perpetuo", in cui il prodotto è sviluppato in un contesto open, con nuove caratteristiche integrate e aggiornate su base mensile, settimanale o anche quotidiana. Non è un caso che servizi quali Gmail, Google Maps, Flickr, del.icio.us, e simili, potranno essere contrassegnati da un logo "Beta" per anni.

Il monitoraggio in tempo reale del comportamento degli utenti per vedere quali nuove funzioni siano utilizzate e come vengano utilizzate diventa un'altra competenza centrale richiesta. Uno sviluppatore per un servizio online tra i più importanti ha affermato: "Noi mettiamo in linea due o tre nuove funzioni in alcune parti del sito ogni giorno e se gli utilizzatori non le adottano, le togliamo. Se invece le apprezzano, le estendiamo all'intero sito".

Cal Henderson, il capo sviluppatore di Flickr, ha recentemente [affermato che loro pubblicano nuovi contenuti ogni mezz'ora](#). Questo è chiaramente un modello di sviluppo radicalmente diverso! Sebbene non tutte le applicazioni web siano sviluppate in modo estremo come Flickr, quasi tutte le applicazioni web hanno un ciclo di sviluppo che è radicalmente diverso da quelli dell'era del PC o dei client-server. È per questa ragione che un recente editoriale di ZDnet è arrivato alla [conclusione che Microsoft non sarà in grado di battere Google](#): "Il modello di business di Microsoft dipende dal fatto che chiunque aggiorni il proprio ambiente di computer ogni due o tre anni. Quello di Google dal fatto che ogni utente esplori cosa c'è di nuovo nel loro ambiente computer tutti i giorni".

Sebbene Microsoft abbia dimostrato un'enorme abilità nell'imparare dalla propria concorrenza, spesso migliorando, non c'è dubbio che ora la concorrenza richiederà a Microsoft (e, estendendo il concetto, a qualsiasi altra società di software esistente) di diventare una società profondamente diversa. Le società native Web 2.0 godono di un vantaggio naturale in quanto non hanno vecchi schemi (e modelli di business o fonti di reddito corrispondenti) di cui liberarsi.

## Una Tesi di Investimento Web 2.0

Il venture capitalist Paul Kedrosky scrive: "La chiave è trovare gli investimenti attuabili nei quali si è in disaccordo con l'opinione comune". È interessante vedere come ogni aspetto il Web 2.0 preveda il disaccordo con l'opinione comune: ognuno enfatizzava che i dati dovessero rimanere privati, Flickr/Napster/et al. li hanno resi pubblici. Non si tratta di un disaccordo per creare fastidi (cibo per animali! online!), ma si tratta di un disaccordo con il quale è possibile costruire qualcosa dalle differenze. Flickr costruisce comunità, Napster ha costruito collezioni.

Un altro modo per guardare a questo fatto è che le società di successo rinunciano a qualcosa di costoso, ma considerato critico, per ottenere gratuitamente qualcosa di valore che, una volta, era costoso. Per esempio, Wikipedia rinuncia al controllo editoriale centralizzato in cambio di velocità e ampiezza. Napster ha rinunciato all'idea del "catalogo" (tutte le canzoni che il venditore promuoveva) e ha ottenuto ampiezza. Amazon ha rinunciato all'idea di avere un presidio fisico per la vendita e ha ottenuto di servire tutto il mondo. Google ha rinunciato ai grandi clienti (inizialmente) e ha ottenuto l'80% dei piccoli, i cui bisogni non erano soddisfatti. C'è qualcosa molto aikido (utilizzare la forza dei vostri oppositori contro di essi) nell'affermazione "sai, hai ragione—assolutamente nessuno al mondo PUÒ aggiornare questo articolo. E indovina, questa è una brutta notizia per te".

--Nat Torkington

## Modelli di Programmazione Leggeri

Una volta che l'idea dei servizi web è diventata familiare, le grandi società sono entrate nella mischia con un complesso insieme di web services progettati per creare ambienti di programmazione altamente affidabili per applicazioni distribuite.

Ma buona parte del web ha avuto successo proprio perchè ha eliminato molta della teoria sugli ipertesti. La sostituzione di un semplice pragmatismo per la progettazione ideale, ha permesso all'RSS di diventare forse il servizio web più ampiamente diffuso grazie alla sua semplicità, mentre i complessi web services delle grandi aziende devono ancora ottenere un'ampia diffusione.

In modo simile, i servizi web di Amazon.com sono forniti in due forme: il primo aderisce ai formalismi dei web services di SOAP (Simple Object Access Protocol), il secondo fornisce semplicemente dati XML nell'HTTP, con un approccio leggero, spesso definito REST (Representational State Transfer). Mentre le connessioni B2B ad alto valore (come quelle tra Amazon e i partner rivenditori, come ToysRUs) utilizzano lo stack SOAP, mentre, secondo Amazon, il 95% dell'utilizzo avviene con il servizio leggero REST.

Questa stessa ricerca di semplicità può essere vista in altri web services "organici". Il recente rilascio di Google Maps da parte di Google è un esempio di questo. La semplice interfaccia AJAX di Google Maps (Javascript e XML) è stata velocemente decriptata dagli hacker, che hanno poi proceduto a miscelare i dati in un nuovo servizio.

I web services relativi alle mappe sono stati disponibili per un certo periodo di tempo presso i fornitori GIS, come ESRI, oltre che da MapQuest e Microsoft MapPoint. Ma Google Maps ha scom bussolato le carte a causa della sua semplicità. Mentre la sperimentazione con qualsiasi dei web services supportati da fornitori richiedeva un contratto formale tra le parti, il modo in cui Google Maps è stato implementato ha lasciato i dati a disposizione di chi volesse prenderli e gli hacker hanno trovato presto il modo di riutilizzare in modo creativo questi dati.

Da ciò si possono trarre alcune lezioni significative:

1. *Supportare modelli di programmazione leggeri che consentano di abbinare sistemi liberamente.* La complessità dei web services sponsorizzati dalle grandi aziende è progettata per consentire abbinamenti rigidi. Mentre questo è necessario in molti casi, molte delle più interessanti applicazioni possono rimanere abbinare liberamente, pur rimanendo fragili. Il punto di vista del Web 2.0 è molto diverso dal punto di vista IT tradizionale!
2. *Pensare alla syndacation e non alla co-ordinazione.* I web services semplici, come RSS e quelli basati su REST, sono propensi a rendere disponibili i dati verso l'esterno piuttosto che a controllare cosa succede quando si arriva all'altra estremità della connessione. Questa idea è fondamentale per internet stesso, il riflesso di quanto è noto come il [principio end-to-end](#).
3. *Progettare "hackable" e remixable.* I sistemi come il web, RSS e AJAX hanno tutti questo in comune: le barriere per il riutilizzo sono estremamente basse. Buona parte del software utile è effettivamente open source, ma anche quando non lo è, c'è poco in termini di protezione della proprietà intellettuale. L'opzione "View Source" del browser web ha reso qualunque utente in grado di copiare le pagine web di chiunque altro; l'RSS è stato progettato per abilitare l'utente a vedere il contenuto desiderato, nel momento desiderato, non su ordine del provider di informazioni; i web services di maggior successo sono quelli che sono stati facilmente portati verso direzioni nuove e diverse, non immaginate dai loro creatori. La frase "some rights reserved," resa popolare da Creative Commons in contrapposizione alla più tipica "all rights reserved," è un utile indicatore.

## **Innovazione nell'Assemblaggio**

I modelli di business leggeri sono una naturale conseguenza della programmazione leggera e delle connessioni leggere. Dal punto di vista del Web 2.0 il riutilizzo è una cosa positiva. Un nuovo

servizio come housingmaps.com è stato costruito semplicemente unendo insieme due servizi esistenti. Housingmaps.com non ha un modello di business (non ancora)—ma per molti servizi su piccola scala, Google AdSense (o forse il programma di affiliazione di Amazon, o entrambi) fornisce l'equivalente di un business model.

Questi esempi danno una panoramica di un altro principio chiave del web 2.0, che chiamiamo "innovazione nell'assemblaggio." Quando i componenti di base abbondano, si può creare valore aggiunto semplicemente assemblandoli in un modo nuovo o efficace. Così come la rivoluzione del PC ha fornito molte opportunità di innovazione assemblando hardware di base, con società come Dell che hanno fatto di questo assemblaggio una scienza, sconfiggendo quindi società il cui modello di business richiedeva innovazione nello sviluppo di prodotto, noi crediamo che il Web 2.0 fornirà alle società molte opportunità di battere la concorrenza sfruttando in modo migliore e integrando i servizi forniti dagli altri.

## Il Software supera il Livello del Singolo Dispositivo

Un'altra caratteristica del Web 2.0 che merita di essere menzionata è il fatto che non è più limitato alla piattaforma PC. Nel suo discorso di congedo da Microsoft, Dave Stutz, a lungo sviluppatore di Microsoft, ha evidenziato che [“un software utile scritto senza vincoli dettati dal singolo dispositivo, otterrà alti margini per un lungo periodo.”](#)

Naturalmente, qualsiasi applicazione web può essere vista come un software svincolato dal singolo dispositivo. Dopotutto, anche la più semplice applicazione web prevede almeno due computer: quello ospitante il server web e quello ospitante il browser. E come abbiamo discusso, lo sviluppo del web come piattaforma estende questa idea alle applicazioni sintetiche composte da servizi forniti da computer multipli.

Ma come con molte aree del Web 2.0, dove il "2.0" non rappresenta qualcosa di nuovo ma piuttosto la più completa realizzazione del vero potenziale della piattaforma web, questa frase ci fornisce l'indicazione chiave su come progettare applicazioni e servizi per la nuova piattaforma.

A oggi, iTunes è il miglior esempio di questo principio. Questa applicazione collega in modo trasparente il dispositivo portatile a un massiccio back-end (apple store), con il PC che funziona da cache locale e da stazione di controllo. In precedenza, ci sono stati molti tentativi di portare i contenuti del web su dispositivi portatili, ma la combinazione iPod/iTunes è una delle prime applicazioni di questo tipo progettate a partire da zero per dispositivi multipli. TiVo è un altro buon esempio.

iTunes e TiVo dimostrano anche molti degli altri principi di base del Web 2.0. Queste non sono applicazioni web a sè stanti, ma aumentano la potenza della piattaforma web, rendendola una parte senza soluzione di continuità, quasi invisibile della loro infrastruttura. La gestione dei dati è chiaramente il cuore della loro offerta. Essi sono servizi, non pacchetti di applicazioni (sebbene iTunes possa essere utilizzato come un'applicazione, gestendo solo i dati dell'utilizzatore locale). Inoltre, sia TiVo che iTunes mostrano, a livello nascente, un utilizzo dell'intelligenza collettiva, sebbene in entrambi i casi, i loro esperimenti siano in guerra con la lobby degli IP. C'è solo una limitata architettura partecipativa in iTunes, sebbene la recente aggiunta del [podcasting](#) modifichi sostanzialmente questa equazione.

Questa è una delle aree del Web 2.0 dove ci aspettiamo di vedere alcuni dei cambiamenti più grandi, con l'aumento dei dispositivi collegati alla nuova piattaforma. Quali applicazioni diventano possibili quando i nostri telefoni e le nostre auto non consumeranno più dati ma li renderanno disponibili? Il monitoraggio del traffico in tempo reale, i flash mobs e il citizen journalism sono solo alcuni dei primi segnali di avviso delle possibilità della nuova piattaforma.

# Web 2.0 Design Patterns

Nel suo libro, [A Pattern Language](#), Christopher Alexander definisce un format per la descrizione concisa della soluzione di problemi architeturali. Egli scrive: "Ogni pattern descrive un problema che avviene di continuo nel nostro ambiente, e quindi descrive il nucleo fondamentale della soluzione a quel problema, in modo tale che potrete usare questa soluzione un milione di volte senza farlo mai allo stesso modo".

## 1. The long tail

I piccoli siti rappresentano il grosso del contenuto internet; nicchie ristrette rappresentano il grosso delle applicazioni internet possibili. *Perciò*: fate leva sul customer-self service e sulla gestione dei dati algoritmici per raggiungere l'intero web, le periferie e non solo il centro, la lunga coda e non solo la testa.

## 2. I Dati sono il Prossimo Intel Inside

Le applicazioni sono sempre più guidate dai dati. *Perciò*: Per un vantaggio competitivo, cercate di possedere una fonte di dati unica e difficile da ricreare.

## 3. Gli Utenti Aggiungono Valore

La chiave per un vantaggio competitivo nelle applicazioni internet è il grado in cui gli utenti siano in grado di aggiungere i propri dati a quelli che voi fornite. *Perciò*: Non limitate la vostra "architettura partecipativa" allo sviluppo del software. Coinvolgete i vostri utenti sia implicitamente, sia esplicitamente nell'aggiungere valore alla vostra applicazione.

## 4. Gli Effetti del Network di Default

Solo una piccola percentuale di utenti si prenderà la briga di aggiungere valore alla vostra applicazione. *Perciò*: Impostate di default un sistema per aggregare i dati degli utenti come effetto laterale dell'utilizzo della vostra applicazione.

**5. Some Rights Reserved.** La protezione della proprietà intellettuale limita il riutilizzo e previene la sperimentazione. *Perciò*: Quando i benefici vengono dall'adozione collettiva e non dalla restrizione privata, assicuratevi che le barriere all'adozione siano basse. Seguite gli standard esistenti e utilizzate le licenze con il minimo di restrizioni possibili. Progettate per l'"hackability" e la "remixability."

## 6. Il Beta Perpetuo

Quando i dispositivi e i programmi sono collegati a internet, le applicazioni non sono più manufatti software, ma servizi in via di sviluppo. *Perciò*: Non inserite più nuove funzioni in release monolitiche, ma, al contrario, aggiungetele regolarmente come parte della normale esperienza dell'utilizzatore. Impegnate i vostri utenti come collaudatori in tempo reale e dotate il servizio di strumenti di controllo così saprete come la gente usa le nuove funzioni.

## 7. Co-operazione, Non Controllo

Le applicazioni Web 2.0 consistono in una rete di data services che collaborano. *Perciò*: Offrite interfacce per i web services e syndication dei contenuti e riutilizzate i data services degli altri. Supportate modelli di programmazione leggeri che consentono sistemi di abbinamento liberi.

## 8. Il Software a Livello Superiore del Singolo Dispositivo

Il PC non è più l'unico dispositivo che consente l'accesso alle applicazioni internet e le applicazioni che sono limitate a un solo dispositivo hanno un valore inferiore rispetto a quelle che sono connesse. *Perciò*: Progettate dall'inizio la vostra applicazione per integrare servizi su dispositivi portatili, PC e server internet.

# La "Rich User Experience"

Fin dai tempi di [Viola browser](#) di Pei Wei, nel 1992, il web veniva usato per trasferire "applets" e altri tipi di contenuti attivi all'interno del browser web. L'introduzione del Java nel 1995 si basava sul trasferimento di detti applets. Il JavaScript e quindi il DHTML furono introdotti come strumenti leggeri per fornire una programmabilità da parte del cliente e esperienze più significative per gli utenti. Alcuni anni fa, Macromedia coniò il termine "Rich Internet Applications" (che è stato adottato anche dal concorrente open source di Flash, Laszlo Systems) per evidenziare la capacità di Flash di fornire non solo contenuti multimediali, ma anche esperienze sulle applicazioni con interfaccia grafica (GUI).

Comunque, il potenziale del web di fornire applicazioni su larga scala non ha colpito il mercato tradizionale sino a quando Google non ha introdotto Gmail, immediatamente seguito da Google Maps, applicazioni web con *rich user interface* e un'interattività equivalente a quelle per PC. L'insieme di tecnologie utilizzate da Google fu [battezzata AJAX](#) in un saggio embrionale di Jesse James Garrett della società di web design Adaptive Path. Egli scrisse:

"Ajax non è una tecnologia. È l'insieme di alcune tecnologie, ognuna delle quali fiorente di suo, unite in modi potenti e nuovi. Ajax incorpora:

- [Presentazioni standard](#), utilizzando XHTML e CSS;
- Visualizzazione dinamica e interattività utilizzando il [Document Object Model](#);
- Scambio e manipolazione di dati utilizzando [XML e XSLT](#);
- Recupero di dati asincroni utilizzando [XMLHttpRequest](#);
- e [JavaScript](#) che collega il tutto insieme."

AJAX è anche un componente chiave delle applicazioni Web 2.0 quali Flickr, ora parte di Yahoo!, delle applicazioni basecamp e backpack di 37signals, oltre ad altre applicazioni di Google, come Gmail e Orkut. Stiamo entrando in un periodo di innovazione dell'interfaccia utente che non ha precedenti, in quanto i web developer sono finalmente in grado di costruire rich web application di valore equivalente alle applicazioni locali per PC

Stranamente, molte delle capacità esplorate ora sono state disponibili per molti anni. Alla fine degli anni 90, sia Microsoft che Netscape ebbero una visione del tipo di capacità che ora sono finalmente state realizzate, ma la loro battaglia sugli standard da usare hanno reso complicata la realizzazione di applicazioni idonee per più browser. Fu solo quando Microsoft vinse in modo definitivo la guerra dei browser, e quindi ci fu un unico effettivo browser da sviluppare, che questo tipo di applicazione divenne possibile. E sebbene [Firefox](#) abbia reintrodotta la concorrenza nel mercato dei browser, almeno per ora sugli standard web non abbiamo visto la concorrenza distruttiva che ha bloccato il progresso negli anni 90.

Ci aspettiamo di vedere molte nuove applicazioni web nei prossimi anni, sia applicazioni veramente nuove, sia nuove implementazioni web di applicazioni per PC. Ogni cambiamento di piattaforma fino a oggi ha anche creato opportunità per un cambio di leadership nelle applicazioni dominanti della piattaforma precedente.

Gmail ha già fornito [alcune innovazioni interessanti nelle e-mail](#), combinando i punti di forza del web (accessibile ovunque, profonde competenze nella gestione di database, ricercabilità) con interfacce utenti che somigliano alle interface PC quanto a usabilità. Nel frattempo, altri client di posta sulla piattaforma PC stanno eliminando il problema dall'altro lato, aggiungendo IM e capacità di rilevare la presenza degli utenti online. Quanto siamo lontani da una comunicazione integrata che unisca il meglio di email, IM, e telefono cellulare, utilizzando il [VoIP](#) per aggiungere anche la voce alle ricche capacità delle applicazioni web? La corsa è cominciata.

È facile prevedere che il Web 2.0 modificherà anche la rubrica. Una rubrica Web 2.0 considererà la rubrica locale su PC o su telefono semplicemente una cache dei contatti che esplicitamente, avete

chiesto al sistema di ricordare. Nel frattempo, un agente di sincronizzazione web, tipo Gmail, ricorderà ogni messaggio inviato o ricevuto, ogni indirizzo email e ogni numero di telefono utilizzato e costruirà una tecnica di ricerca all'interno della rete sociale per decidere quale offrire come alternativa quando non viene trovata una risposta nella cache locale. In mancanza di una risposta, il sistema farà una ricerca nel network sociale più ampio.

Un elaboratore di testi Web 2.0 supporterà l'editing collaborativo, tipo Wiki, e non solo di documenti indipendenti. Ma supporterà anche i rich format che ci aspettiamo dagli elaboratori di testi su PC. [Writely](#) è un buon esempio di applicazioni di questo tipo, sebbene non abbia ancora ottenuto un ampio seguito.

Nè la rivoluzione del Web 2.0 sarà limitata alle applicazioni per PC. Salesforce.com dimostra come il web possa essere utilizzato per fornire software come un servizio, in applicazioni adattate alle esigenze dell'impresa, come il CRM.

L'opportunità competitiva per i nuovi entranti è di abbracciare completamente il potenziale di Web 2.0. Le società che avranno successo, creeranno applicazioni che imparano dagli utenti, utilizzando un'architettura partecipativa per costruire un vantaggio competitivo non solo nell'interfaccia software, ma anche nella ricchezza dei dati condivisi.

## **Competenze Centrali delle Società Web 2.0**

Esplorando i sette principi suddetti, abbiamo evidenziato alcune delle principali caratteristiche del Web 2.0. Ogni esempio che abbiamo esplorato dimostra uno o più di questi principi chiave ma possiamo non tenerne in considerazione altri. Chiudiamo, perciò, riassumendo quello che noi crediamo essere il centro delle competenze delle società Web 2.0:

- Servizi e non pacchetti di software, con una scalabilità efficace dal punto di vista dei costi
- Controllo su fonti di dati uniche e difficilmente replicabili che si arricchiscono man a mano che vengono utilizzate
- Dare fiducia agli utenti come co-sviluppatori
- Sfruttare l'intelligenza collettiva
- Influenzare "the long tail" attraverso il customer self-service
- Il software a un livello superiore rispetto al singolo dispositivo.
- Interfacce utenti modelli di sviluppo e modelli di business leggeri

La prossima volta che una società dichiara di essere "Web 2.0," verificate le sue caratteristiche a fronte di questa lista. Più punti otterranno, più si meriteranno questo nome. Ricordate, però, che l'eccellenza in una sola area può essere più efficace di alcuni piccoli passi compiuti in tutte e sette.