Tecnologie Web 2

1 Prima lezione: Introduzione al web

Nel 1945,
nascita di **MEMEX**, è un calcolatore analogico dotato di un sistema di archiviazione, ideato dallo scienziato e tecnologo statunitense Vannevar Bush. Rendeva possibile anche la visualizzazione dei dati inseriti.
Il fulcro innovativo del MEMEX è che riesce a collegare più informazioni, quindi dalla prima si può arrivare alla seconda, mediante un link primitivo, per esempio dalla pagina Indiano si può arrivare alla pagina di descrizione dell'arco. E' il concetto primitivo di hyperlink.
Nel 1960,
Douglas Engelbart inventa L'oN Line System (**NLS**), era un primo computer formato da mouse, tastiera e un device per la scrittura veloce.
Ted Nelson, negli stessi anni inventa la definizione di ipertesto e le sue applicazioni, inoltre allarga la definizione di ipertesto a ipermedia. Nel 1967 lancia **XANADU**, è la prima bozza di Web dove applicava gli ipertesto, il concetto base era point and click. E' stato un progetto molto innovativo, in esso era incluso anche una forma di micropagamenti per le licenze d'uso di programmi o tool di altri utenti, permettendo di allargare il proprio "sito" in maniera semplice e veloce, questa tecnologia non ebbe successo e scomparì. Lo schema degli indirizzi era più potente dell'attuale e i link erano dotati di BIVISIBILITY e BIFOLLOWABILITY. Progetto mai realizzato.
Una tecnologia molto simile è ricomparsa negli ultimi anni con amazon, apple store, ...
Inoltre XANADU permetteva di collegare singole stringhe con un link, che al giorno d'oggi non è ancora possibile. Infine teneva traccia delle varie versioni del sito cosicché non poteva mai accadere che un link fosse broken ma veniva collegato alla versione precedente se non presente. L'unico problema è che non è mai riuscito a realizzarlo completamente, è stato sempre una idea mai messa in pratica in maniera completa! Un altro ostacolo molto ampio per la sua realizzazione è che l'idea era stata brevettata e quindi nessuno poteva lavorarci liberamente, se fosse stato free magari qualche altro gruppo

avrebbe tentato la sua implementazione.

Nel **1980**,
Tim Berners-Lee riciclo tutte le idee precedenti e a ruota libera mise delle idee su carta. Nel 1989 scive un documento: information management and proposal. In questo documento descrive la **prima idea di Web**, rappresentato nell'immagine seguente.

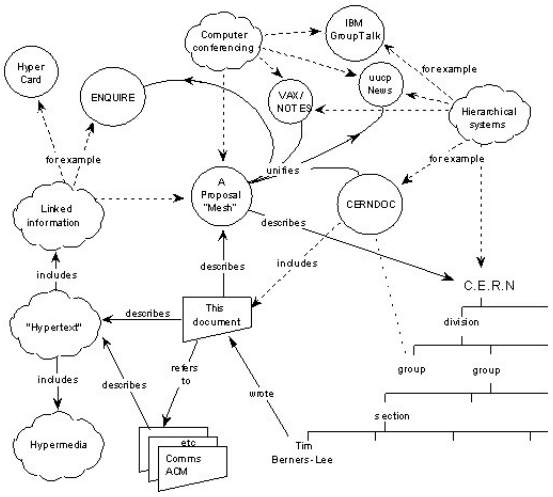

Figura 1: 1980 La prima idea di Web

Nel 1990 riceve un finanziamento dal CERN di Ginevra e gli viene inviato il miglior computer in commercio, il Next(creato da Steve Jobs).
Dopo vari tentativi sceglie il nome definitivo per il suo progetto: World Wide Web.
Inoltre si crea anche il browser per usarlo, sempre chiamato WorldWideWeb! Questo browser è un po' particolare, infatti permette oltre che alla navigazione, permette l'editing delle pagine e include all'interno del browser stesso un lato server!
Nel 1991 alla conferenza internazionale dell'hyperlink presenta il suo progetto, ma gli viene rifiutato, perché definito troppo semplice!
Il progetto comprendeva:
Archie: è il primo motore di ricerca al mondo, ma non lavora nel web, ma via FTP.
WAIS: è un motore di ricerca all'interno di un documento di testo (dentro i database sparsi su internet).
Gopher: è un protocollo di rete basato su un modulo server che razionalizza le informazioni e le gestisce in una struttura ad albero accessibile al client. Le pagine web caricate in Gopher potevano essere di due tipi, o menù o testo.
Nel 1992 venne inventata **Veronica**, un motore di ricerca per Gopher. Visto il continuo aumento popolarità di Gophie, Tim decide di renderlo a paga-

mento. In breve tempo perse molta della popolarità e sparisce.

Nel 1993,
il WWW raddoppia i suoi utenti, perché è gratis! Il browser di riferimento è **Mosaic**, rispetto agli altri browser visualizzava le immagini all'interno della pagina e non in un link separato. Non era specificato in nessun standard che le immagini dovevano essere visualizzate all'interno della pagina ma delle volte è la grafica che conta e non il sistema!
I server presenti in WWW passano da 50 a 200 circa!

Nel 1994,
il traffico gestito dal CERN aumenta del 2000%, si passa da 200 server a 2200, il WWW ha una crescita esponenziale.
Il CERN decide di non finanziarlo più!!! Allora a grande richiesta viene spostato l'intero progetto al MIT dove viene fondato **W3C**.
Il WEB è nato in Europa, ma lo abbiamo regalato agli Americani!!
Il browser del momento diventa **Nascape** Navigator, permette una visualizzazione delle pagine in maniera incrementale, il cambio di browser fa capire che il tempo di attesa è fondamentale.

2 Seconda lezione: usabilità dei siti web

Un sito si deve considerare come una casa o un negozio, si guarda la vetrina e poi si decide se entrare o no. L'homepage può essere vista come la vetrina del negozio, il visitatore vuole delle informazioni e quindi la nostra home deve fornire una sintesi di tutto ciò che offriamo. L'homepage è il cuore del sito, deve essere gestita come un testo giornalistico, quindi deve rispettare le 6W:

- Where
- Who
- Why
- What
- When
- How

Nella scrittura di un sito valgono le stesse regole riadattate un pò al contesto:

- Where: a che tipo di sito sono arrivato, il contenuto
- Who: chi rappresenta il sito
- Why: benefici del sito
- What: offerte del sito
- When: ultime novità, le news del sito
- How: come arrivare alle sezioni principali del sito

Il problema principale di un utente è il tempo. Il sito quindi oltre a rispettare le 6W, deve anche saper riassumere gli argomenti in un numero di parole esiguo per pagina.
Gli utenti hanno delle aspettative ma anche un tempo limitato. Un utente medio resta nella homepage circa 31 secondi, quindi si hanno 31 secondi per convincere un utente a rimanere nel sito visualizzato. Inoltre si deve tener conto che un utente sopra la media riesce a leggere 180 parole al minuto, quindi facendo i calcoli in 31 secondi riesce a leggere circa 93 parole! In realtà si deve tener conto del tempo che perde nel guardare il layout visivo, quindi realmente le parole che legge sono ancora meno!

Inoltre un'altra esigenza minima che ci si deve prefissare è che non solo l'utente resti nel sito, ma che ci ritorni. Ad un suo ritorno cambiano le regole, infatti non si deve più considerare le domande Who, Where e Why, ma si deve rispondere alle domande What, When e How. Inoltre una visita successiva alla prima implica una maggiore esigenza e un minor tempo disponibile!!

Tempi medi di permanenza:

- 2° volta: 25 secondi
- 3° volta: 22 secondi
- 4° volta: 19 secondi

Dalla 4° volta in poi il tempo si stabilizza. Se puntiamo ad un sito ricorrente, significa che abbiamo 19 secondi per rispondere alle domande What, When e How! Quindi circa 54 parole.

Dopo che l'utente si addentra nel sito, quindi lascia la homepage gli assi non servono più, perché si crea un "legame" che rende meno difficile l'abbandono! Infatti nelle pagine interne dedica in media 51 secondi, permette quindi pagine più specifiche, bisogna comunque non esagerare nei contenuti, si hanno a disposizione circa 150 parole. Se l'informazione eccede, si deve suddividerla in moduli della taglia esatta e separarla in più pagine.

Un altro vincolo molto importante è il **tempo globale** di permanenza in un sito, perché l'utente sia soddisfatto, cioè abbia trovato ciò che cercava. Ci sono due categorie principali:

2.1 Tempo preliminare

Il tempo che dedica per farsi un'idea del sito e se restare o no, è detto tempo di scelta.
Il tempo di scelta medio per un sito generico è di **1:49 secondi**, se dopo questo tempo l'utente non è soddisfatto lascia il sito anche se i contenuti c'erano. Quindi riesci a visitare la homepage(31 sec), e due pagine interne(53 sec cad)! Ma se l'utente lascia il sito qual'è la probabilità che ritorni? Nel 88% un utente non ritorna più!
È il tempo più importante in quanto è il tempo complessivo per convincere l'utente a restare con noi.

2.2 Tempo di successo

O tempo di soddisfazione, è il tempo in cui dobbiamo soddisfare l'utente.
L'utente ha deciso di restare, quindi vuole l'informazione che cercava, il tempo medio che dedica prima di essere insoddisfatto è di **3:49 secondi**.
E' molto importante il bilanciamento che si da tra homepage, pagine interne e cammini per arrivare all'informazione. Al primo accesso infatti l'utente visita, in media, la homepage e una pagina interna e poi decide se restare! Guardando l'albero del sito, un utente che parte dalla home in 3 click dovrebbe arrivare all'informazione che cercava.
A meno che...
Negli ultimi tempi la navigazione è cambiata, un tempo si partiva sempre dalla home per navigare in un sito e poi si entrava nelle pagine interne...

Ora invece con i motori di ricerca, al più delle volte si entra direttamente in una pagina interna contenente l'informazione che si cercava, saltando la home. Per gli utenti che arrivano per la prima volta in un sito e direttamente in una pagina interna scombinano un po' tutti i conti fatti in precedenza! Questo fenomeno è detto **Deep Linking**("collegamento in profondità"). Ogni pagina quindi deve essere una piccola homepage!! Valgono di nuovo le 6W con qualche modifica:
Assi opzionali:

- When: a che tipo di sito sono arrivato, il contenuto
- Why: breve descrizione dei benefici del sito
- How: è sufficiente uno search in alto a destra

Assi obbligatori:

- Who: tipicamente visualizza il logo in altro a sinistra e una breve descrizione di chi siamo
- What: link diretto alla home
- Where: non basta il what perché obbligo l'utente ad effettuare un link in più per capire in che sito si trova e quindi una perdita di tempo.

Bisogna inoltre in ogni pagina interna visualizzare la posizione, rispetto alla home o il percorso fatto dall'utente, nel sito.
Si usa la tecnica detta **Breadcrumb**, che letteralmente significa briciole di pane e si riferisce alla fiaba di Hänsel e Gretel che lasciavano una scia per ritrovare la via di casa!
Ci sono 3 tipi di Breadcrumb:

- **location**: mostra ci troviamo in base alla gerarchia del sito, dando una struttura a livelli: home>news>notizia;
- **attribute**: mostra gli attributi della pagina data, quindi non necessariamente corrisponde alla location, ma contiene il percorso dal tipo principale e nei suoi sottotipi(per esempio un sito di vendita di componenti può avere HW / schede grafiche / nvidia / geforce). Ogni pagina ha dei tag che aiutano a capire la categorizzazione;
- **path**: mostra il cammino fisico fatto dall'utente per arrivare alla pagina. Sarà di tipo dinamico per evitare problemi derivati da quello prefissato (non è detto che un utente arrivi alla pagina seguendo il percorso prefissato).

I separatori classici per i Breadcrumb sono: o il maggiore(>) o la slash(/).

3 Terza lezione: usabilità dei siti web parte 2

In un sito web posso riscontrare due tipi di problemi di usabilità:

- Persistenti
- Non persistenti

Il primo problema **persistente** è: **Lost in navigation**,
l'utente non sa in quale posizione si trova durante la navigazione, quindi manca o è carente l'asse Where. Non aiutare l'utente a ricordare dov'è stato aumenta lo sforzo computazionale.
Una soluzione a questo problema, che è stata vista nel capitolo precedente, sono i breadcrumbs, anche se non soddisfano pienamente le aspettative di un utente. Infatti non descrivono in dettaglio il percorso fatto dall'utente, una soluzione comune è colorare i link già visitati di un colore diverso rispetto ai link da visitare, permettendo all'utente un colpo d'occhio immediato sulle pagine già viste. Se invece lascio all'utente il compito di ricordare dove è già stato, crea malumore e stress nel visitatore, rischiando il suo abbandono.
I link colorati sono stati introdotti da Nescape Navigator e non sono uno standard, quindi non è scritto in nessuna regola che si deve applicare il cambio di colore, ma è una buona norma che è consigliato rispettare per garantire una maggiore usabilità.
Il 74 % dei siti web usa il cambio di colore dei link.

Un'altra esigenza degli utenti è il movimento veloce tra le pagine, infatti per ritornare alla pagina precedente quasi tutti gli utenti utilizzano il tasto back anche se esiste un link di ritorno. Minimizzano lo sforzo computazionale anche se per tornare, per esempio alla home servono svariati click e invece esiste un link diretto per essa! Un utente medio è disposto a cliccare fino a 7 volte per tornare alla home anziché usare il tasto di indirizzamento diretto! Non devono spendere nessun sforzo per cercare il link di ritorno! Questo uso comune è detto backtracking, cioè la continua visita dei link con ritorno tramite back.

Un secondo problema persistente è l'aprire più di una finestra di navigazione, anziché usare sempre la stessa. Innanzitutto l'uso del back button non funziona più, inoltre, crea molta confusione nell'utente medio. Possono essere aperte due tipi di finestre, o sotto forma di tab o nuove finestre vere e proprio. La scelta peggiore è l'apertura di una nuova finestra, l'utente infatti può non sapere come tornare indietro e nel caso la finestra non prenda tutto lo schermo la lascia aperta e clicca sulla finestra precedente, mettendola in background. Se in un successivo momento l'utente clicca sullo stesso link della finestra in background il browser aggiornerà essa, facendo vedere a schermo nessun cambiamento. Questo porta molto disagio nell'utente e quindi assolutamente da evitare!

Un altro problema sono i pop up, che creano difficoltà di navigazione e disturbano per la non voluta apparizione. Sono nuove finestre che compaiono senza il permesso dell'utente.
Altro problema è il non rispetto delle convenzioni, come il colore dei link che non sono standard obbligatori ma che aiutano di molto la navigazione.

Altri problemi:

- Non rispettare l'asse what;
- Uso di un linguaggio vuoto o con pochi contenuti;
- Se il contenuto è importante allora è importante anche la forma.

3.1 Legge di Jakob

Gli utenti spendono la loro maggior parte del tempo su altri siti web e non sul vostro!
Questa legge riassume il concetto di non creare siti troppo originali, che possono creare problemi nella navigazione e scoperta del sito! Ma attenersi il più possibili alle convenzioni.

4 Quarta,quinta, sesta lezione: Il contenuto...

I problemi riguardanti il contenuto sono del tipo **non persistenti**. Un utente medio si aspetta che il testo contenuto in un sito sia ricco di contenuti, non sia un testo privo di significato tipo ”polichese”. Un altro punto molto importante è la forma del testo, non deve essere ne troppo complicato ne troppo elementare. Infine si devono sempre rispettare, come spiegato precedentemente, i timer. Certi siti possono non rispettare i timer, perché magari sono unici del genere, per esempio i siti Governativi.
Il testo web deve essere ancora più leggibile rispetto ad altri testi, per vari motivi di lettura a schermo.

4.1 Regole testo web

Se un testo comune, su carta o su altro mezzo, usiamo 100 parole, un testo web per essere di pari difficoltà di lettura, comprensione e velocità di apprendimento deve essere di 50 parole. Se si vuole che sia comprensibile a tutti, deve essere di 25 parole rispetto alle 100 iniziali.
Un buon metodo di scrittura di un testo web è partire dalla conclusione.
Problemi non persistenti:

- **pagine splash**: è la pagina iniziale di benvenuto, è da evitare a tutti i costi, 80% degli utenti la detestano, inoltre i timer scorrono e crea solo perdita di tempo ancora peggio se sono animate. Un ulteriore peggioramento delle pagine splash è la pagina di registrazione iniziale. Crea sforzo computazionale per ricordare nome utente e password, inoltre, se la registrazione viene richiesta nella pagina splash può provocare abbandono del sito perché richiede una fiducia nel rilasciare i propri dati.

- **scrolling**: un utente medio è disposto ad usare lo scroll per 1,3 schermi, quindi 2,3 schermi totali. Nella prima pagina il 23% scrolla, invece nelle pagine interne il 42%. La seconda visita alla home, scrolla il 14%! La taglia di riferimento per un sito è 1024x768, ma la taglia minima da considerare è 800x600. Spesso, soprattutto nei netbook, data la loro piccola dimensione o usando del layout fisso (**frozen layout**), ci si concentra troppo sulla taglia di riferimento e ci si dimentica delle più grandi. Oppure ci si focalizza troppo sull’asse verticale e si crea scroll orizzontale, creando tantissima difficoltà computazionale. Lo scroll orizzontale infatti, cambia la modalità informativa classica, nel quale l’asse X è fissato. Molto spesso l’informazione visibile dopo aver scrollato orizzontalmente è persa per sempre!

4.2 Problemi creatore sito, utenti

Bloated design: si definisce un design di un sito quando è troppo spinto! Molto bello ma esagerato, difficile da usare e che crea un’alta difficoltà computazionale.
La guerra dei browser ha portato a introdurre comandi nuovi, spettacolari ma eccessivi per la comprensione di un utente medio. Per esempio

blink(testo lampeggiante), testo che ruota, testo in movimento, ecc..
Un esempio di design esagerato è un design con il menù in un albero di ovetti. Sei utenti su dieci non lo riconoscono e quindi se ne vanno dal sito. Oltre ad abusi visivi, ci sono **abusi multimediali**. Un esempio è l'**audio** nel sito, odiato da quasi tutti gli utenti.
Un altro esempio è il sito con **effetti 3D**, anch'esso è odiato da molti utenti, è preferibile usare il 2D classico (implica uno sforzo computazionale minore).
Un altro problema è l'uso dei **plugin**. Il loro problema principale è che non sono standard e quindi richiedono una installazione per il corretto funzionamento del sito e quindi sforzo computazionale. Un plugin per essere usato si dovrebbe aspettare almeno un anno dal suo rilascio (fattore TRUST, il pulgin non sembra una cosa sicura), anche se crea comunque problemi, infatti il 90% degli utenti che incontra un sito con plugin non li installa ma abbandona il sito.
Un esempio di multimedia da usare in tono moderato è l'inclusione di **video**. Può funzionare, ha scarso sforzo computazionale ma non altera il timer degli utenti. Il problema principale è che serve molta banda per vederlo in un tempo ragionevole anche se non entra nei timer. Il tempo medio di un video non deve superare i 2 minuti, il tempo ideale è di 1 minuto! Ovviamente fatta eccezione per i siti dedicati ai video, per esempio youtube, qui i timer non sono validi.
Metafore visive: certi simboli o segnali che implicano una cosa, invece vengono delusi. Per esempio un link finto, cioè un link non cliccabile! Metafore metafisiche: certi oggetti che nella realtà funzionano in una certa maniera e nel sito cambiano il loro funzionamento. Per esempio, un bottone nella realtà è cliccabile in ogni suo punto, invece certi siti è possibile cliccarlo solo sulla parola scritta al suo interno. Crea problemi di usabilità.
Un altro altro problema è di portare degli **aspetti desktop nel web**. Si è portato ormai da un po' di tempo l'oggetto menù nell'ambito web. Il problema è che non sono proprio identici, infatti il menù desktop contiene comandi, invece il menù web contiene informazione. Questo può portare ad una esplosione dei menù.
Un altro problema legato ai menù è la taglia di referenza, il menù è troppo grande sullo schermo, creando difficoltà di visualizzazione. Inoltre per visitare il menù 83% degli utenti non riesce a centrare le caselle! Ed il 54% esce erroneamente, questo succede perché la mente umana pensa sempre il percorso minimo da A a B e se il menù è aperto a cascata e si chiude all'uscita con il mouse dall'area del menù stesso, percorrendo il percorso minino si uscirà sicuramente creando frustrazione e sforzo computazionale. Un menù senza timer di chiusura usato da un utente per la prima volta, provoca il 92% di chiusure errate!
Il numero massimo di sottolivelli consigliati nei menù è 2 e avere fault tollerant (ovvero se esco dal menù questo non si chiude subito).

Nel testo, oltre a tutte le regole dette precedentemente, si deve stare attenti alla grandezza del testo stesso e del contrasto con lo sfondo.

Regole del testo:

- Testo leggibile, grandezza minima 10pt
- Dare opzioni per il resize del testo, il modo migliore è con le varie grandezze già cliccabili.
- Il testo è testo, l'utente deve riconoscere che è testo e quindi lo possa usare come tale, con le più comuni operazioni(copy, past, taglia, ecc).
- non usare troppi font e troppi effetti, la regola d'oro è di usare un solo font per tutta la pagina, massimo due. Font preferito Verdana.
- Tenere un contrasto molto alto tra testo e fondo.
- Il testo scritto in maiuscolo si legge più a fatica anche se porta più enfasi, un utente medio che legge un testo in maiuscolo ci impiega il 10% in più.
- evitare l'uso di immagini contenenti testo al posto del testo comune, reagisce male al resize, aumenta il peso della pagina, non permette le operazioni comuni, interagisce malissimo con i motori di ricerca.

4.3 Maledizione di LOREM IPSUM

I limiti del bene e del male.
Normalmente un design prima crea il layout e poi inserisce il testo, per testare il layout con del testo inserisce parole a caso dette appunto lorem ipsum. Poi in un secondo momento le sostituisce con il vero testo.
Si porta quindi, a dividere il layout dal contenuto, dando priorità al primo e adattando il secondo. Un utente quando visualizza per la prima volta una pagina web non inizia a leggerla ma la mente fa uno scanning veloce dell'intera pagina e poi in un secondo momento inizia la lettura. Un buon design tiene conto dello scanning, che viene fatto in maniera continua e ne minimizza lo sforzo. Grossi blocchi di testo aumentano la lentezza di scanning, occorre dare una struttura al testo per aumentarne la velocità.
Quindi il lorem ipsum crea problemi, infatti prima devo conoscere il contenuto e poi creare il layout.
Come si **struttura**:

- testo suddiviso a blocchi
- strutturare i blocchi con dei **titoli descrittivi**, aiutando la catalogazione primaria fatta dallo scanning
- Si possono usare **parole chiavi** evidenziate(bold) per aiutare la mappa virtuale, senza esagerare sennò si crea l'effetto contrario. Devono esere brevi e pertinenti. Un utente medio memorizza 6/7 keywords.
- I **link** non devono essere troppo lunghi e contenere il titolo della sezione perché crea confusione.

• non usare link con nomi troppo simili, ancora peggio usare l'indirizzo fisico della pagina

• mai usare come nome di un link clicca qui, aumenta solo la fatica computazionale, è ovvio che un link va cliccato!

• usare **liste** nel testo, crea ordine, aiuta la memorizzazione e aumenta la soddisfazione dell'utente di ben 47%, si devono usare con almeno 4 elementi. Evitare di usarne troppe, diminuiscono la loro memorizzazione linearmente al numero di liste presenti(3 liste rende 1/3 rispetto a 1 sola) verticalmente. Il numero di liste orizzontali invece diminuisce l'effetto di memorizzazione esponenzialmente!

• estremi di LOREM IPSUM, il testo inserito è troppo grande rispetto al layout creato e quindi si crea l'effetto ghigliottina, il testo viene tagliato, oppure si crea scroll interno al paragrafo, da evitare assolutamente.

• evitare l'**effetto bionda**, un paragrafo con un titolo o un immagine molto interessante ma poi il testo non contiene sostanza, o non è coerente con il titolo o l'immagine che lo rappresentano.

4.4 E-commerce

Di un sito commerciale si pensa comunemente che la cosa più importante sia il prodotto, ma non è del tutto vero. In verità è il prodotto ed il prezzo! Gli utenti vogliono il prezzo ed il prodotto accanto visibili insieme. Il prezzo, come in un negozio reale va posizionato accanto al prodotto.
Problema della **iper-associazione**: si mostra una sintesi dei prodotti senza info e prezzo e se un utente vuole saperlo devi cliccarci, creando sforzo computazionale. Di conseguenza l'utente non sa creare rapidamente l'associazione prodotto-prezzo.
Questo problema crea **gambling click** (scommessa): sono i click a scommessa e non è detto ,quindi, che l'azione sia completata.
Questo problema crea il 40% in meno di gradimento del sito. Il 30% degli utenti non clicca mai sulle immagini dei prodotti, che di solito contengono il prezzo! Quindi da evitare e riportare il prezzo allo stesso piano del prodotto.

Se consideriamo il sito di una ditta produttrice di un settore di prodotti ma che non vende al dettaglio, normalmente non visualizza i prezzi. L'utente non è soddisfatto anche se le ragioni sono ben chiare. Bisogna sempre visualizzare il prezzo se si presenta un prodotto, si può dare un prezzo approssimato, o un range di prezzi.

4.5 Pubblicità classica e pubblicità web

Nella pubblicità classica l'utente legge il messaggio pubblicitario per poco tempo, quindi va impressionato e colpito. Ci sono vari modi per invogliare, tra qui invogliare sul prezzo. Nella maggior parte dei casi vengono usati due metodi:

- **fishing price**, prezzo esca, si espone un prezzo "falso" per invogliare l'utente;
- **net price**, prezzo senza tasse o costi aggiuntivi obbligatori sul prezzo finale.

In entrambi i casi il prezzo finale non è quello presentato. Molte campagne pubblicitarie del web utilizzano queste due tecniche, sbagliando. Per capire il motivo di questo errore si deve partire dal cervello, esso è formato da due parti, memoria a breve termine e memoria a lungo termine. la pubblicità sfrutta la short memory, fa capire che c'è un affare in quell'articolo e in quel negozio, tutto il resto si cancella, compreso il prezzo falso!
Questo funziona perché il lasco di tempo tra la lettura della pubblicità e l'ingresso nel negozio è abbastanza per cancellare quasi tutto dalla short memory, ma nel web come in un negozio la distanza temporale e fisica è talmente corta che non si dimentica niente, quindi crea l'effetto contrario. Un sito che utilizza per esempio il fishing price viene abbandonato dal 90% degli utenti e il restante 10% resta ma con una notevole perdita di fiducia nel sito.

Il net price invece 85% degli utenti se ne va, in questo trucco oltre alle tasse non vengono incluse le spese di spedizione e l'eventuale assicurazione sul prodotto.
Una terra di mezzo è specificare il prezzo finale e visualizzarlo nel carrello, questo crea gambling click, perché l'utente spera che siano specificate le spese, metodo non ottimale.
Un'altra cosa invece da valorizzare che spesso non è specificata con adeguata enfasi sono i prodotti gratis. Tipo il catalogo di un sito che viene spedito gratuitamente a casa, va esplicitato il termine GRATIS con caratteri cubitali, in quanto provoca piacere!

4.6 Prodotto

Il prodotto ha la stessa importanza del prezzo, l'errore molto comune che si commette è pensare che un utente conosca già il prodotto, fornendogli le informazioni al minino su di esso. L'utente medio ritiene fondamentale una descrizione dettagliata di esso oltre che al prezzo. Se non trova la descrizione cerca, tipicamente, in altri e-commerce per avere la descrizione e inoltre il sito viene considerato poco professionale. Il 99% degli utenti guardano un altro e-commerce per la descrizione e non confrontano i prezzi, fino al 20% di risparmio restano nel sito con la descrizione soddisfacente!!
Oltre alla descrizione è importante la visione del prodotto. Quando l'utente vuole vedere in dettaglio un prodotto i timer si spengono! La visualizzazione in dettaglio deve essere opzionale, se forzata restano i timer!
Se un utente è interessato al prodotto vuole vederlo anche in full screen, se invece non è interessato il full screen crea solo fastidio.
Un accorgimento da tener presente è di non usare mai immagini riempitive per il prodotto ma usare sempre immagini reali e magari in proporzione.
L'idea è di dare più angolazioni possibili, quindi dare una visione 3D con immagini in 2D (Amazon).
Rispettare sempre le regole principali di usabilità, quindi mai aprire una nuova finestra per i dettagli. Si deve ricordare che c'è sempre bisogno di dettaglio!

5 Settima lezione: Studio dell'attenzione sul testo e immagini

Nel 1990-1991 è stato condotto uno studio comportamentale sulla lettura e focalizzazione dell'attenzione in giornali e riviste. Emerse che esistono delle classi di comportamento ben precise, la parte più attrattiva sono prima di tutto le foto e poi il colore usato nella pagina, una pagina con parti colorate e immagini colorate rende l'idea di una ricchezza informativa maggiore, rispetto ad una pagina in bianco e nero!
Tra testo e immagini in un quotidiano prevalgono le immagini, 80% rispetto al 20%. Se abbinati insieme gli articoli con immagini saltano più all'occhio rispetto ad un articolo di solo testo.
Nella prima pagina di un quotidiano viene innanzitutto guardata l'immagine iniziale e poi letto il testo del titolo anche se scritto in caratteri cubitali comunque prima si guarda l'immagine. Un'altra caratteristica di un quotidiano che crea molto piacere e che non è ancora stata riprodotta nel web è la possibilità di visualizzare due pagine contemporaneamente, aprendo il quotidiano a metà e rendendolo una unica grande pagina (due pagine aperte sono percepite come una pagina)!

5.1 Il Web

Successivamente venne condotto uno studio simile al precedente nell'ambito web.
Si pensava che fossero valide le regole sopra descritte, invece non è vero, si deve considerare il web come un altro mezzo di comunicazione.
Il punto di maggior interesse e di prima visualizzazione è il contenuto in alto a sinistra (non più l'immagine più grande come nei giornali). Di solito proprio in alto a sinistra si posiziona il logo di rappresentazione della società del sito senza testo, questo crea disorientamento e malumore nell'utente.
Analizzando una pagina con un **eye tracker**, cioè un lettore di sguardo, si è potuto avere i punti di fuoco di una pagina generica di un utente medio: si crea una termografica ad F o a cono di gelato!
Questo nella prima visualizzazione di una pagina, ma poi con lo scroll?
Lo scroll viene quasi esclusivamente fatto per schermata, quando si effettua uno scroll si crea un punto cieco nel punto in cui si effettua lo scroll ed è dipendente dalla taglia dello schermo, infatti si vede dalle termografie che dove si scrolla il testo non viene quasi guardato.

Tra testo e immagini nel web vince il testo!
Come migliorare una pagina web:
Impaginazione a più colonne è da evitare perché crea molta difficoltà nello scanning.
Parole chiave: se si hanno più parole chiave sulla stessa linea si crea un effetto faticoso per la fase di scanning (max una per riga), se si usa solo il bold per le parole chiave diventa pesante, è consigliabile o usare la sottolineatura ed il

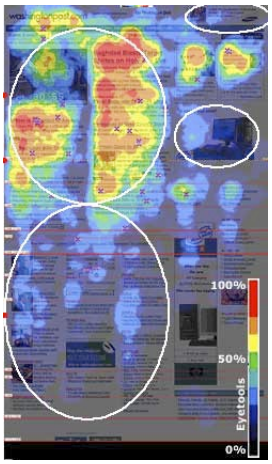

Figura 2: Un esempio di termografia a cono

bold o ingrandire il testo unito al bold o separare la riga della parola chiave per conto suo, facendola diventare una specie di minititolo.

Paragrafi: i paragrafi corti attraggono il doppio rispetto ad un paragrafo unico con lo stesso testo. Molti paragrafi piccoli rilassano i timer e invogliano alla lettura, del 100% in più!

Titolo: un titolo corto è più di effetto rispetto ad uno lungo. Un'altra miglioria del titolo è aggiungere sotto il sommario(blurb). Si ha quindi la possibilità di avere un titolo più corto e nel blurb ampliare la sua spiegazione. I **Blurb** aumentano i punti di fuoco nella mappa termografica, portano a una maggiore digestione dell'informazione, ma non aumentano la voglia di proseguire nella navigazione del sito. Aumentano però il tasso di ritorno(+20%). Ogni blurb ha la sua microzona: essendo asimmetrici la parte più a sinistra è la prima letta, quindi più importante, rispetto alla parte destra.

Il layout deve essere **compatto o no?** Un utente appena acceduto in una pagina preferisce il largo, ma poi nello scroll vince il compatto. Quindi si deve trovare un compromesso tra i due, infatti:

Separazione = scanning più veloce, consigliato nelle pagine per la navigazione, ricche di link.

Compatto = migliore apprendimento del contenuto, se ci fosse separazione si creerebbe **diluited design** che porterebbe solo fastidi.

Immagini

Sono sempre di serie B rispetto al testo ma anch'esse importanti. L'utente vuole vedere l'immagini in una formato abbastanza visualizzabile, minimo 210x230 px, se l'utente è interessato all'immagine.

Immagine vs testo, preferisce sempre il testo, anche se un'immagine è posta nella zona calda! Le immagini però attirano i click, anche se poco viste il 20% degli utenti clicca sulle immagini anche se non specificato che ci sia

un'azione. E' opportuno quindi rendere sempre le immagini cliccabili.
Un'altra caratteristica importante è la via di fuga da una pagina che deve essere sempre ben visibile per non creare difficoltà all'utente.

La mappa termica è uno strumento importante ma non tiene conto dei timer, quindi del tempo!

Legge di fitts:

$$T = a + b * \log_2(1 + D/W)$$

Questa legge è molto usata, infatti ha una corrispondenza del 98%!
Indica il tempo che impiega un utente a portare il mouse da un punto A ad un punto B.
a= start/stop, tempo di accensione
b= co-velocità (lentezza) – quanto siamo veloci a muovere il mouse
D= distanza di percorrenza
W= ampiezza, grandezza della destinazione

Quindi a basandoci sulla legge precedente è da preferire un point and click o un drag and drop?
Sicuramente il **point and click**, infatti il drag and drop ha una velocità più bassa di esecuzione dovuta alla tensione muscolate, inoltre aumenta la distanza tra a e b in maniera lineare.

Delle buone norme da rispettare sono:
Minimizzare la distanza tra A e B e creare destinazione abbastanza grandi, in genere si deve cercare di creare una via di mezzo tra le due.

Con queste regole si può comprendere ancora meglio perché un utente medio non ami i menù in un sito e perché gli creano frustrazione nell'usarli. Quando si è visto che la legge di Fitts funziona, si è cercato di rispettarla creando un menù bilanciato, ogni sottomenù si centra, permettendo una strada diretta ad esso.
Inoltre sempre per rispettare Fitts è stata creata la **target size rule**: un bottone più usato viene visualizzato di una grandezza maggiore rispetto ad un bottone meno usato(per esempio office). Un altro stratagemma per rispettare la legge di Fitts è usare i bordi dello schermo. Fitts infatti vede i bordi come un bottone infinito, quindi molto efficiente. Con questa legge è più efficiente per esempio usare un bottone sul bordo dall'altra parte del schermo che usare un bottone semplice a pochi pixel di distanza. L'uso dei bottoni a bordo è molto sfruttato dall'interfaccia Mac, si è calcolato che usare il menù Mac è 5 volte più veloce di un menù Windows.
Il touchpad, può sfruttare anch'esso la legge di Fitts, sfruttando i bordi laterali per operazioni particolari, come per esempio lo scroll.

Gli angoli sono ancora meglio, hanno un tempo di accesso migliore perché per raggiungerli ci si impiega ancora meno tempo.
Per facilità d'uso, dal più importante al meno importante:

- in basso a destra
- in alto a sinistra
- in alto a destra
- in basso a sinistra

Questo per quanto riguarda i destri-mano, per i mancini è simmetrico.
Ancora più importante e che rispetta al massimo la legge di Fitts è il punto dove ci si trova! Sfruttare quindi il tasto destro per visualizzare dei popup menù in quanto hanno il top di accessibilità.

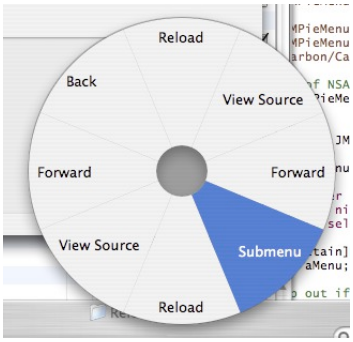

Figura 3: Un esempio di Pie Menù

Il menù migliore è il **Pie Menù** clicco con il tasto destro e appare una torta attorno al punto cliccato, con i vari campi alla stessa distanza. Riduce lo sforzo computazionale.
Una variante dei Pie Menù sono i **Fun Menù**, sono molto simili al Pie, ma si trovano non nel punto di fuoco ma negli angoli o bordi dell'applicazione. Ovviamente il Pie Menù rispetto al menù classico è migliore. Una variante è la combinazione dei due.
I menù **Lineari** invece funzionano bene se hanno tante voci o quando la loro descrizione richiede molto testo.
L'applicazione di Fitts e delle sue regole è molto usata nei videogiochi, perché devono essere il più usabili possibile.

6 Ottava lazione: La pubblicità

Gli utenti odiano la pubblicità, solo il 0.4% ci clicca sopra!
Come fare per aumentare la percentuale di click, innanzitutto deve essere bella ed attraente, ed avere un buon posizionamento, **posti migliori**:

- colonna di sinistra

- top della pagina
- colonna di destra

Il posto peggiore è a piè di pagina.
La pubblicità è importante perché fa parte del modello di business classico delle applicazioni web. Permette di rendere un servizio gratis, il che comporta un aumento del numero di utenti e i ricavi si ottengono tramite essa. La pubblicità dentro il nostro sito, solitamente, proviene da siti esterni.
In alternativa, se un messaggio pubblicitario viene posto tra del testo interessante, viene notato maggiormente. Anche la forma e la taglia del testo stesso influenzano positivamente.
Da evitare assolutamente:

- 10° suona in automatico, da un messaggio sgradevole 79%
- 10° in movimento 79%
- 9° lampeggiante 87%
- 8° occupa tutta o gran parte della pagina 90%
- 7° si sposta sullo schermo 92%
- 6° non dice di cosa si tratta 92%
- 5° copre il testo che stai leggendo 93%
- 4° non si capisce come toglierlo 93%
- 3° cerca di farsi cliccare 94%
- 2° si carica lentamente 94%
- 1° pop up 95%

Allora bisogna usare altri stratagemmi per attirare lo sguardo degli utenti, nel reale le persone belle o i colori vivaci attirano, ma nel web? Le immagini come già detto sono di serie B, perché si attiva nel nostro cervello l'**effetto zapping**, si pensa già vedendo un'immagine che sia una pubblicità e si scarta. Si può evitare l'effetto zapping proponendo persone comuni che si comportano in maniera strana, o che disprezzano il prodotto stesso! (un esempio di effetto opposto è il grande fratello).
Se inserisco una pubblicità nel mezzo di un testo interessante, ma l'immagine ha un bordo, creo separazione e quindi si attiva l'effetto zapping. Un esempio positivo invece è l'effetto **Blending**, si mescola il contenuto normale con la pubblicità, si confonde il training set. I giochini web sono un buon modo per attirare gli utenti ed hanno un'alta efficacia anche se banali! Google per esempio usa il Blending inserendo in ogni risultato delle ricerche, al posto dei primi tre risultati, messaggi pubblicitari, separati dai risultati reali dal un bordo quasi invisibile color giallo chiaro (obbligatorio per legge).
Altri effetti per attirare l'attenzione:

- Se nell'immagine c'è qualcuno che guarda qualcosa, gli utenti si concentrano su quella cosa.
- Nelle immagini con persone visualizzate per intero, ci si focalizza di più sulle parti intime.

Bisogna prestare attenzione a non inserire pubblicità troppo scollegata al contenuto, si crea **distraction effect**, i timer scendono del -40%, crea frustazione e la voglia di ritornare nel sito scende del -80%! Per questo motivo si cerca sempre di più di usare non una pubblicità generalista ma **Behavioral Advertising**, cioè la pubblicità si adatta al contenuto e all'utente che la visualizza, tracciandogli un profilo, quindi registrando gli interessi e hobby. Questo è possibile perché nel web si possono diversificare gli utenti, questo tipo di pubblicità ha un'efficacia fino al 100 volte superiore a quella normale e un fattore di vantaggio di minimo 10 volte! Inoltre questa tecnica non ha effetti negativi sui timer e aumenta la voglia di ritorno di ben 100 volte!

6.1 La ricerca

Soglie critiche per differenziare la ricerca:

- fino a 100 elementi informativi, non è obbligatorio fornire tool di ricerca
- da 100 a 1000 è cruciale dare un buon tool di ricerca
- oltre le 1000 è fondamentale fornire un eccellente tool di ricerca che copra al 100% il sito.

In media un tool di ricerca viene utilizzato dal 100% degli utenti.
Dato che l'utente è abituato ai motori di ricerca, si aspetta di trovare una ricerca pressoché uguale anche nel sito. Inoltre dato che può arrivare in qualsiasi pagina del sito, **Deep Linking**, il 60% degli utenti sfrutta la ricerca immediatamente.

Soluzioni:

Simulare un motore di ricerca, per esempio i tool di google, in locale per il sito, è una soluzione a costo zero. La ricerca però, non funziona bene, perché non progettato per funzionare a bassa scala, se l'utente riusa la ricerca nel sito ed è arrivato con un motore di ricerca, significa che il motore di ricerca non gli ha fornito l'informazione cercata e quindi la ricerca fatta con lo stesso metodo gli visualizzerà lo stesso risultato. Un altro punto negativo è che i motori di ricerca troncano il 50% delle indicizzazione e quindi verrà fatto anche in locale con perdita della metà delle informazioni. La site map non serve a niente per l'indicizzazione.

Soluzione2:

Creare una ricerca locale, ma prestando attenzione a rispettare le abitudini di un utente medio, essendo abituato ai motori di ricerca deve avere un'interfaccia molto simile, con una box di testo e un pulsante cerca, il bottone deve avere il nome preciso search o cerca o ricerca, per non confondere l'utente e deve essere posto a destra (il bottone a sinistra crea un ritardo di 2 secondi).

Less is more.

Mettere troppe cose crea un effetto negativo, un semplice search con una box accanto è il più immediato ed il più apprezzato. Google ha tolto il bottone ricerca avanzata dalla pagina principale perché il solo leggere il bottone creava difficoltà all'utente!

Bisogna essere il più minimali possibile stando attenti a non esagerare.

Mai rimandare un search locale al globale, inviando l'utente alla pagina dei risultati di google!

Ricerca vincolata

Una ricerca con vincoli può essere utile se affiancata alla ricerca classica, mai visualizzata da sola. Essa è efficiente e agli utenti piace, ma bisogna prestare attenzione se renderla statica (si inseriscono i dati e poi si clicca su cerca) o dinamica (ad ogni selezione di una caratteristica visualizza i risultati). Nella **search statica** l'utente applica dei vincoli e funziona solo su esplicita azione dell'utente. Crea confusione, l'utente vedendo un pulsante search cerca il box per inserire il testo, ma non lo trova!! La **ricerca dinamica** invece richiede qualche secondo ogni volta per visualizzare i risultati. Nella ricerca dinamica appena compilo un campo parte la ricerca, vale la pena usarla quando i vincoli sono pochi per non far perdere tempo all'utente.
Molti utenti pensano che la ricerca statica sia dinamica, selezionano la ricerca ed aspettano i risultati, poi si accorgono che bisognava cliccare sul pulsante di ricerca, creando frustrazione.
Si usa la statica con molti vincoli, mentre è meglio usare la dinamica con pochi vincoli. Un'unione dei due è la **ricerca ibrida**, se sono stati riempiti tutti i vincoli la ricerca parte da sola, sennò è statica e parte cliccando sul bottone.

Risultati della ricerca

Nei risultati della ricerca è molto utile permette il sort su tutti i risultati e in tutte e due le direzioni (prezzo da + alto al + basso e viceversa).
Se il risultato contiene 0 dati la migliore soluzione è visualizzare un messaggio di nessun risultato. **God is in the details**, quindi nei 0 risultati per esempio!

6.2 link

I link di una pagina possono rompersi. Si possono scegliere tre strade di comportamento, o si imposta di essere renderizzati automaticamente alla home, o si visualizza una pagina con l'errore, o si lascia al browser la visualizzazione di default del link rotto. L'ultima opzione è la più sbagliata. La cosa più giusta è visualizzare un messaggio chiaro e semplice che la pagina è stata rimossa.
Un caso simile ai 0 risultati si può verificare con l'**effetto 404**. Mai non gestire le pagine non esistenti. Aiutare il più possibile l'utente dando più informazioni possibili, ad esempio aggiungendo una barra di ricerca. Il sito b3ta.com per la gestione delle pagine in 404 hanno preso tutti i più divertenti 404 per creare il loro. Questa soluzione è molto gradita, l'utente medio passa 2 min e 30 a guardare le immagini.

6.3 esposizione

Una visualizzazione dei prodotti a griglia è più compatta e si visualizzano più prodotti in una stessa pagina rispetto a una visualizzazione a lista. Il problema è la ricerca di un prodotto, il tempo per trovarlo è molto variabile, si usa il così detto **random walks** perché il grado di libertà per cercare è troppo grande, avviene una esplosione dinamica nel cervello. Per questo motivo i motori di ricerca usano una visualizzazione dei risultati a lista, perché la griglia crea una grande perdita di tempo nella micronavigazione.

6.4 box di ricerca

All'inizio dell'uso del web gli utenti quando usavano un box di ricerca, inserivano nella maggior parte dei casi singole keywords, con il passare del tempo gli utenti usano sempre più parole. La lunghezza media consigliata per un box di ricerca è di 30 caratteri, questa misura accontenta il 90% degli utenti. (Google da 62 caratteri, Bing 42 caratteri).

- Google: 64 caratteri -> 100%
- Yahoo: 36 caratteri -> 95% (ma la ricerca è in secondo piano)
- Bing: 50 caratteri -> 97% (scelta di design)

Se un box è troppo piccolo, avviene scroll, crea stress negli utenti, l1% per ogni carattere di scroll, inoltre un box piccolo induce l'utente a creare query più corte e quindi meno specifiche, quindi meno precise e risultati più scarsi. Si può usare anche una visualizzazione ibrida, quando il box non è usato si può visualizzarlo piccolo, una volta che si è cliccato sopra si ingrandisce (Volunia).

6.5 Ricerca 2.0

Aumenta la soddisfazione del -43%!!! Crea danno!
Basta ricordare il less is more, l'utente vede l'alto livello di iterazione, quindi le aspettative crescono di molto, ma non è in grado di soddisfarle, quindi frustrazione. Se l'umanoide è più un cartone animato, quindi si vede che è virtuale, l'aspettativa cala e quindi il gradimento sale. Il sito dell'ikea da come alternativa alla classica ricerca la ricerca 2.0 e con una interfaccia 2D base fatta stile cartone, che rende le aspettative molto basse, il gradimento è del +70%!!! Il 10% lo usa.

Voice XML

Il web visto come media visivo è molto limitativo, la voce stà diventando un media.
VoiceXML è nata per passare dallo schermo al telefono, la voce viene tradotta in testo e si interagisce sempre con solo testo direttamente con il computer. VoiceXML usa varie tecnologie:

- Speech synthesis marchup language(SSML), passa dal testo arrichito a voce, quindi da la possibilità di dare gli accenti e tutti gli arrichimenti tonali alla voce del computer, mediante caratteri speciali.
- Pronuciation Lexicon Specification(PLS),
- Call Control XML(CCXML), gestisce l'interazione uomo macchina, è il direttore di orchestra!
- State Chart XML(SCXML), è una evoluzione del CCXML, è più generale
- Speech Recognition Grammar(SRGS), definisce la grammatica di azione, il contesto in cui stà parlando
- Dopo aver applicato SRGS, interpreto il testo risultante.

Il SRGS in voiceXML vengono definiti con un file chiamato < nome > .gram
Il riconoscimento vocale avviene non con VoiceXML ma trammite un engine

esterno. Per una maggiore qualità il riconoscimento vocale di siri avviene in un server dedicato e non internamente al cellulare.
Nel 1966 Eliza(primo psicologo virtuale)
Nel 1975 primo videogioco di avventura, si poteva scrivere liberamente quello che si voleva fare.
Al giorno d'oggi vengono fornite delle frasi già fatte all'utente, il grado di libertà è sceso di molto perché molto spesso non si veniva capiti e quindi si creava forte frustrazione nell'utente.

Inoltre si è notato che far muovere i personaggi rende più piacevole di un ordine di grandezza, i movimenti non volontarri, detti rumori frattali.
Il rumore infatti è la chiave, si crea una curva che parte con una grande ampiezza e poca vitalità e si creano tutti spezzoni continuando ad aumentare la vitalità e diminuendo l'ampiezza. Alla fine si sovrappongono e si crea un effetto di qualsiasi cosa molto reale, per esempio le onde o un terreno o il movimento dell'erba. Queste tecniche rendono ogni cosa più reale!

6.5 Interfacce 2.0

La **componente rumore** crea la sensazione di verosimiglianza. Ha un comportamento frattale e se il rumore è sfasato ci da una strana sensazione. Un'interfaccia fatta con linee a mano, piena di rumore, è più calorosa per l'utente rispetto ad una classica con linee dritte.
I **wearable** per ora non funzionano, interferiscono con la nostra realtà. I **Google glass** hanno un input troppo invasivo e stressante. L'output come popup è negativo e fastidioso e il supporto fisico è pesante sia a livello fisico che psicologico. Gli **smartwatch** hanno maggiori potenzialità rispetto ai GG.

7 Motori di ricerca

SERP: è la lista ordinata dei risultati di una ricerca, che viene visualizzata da un motore di ricerca.
Più del 95% dei click se li assorbe la top ten, in dettaglio:

- 1° posizione prende il 51% dei click, quindi più di tutte le altre 9 della pagina;
- 2° posizione prende il 16% dei click;
- 3° posizione prende il 6% dei click;
- 4° posizione prende il 6% dei click;
- 5° posizione prende il 5% dei click;
- 6° posizione prende il 4% dei click;
- 7° posizione prende il 2% dei click;
- 8° posizione prende il 1% dei click;
- 9° posizione prende il 1% dei click;

La decima posizione...
Effetto Black Jersey(in italiano effetto **malabrocca**)
all'ultimo posto della top ten il click rate raddoppia! infatti i click sono al 2%, la visibilità e maggiore rispetto ai due posti precedenti.

Una caratteristica essenziale per un sito è essere visto all'esterno. Tipicamente gli utenti arrivano in un sito tramite i motori di ricerca. Bisogna quindi essere il più in alto possibile nella graduatoria dei motori di ricerca per avere visibilità.

Mix immagini testo nei risultati creano differenze nelle percentuali di lettura? No, quando l'utente entra nella sezione dedicata alle immagini le percentuali entrano in stallo, una volta ritornato nella pagina dei risultati riprende da dove aveva lasciato.

7.1 Spam Index

Anche detto **SEO** o SEP, calcola il punteggio di una pagina, cioè il punteggio che gli assegna il motore di ricerca per posizionarlo. Esso è formato da una componente testuale(scritto + codice) e una componente ipertestuale.

TFIDF (Term Frequency Inverse Document Frequency)
Dà il peso di quanto una parola sia importante per una pagina. E' formata: TFIDF= TF * IDF

TF
Esempio: data una pagina web formata da 100 parole, "pippo" è contenuta 5 volte in essa, quindi TF=5%
Dando una definizione più formale è la percentuale di occorrenza di una parola in una pagina. Il problema principale usando solo TF, che con parole comuni come per esempio un articolo, il TF è altissimo. Si deve tener conto che se una parola compare troppo è di collegamento e non di contenuto.

IDF
E' l'inverso della frequenza della parola, in scala logaritmica. Esempio: data una pagina web di 1000 parole, l'articolo il appare 980 volte, quindi 98% di frequenza, log(1/0,98)=0,008.

Per calcolare il valore finale di TFIDF= TF*IDF

Dando una definizione semplificata, un motore di ricerca prende tutte le pagine contenenti la parola cercata e calcola di ognuna il TFIDF (solitamente già pre-calcolato), se più di una parola si sommano i TFIDF. Alzare troppo il TFIDF su alcune parole lo fa abbassare su altre, allora si deve cercare di puntare su un gruppo di parole e lasciar perdere le altre.

7.2 Aumentare il posizionamento

Per aumentare il posizionamento, si deve cercare di aumentare il TFIDF delle parole chiave del sito. Ci sono varie tecniche:

• **Body spam**, si mettono le parole nel body della pagina HTML, è efficace ma svantaggia la lettura per l'utente.

• **Title spam**, si inseriscono le parole chiave nei titoli della pagina, il contenuto viene modificato molto meno.

• **Mega tag spam** (<meta name="keywords" content="…">), il vantaggio principale è che il contenuto non viene modificato, ma nei motori di ricerca odierni crea svantaggio!

• **Anchor text spam**, inserire keywords nel testo dei link, il punteggio assegnato rispetto alle comuni parole è maggiore, inoltre, anche la pagina di destinazione riceve dei benefici in termini di punteggio, aumenta i target dei link.

• **URL spam**, inserire le parole chiave nell'indirizzo della pagina stessa, anch'essi ricevono bonus rispetto a comuni parole nel testo.

Tecniche di inserimento delle keywords:

• **Repetition**: la parola si ripete più volte, stando attenti al TFIDF, che se troppo ripetitiva crea danno. Questa tecnica è facile da individuare da parte dei motori di ricerca.

• **Dumping**: inserimento di tantissimi termini poco usati anche se non inerenti, se hanno TFIDF basso saranno presenti in pochissimi posti e quindi il punteggio sarà alto!

• **Weaving**: inserisco nel sito pezzi di altri siti web creando un maggior contenuto interessante, visto dai motori di ricerca come un pregio per un maggior punteggio.

• **Stitching**: si fa copia e incolla di altre pagine web e si assemblano insieme al sito, si crea contenuto interessante per popolare il sito, dato che i motori di ricerca considerano la percentuale di contenuto, questa tecnica aumenta i bonus globali.

• **Broadening**: inserisco non solo le keywords ma anche i sinonimi, le parole simili e frasi correlate. Non solo per coprire le richieste degli utenti, ma i motori di ricerca usano la tecnica di somiglianza delle keywords, le parole dello stesso contesto fanno ricevere alle pagine bonus aggiuntivi. Per determinare se due parole sono nello stesso contesto usano la percentuale di accoppiamento delle parole negli altri siti.

Per scegliere le keywords ci sono vari modi, la più semplice è usare Google Adwords, che visualizza le parole consigliate e la percentuale di ricerche con quelle parole da parte degli utenti.

Usare spam index (text spamming) però per l'utente non è molto gradevole e crea disapprovazione in quanto si cambia il contenuto della pagina. Ci sono varie tecniche per ovviare a questo problema:

Hiding(nascondere), suddiviso in vari sottogruppi:

- content hiding, mettere il testo di spam dello stesso colore dello sfondo
- inserire immagini 1x1px.
- tecnica di **redirection** (tecnica 302), dalla pagina con lo spam a quella pulita utilizzando il refresh. Non è molto efficace ed è facile da intercettare, è molto più efficace se fatta con javascript, perché i motori di ricerca fanno molta più fatica ad interpretare un codice javascript.
- **cloaking**, è un redirection all'ennesimo grado, non usa un redirect, ma controlla se la pagina è chiesta da un utente o un motore di ricerca e a seconda della risposta ricevuta visualizza pagine differenti. Se il motore di ricerca scopre che un sito usa cloaking vienne bannato per un periodo.

Una buona parte di pagerank è data dai link di ipertesto.

7.2 Pagerank

Una buona parte di pagerank è data dai link di ipertesto.

$\pi_v = \Sigma_{(w,v)} \pi_w / \alpha_w$

Calcolato in modo ricorsivo, conta il numero di link entranti per determinare il punteggio. Pi(w) è il numero di link che puntano a v, a(w) è il numero di link della pagina.

Questo calcolo di pagerank era facile da manipolare con i stratagemmi visti in precedenza, quindi si è pensato ad una riformulazione, si è deciso di applicare le **catene di Markov** e **random walks**. Le catene di Markov gestiscono sistemi complessi, in generale affermano che per costruire l'attimo dopo si può approssimare la storia all'attimo prima. Quindi dato che la metafora del web è **random surfer**, cioè girovagare per il web a caso, si è deciso di tener conto di questo fattore casuale:

Calcolo **iterativo**
Dato il seguente web semplificato:
A<−>B<−>C

A=C=1/4
B è doppio quindi 1/2
Si deve cercare di evitare di causare il così detto **spider traps**, cioè la navigazione infinita in un sito dei spider (è il bot di Google che controlla i punteggi) di un motore di ricerca, per esempio: un calendario che se continuo ad entrare nel giorno successivo posso andare all'infinito. Questo problema succede spesso!

Effetto **Island**: un sito con soli link entranti si intrappola di sicuro, il "liquido" pian piano verrà portato tutto nel sito, quindi pagerank 1. Per esempio Microsoft usava questo stratagemma per avere pagerank alto.
Se ho questi due spezzoni di web:

A− > B− > C− > (A)
Q− > P − > R− > (Q)
Verso il liquido non in maniera uniforme, ma un po' di più nella prima isola rispetto alla seconda, il liquido rimarrà sempre in abbondanza nella prima rispetto alla seconda data l'assenza di collegamento. Va cambiato il pagerank.

Calcolo con **Teleportattion**
$\pi_v = (1-\varepsilon)(\Sigma_{(w,v)}\pi_w / \alpha_w) + (\varepsilon / N)$
N è il numero di pagine nel web.
Il bot non fa solamente una navigazione ma va può andare in un punto qualunque del web. Serve per evitare le spider tracks, perché anche se ho tante pagine prima o poi uscirò.
Si è deciso di inserire dei link artificiali per collegare le parti sconnesse, si prende e si teletrasportare in una pagina a caso del web. Se il teletrasporto è disattivato il pagerank è quello originale, sennò è quello modificato (flag teleport=0 o =1).

Calcolo con **Totalrank**

$T = \int_0^1 r(a)\,da$
Fa una media integrale di tutte le possibili scelte. Fa la media tra teleport=0 e teleport=1. Il costo computazionale è lo stesso di quello iniziale!

Esempio
A− > B < − > C
B e C = 1/2
A=0

Rimane sempre il problema dei **dead ends**(strada chiusa)
Una pagina senza link uscenti accumula la maggior parte del liquido anche se si applica il teletrasporto, la soluzione adottata è la penalizzazione! Si è deciso di penalizzare di molto le pagine senza link uscenti cosicchè si abbassa il liquido!
I motori di ricerca per avviare al problema aggiungono dei link, modificando la struttura.

7.3 Norme per aumentare il pagerank

Aggiungere In-link, il pagerank aumenta sempre!
Tecniche:

- **infiltration**: ci si infiltra in altri siti per inserire link verso di noi, per esempio in blog, bollettini, wiki, ecc.
- **honey pot**(barattolo di miele), creo un contenuto appetibile e quindi attraggo link! Si può crearlo facendo copia ed incolla di altri siti.
- **link exchange**, scambio di link mettendosi d'accordo con un altro sito.
- **resurrection**, muore un dominio, si decide di prenderlo e mettere un link alla propria pagina, sicuramente avrà avuto dei link entranti e quindi portano liquido alla nostra.

La pagina da un milione di dollari ha funzionato, in maniera inconscia perché avere un link li significava essere in una pagina con un sacco di link e quindi si riceveva un sacco di liquido.

Aggiungere Out-link
Per simmetria il pagerank si abbassa.
Tecniche:

Socidity, di per se l'aggiungere out-link non aumenta il pagerank ma in realtà si! Perchè l'analisi della pagina può cambiare in positivo!

Spam farm

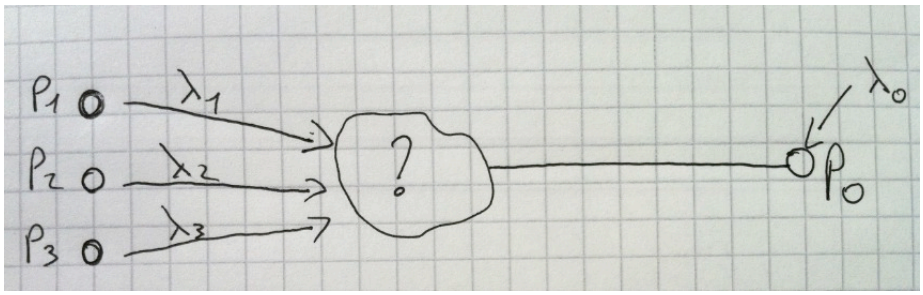

Figura 4: Un esempio di spamfarm

Prendere delle pagine sotto il nostro controllo e farle puntare verso la pagina a cui desideriamo dare importanza.
Come si può notare, in questa spamfarm non è rispettata la **reachability** (ovunque io sono riesco ad arrivare in qualsiasi altro punto), cioè tutte le pagine sono collegate, quindi i spider riescono a visitarle tutte! Per farla diventare vera basta inserire dei link di infiltration, cioè artificiali sui nodi P1, P2 e P3.

Spamfarm ottimale è:

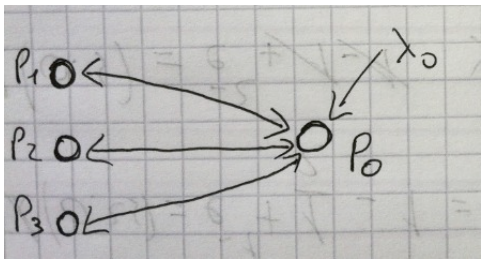

Figura 5: Un esempio di spamfarm ottimale

Dando anche il link di ritorno. Questa spamfarm usa il minor numero di link e mantiene la reachability!

7.4 Le alleanze tra siti

In questa sezione vediamo come le alleanze tra due o più siti possono portare ad un miglioramento ed a una maggiore stabilità del pagerank di entrambi. Ci sono vari modi per collegare due o più siti:

Alleanza profonda

Il pagerank in questo caso è la media del pagerank di ogni sito, il vantaggio

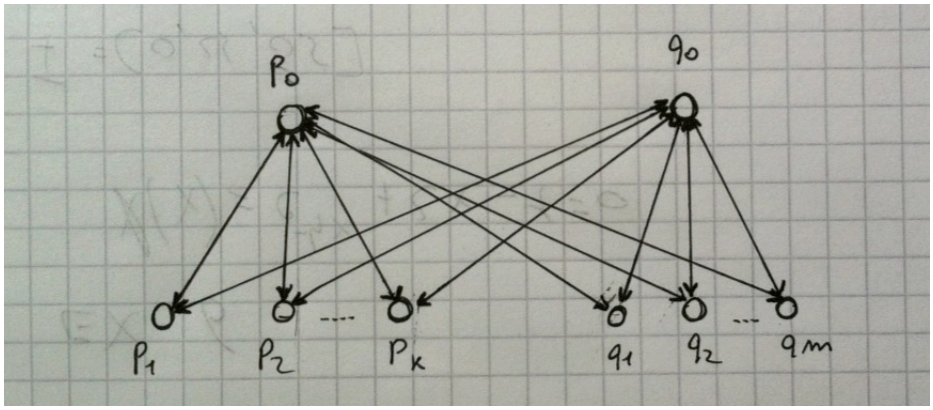

Figura 6: Un esempio di alleanza profonda tra due siti

è che se uno dei due siti perde pagerank, con la media rimane stabile.
Le nostre pagine secondarie puntano sulla nostra pagina target e su quella del nostro alleato e viceversa.

Alleanza superiore

Il pagerank si calcola come il massimo tra i due pagerank dei singoli siti.

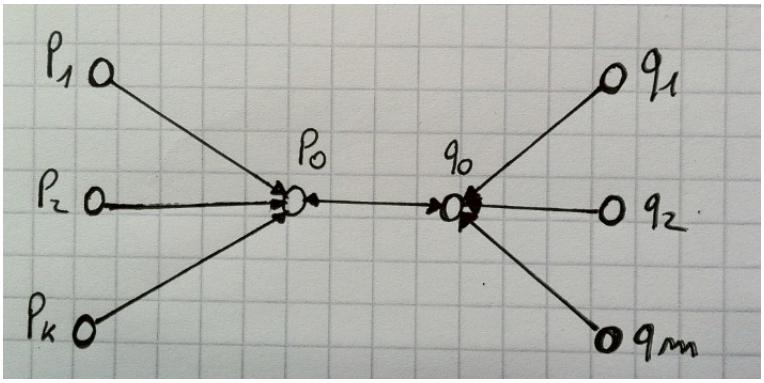

Figura 7: Un esempio di alleanza superiore tra due siti

Struttura a ring

La struttura a ring permette un pagerank alto, ma se uno dei link interni

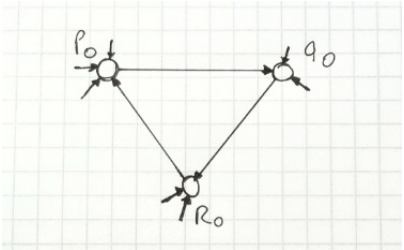

Figura 8: Un esempio di struttura a ring tra tre siti

cade, si creano molti problemi di collegamento e quindi di bilanciamento del pagerank.

Complete came

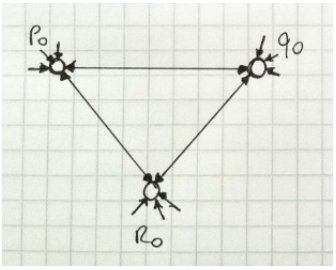

Figura 9: Un esempio di struttura a ring come tra tre siti

7.5 Contromisure dei motori

I motori di ricerca ovviamente devono trovare delle contromisure per limitare gli stratagemmi descritti prima. Sulle strutture complete come ad anello, li identificano e penalizzano tutti i siti che ne fanno parte. Per identificarli basta scovare tutte le pagine focus e le connessioni forti tra di esse. Il problema principale è la quantità di connessioni da controllare, si pensi che per:
N=3 siti, devo controllare 18 collegamenti circa
N=4, 1606 collegamenti
N=5, 565080 collegamenti
e così via, quindi con 10 siti avrò una quantità di collegamenti indescrivibile. Non riesce a scovare le spam farm.
Si è dovuto quindi effettuare dei controlli in maniera diversa, si è partiti dall'affermazione che il web ha una forma, si prende quindi il sito web e si analizza la sua forma, se discosta di molto dalla forma "standard" allora significa che in esso vengono usate delle scorciatoie e viene penalizzato! Inoltre si analizzano i link entranti ed uscenti e tutti i link che si discostano dalla media vengono penalizzati.
Un'altra contromisura è l'uso dei due tipi di pagerank, quello originale e quello con teletrasporto, si calcolano entrambi e la loro differenza è detta **massa di spam relativa**. Se la massa di spam è molto elevata il sito è sospetto e quindi penalizzato. Il rate di successo con questa tecnica è del 95%-100% nel trovare spam.

Guardando un grafico sulla comparazione tra spam e posizionamento si denotano classi abbastanza marcate, al primo posto c'è una minoranza di siti che fa spam e non vengono scoperti quindi il posizionamento è eccellente, il secondo posto invece ci sono i siti che non applicano tecniche di spam, il terzo posto è praticamente vuoto e a metà grafico si trovano la maggior parte dei siti che fanno spam e che riescono ad ottenere un posizionamento medio.

Il pagerank può essere visto come: $(1 - e)1^T N + eL$
con matrici invece: $(1 - e)1^T N + P L$
dove P è una matrice che permette la scelta di una personalized pagerank, i nodi non sono tutti uguali ma hanno delle preferenze.
Google utilizza il personalized pagerank, infatti decide tramite un team appositamente creato di favorire o sfavorire dei siti con teletrasporto a seconda del proprio giudizio personale.
Per sempio searchKing era un concorrente di Google ed è stato penalizzato manualmente!! SearchKing gli fece causa ma la perse dato che il pagerank è una opinione e quindi può essere soggettiva!
Google quindi può fare quello che vuole.

Sarebbe ancora meglio dare all'utente la sua personalizzazione, qualcosa di più vicino a lui, la sua classifica personale, piuttosto che solamente le opinioni di uno (Google).
Ogni profilo personale utente richiede il calcolo di un pagerank ottimizzato per visualizzare classifiche personalizzate, considerando anche il fatto che il profilo e le query che fa sono dinamiche (deve essere dinamico anche il calcolo del pagerank).

La **complessità** di un calcolo personalizzato è: se ci sono N pagine nel web, anche se ci restringiamo a due personalizzazioni per pagina, abbiamo 2^N personalizzazioni possibili. Ci sono delle **buone notizie**: il pagerank personalizzato è linearmente componibile. Posso calcolare i pagerank separati e poi sommarli, senza ricalcolarli. Quindi avremo N pagerank personalizzati (ad es: visito 1000 pagine, allora basta sommare i 1000 mini pagerank di quelle 1000 pagine). Scala ad una complessità N, che è comunque un numero grandissimo (numero pagine web). Nella pratica:

- **Pagerank di tipo topic**: selezioniamo un certo numero di keywords e su quelle pre calcoliamo il pagerank personalizzato. Poi a seconda del profilo utente possiamo costruire il suo pagerank privato dinamico sommando i pagerank personalizzati delle singole keywords.

Bisogna fare attenzione che ogni variante del pagerank che si usa sia compatibile con le contromisure. Il pagerank personalizzato è compatibile con tutte le tecniche basate sulla struttura web.

7.6 Janus graph

Come determinare che un sito sia buono?
Fino a poco tempo fa il rank dei siti si considerava solo la parte buona. ranking= misura di bontà. Il pagerank nella prima versione andava bene finché i siti non hanno iniziato ad adattarsi. C'è stato un cambio di prospettiva in quanto aspetto positivo e negativo stanno sullo stesso piano. Per questo si è deciso di usare un pagerank con due misure associate:
W^+ parte buona
W^- parte cattiva
Quindi il pagerank vecchio, si calcola con la combinazione tra la parte positiva e la parte negativa.
$R_{GOOD}(G^+)$
$R_{BAD}(G^-)$
G è una parte del grafo web contenente molti W.
Dato un pagerank vecchio stile, per calcolare il nuovo **pagerank duale** e dato che la parte positive di G riflette la parte negativa di G posso focalizzarmi sul calcolo di una sola delle due e poi rifretto. Quindi:
$R^J = R(G^+) - R((G^R)^-)$
dove G^R è il web riflesso.

Possiamo ora focalizzarci sul calcolo di W positivo e W negativo.
Si possono calcolare prendendo più fonti informative possibili, per esempio le email sono uno spazio informativo molto ricco. Le email sono già una struttura a rete, quindi basta integrarle. Ogni volta che si menziona un URL si crea un link da un oggetto mail ad un oggetto web e viceversa. La popolarità del link oltre a essere calcolata sul numero di link presenti nelle mail, si determina anche il giudizio negativo o positivo usato nella mail per quel link. Inoltre si determina la popolarità del link a seconda del numero di risposte che la mail contenente l'URL riceve.
Altre informazioni che si possono ricavare dalle mail sono: l'età della mail, la storia misurata sulla quantità temporale di mail, ecc...

Anche la parte cattiva può essere influenzata dalle mail, infatti ogni volta che una mail viene considerata spam e contiene un URL, la cattiveria viene trasmessa al pagerank del sito citato. Quindi il lavoro inizialmente assegnato ad un team pagato per dare un peso ai vari siti ora viene fatto direttamente da milioni di utenti ogni giorno, semplicemente usando il tasto di spam!

Ultimamente i motori di ricerca stanno cercando di collegare sempre di più le parti sociali del web per personalizzare le ricerche, l'introduzione delle mail per esempio ha portato ad una maggiore qualità di personalizzazione, perché le mail sono molto più veritiere, in quanto usano molti sistemi di sicurezza per evitare di essere generate in automatico(captcha), inoltre gli utenti possono essere tracciati e può venir creato un loro profilo(cookies, logs, ecc), le pagine web invece no.

Ancora in via di test è il SIS(social information system) ad ogni oggetto viene associato un VID(user identifier), se due oggetti hanno il VID diverso allora significa che sono di soggetti diversi, invece se uguali, è lo stesso soggetto e quindi si prende uno solo dei due.

Dato che il rank odierno si basa solo su cammini e non considera la parte sociale (**social ranking**) si sta tentando di inserire questa variabile nel calcolo del pagerank. Se per esempio un cammino passa per dei social, significa che sicuramente è di soggetti diversi, quindi più valore. Google ha già iniziato ad integrare questi sistemi.

Se ho un URL che è contenuto in varie pagine ma con lo stesso dominio, significa che è dello stesso soggetto quindi socialmente vale 1. Cade quindi l'unità della pagina web e i cammini possono essere rotti.

Si sta cercando di andare non più verso un unico sistema informativo ma una socio-sfera informativa che contiene tutti i mezzi per avere informazione utile(EMAIL 2.0).

8 torre di babele

10 anni fa si pensava che l'evoluzione del web sarebbe stata utilizzando agenti automatici, che localmente si interfacciavano con il web e svolgevano le richieste dell'utente.
Esempio, l'utente richiede all'agente: manda una rosa alla mia ragazza.
Il problema principale è che 10 anni fa si usava solo HTML che è semplice ma limitato, quindi l'agente non riusciva a comprendere al 100% le richieste, per esempio senza avere una spiegazione precisa del termine rosa, come fiore poteva cercare nel web qualsiasi cosa si chiamasse rosa, e inviare quella con il prezzo minore(per esempio un kit sadomaso!!). Se si fosse usato XML il problema non sussisteva.
Il pregio/problema del XML è che è flessibile, quindi esistono molti "dialetti".
Dato che il web è distribuito, per funzionare bene con agenti automatici bisogna riuscire a creare aggregazione delle varie parti, il problema principale è che se XML usa tutto lo stesso dialetto, funziona bene, altrimenti no. Fallisce in questo caso la filosofia del open world.
Dato che XML funziona ad alberi, è molto difficile aggregare più alberi insieme, quindi l'informazione non funziona bene. In quanto, quando si uniscono due alberi non si ha più un solo albero, ma una foresta di documenti xml.

8.1 Web semantico

Si è cercato quindi di implementare l'aggregazione automatica dell'informazione attraverso strumenti dedicati.

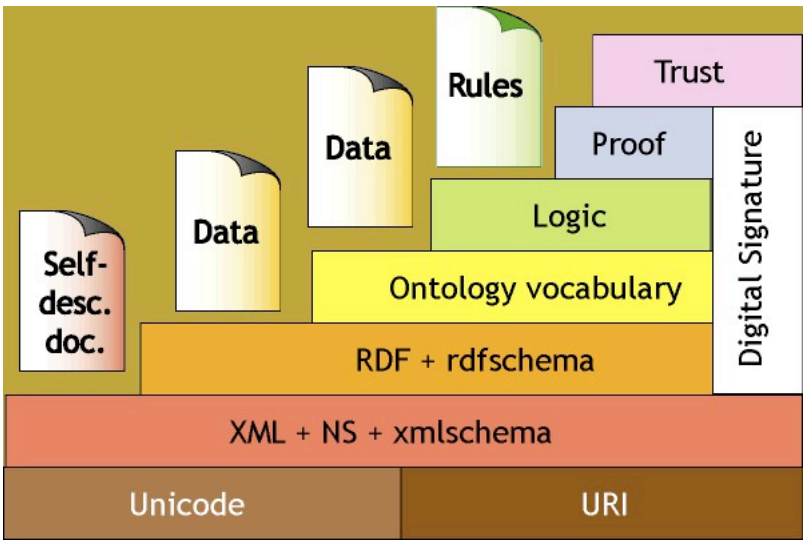

Figura 10: Schema del web semantico

8.1.1 RDF

Lo strato di base rappresenta il linguaggio universale da usare, RDF (Resource Description Framework) descrive le risorse ed è la pietra miliare del Web 2.0, esso descrive relazioni, crea metadati, dati su dati, descrizioni e concetti interoperabili. RDF usa il principio della semplicità, infatti se una cosa non è semplice nel web è già morta! E' un modello Enriched Entity Relationship. Usa la seguente grammatica di base: soggetto, predicato, complemento oggetto. RDF usa i grafi:

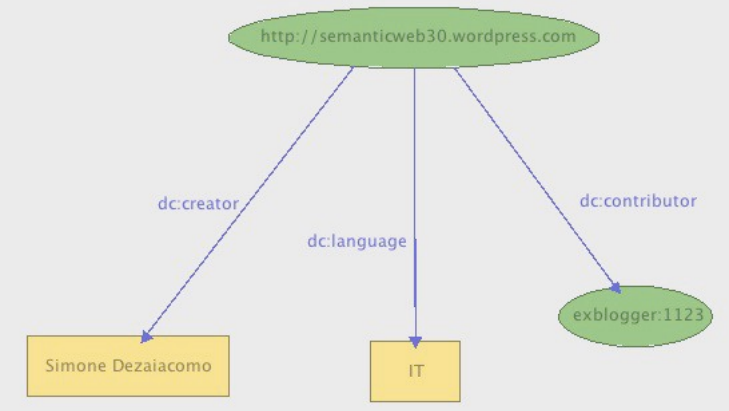

Figura 11: Esempio di un grafo RDF

RDF inoltre implementa due modalità di scrittura:

- XML
- N-triple

XML

```
<?xml version="1.0"? >

<rdf :RDF
xmlns : rdf="http ://www.w3. org/1999/02/22-rdf-syntax-ns\#"
xmlns : cd="http ://www. recshop . fake/cd\#">

<rdf:Description
rdf : about="http ://www. recshop . fake/cd/Empire Burlesque">
<cd : artist >Bob Dylan</cd : artist >
  <cd : country>USA</cd : country>
  <cd : company>Columbia</cd : company>
  <cd : price >10.90</cd : price>
  <cd : year >1985</cd : year>
</rdf:Description>

<rdf:Description
rdf : about="http ://www. recshop . fake/cd/Hide your heart">
<cd : artist >Bonnie Tyler</cd : artist >
  <cd : country>UK</cd : country>
  <cd : company>CBS Records</cd : company>
  <cd : price >9.90</cd : price>
  <cd : year >1988</cd : year> </
rdf:Description>



</rdf:RDF>
```

N-triple

Questo linguaggio invece è molto più semplice dato che è stato creato appositamente per RDF.
Esempio:

Come si **integra RDF nel web**? XHTML2 mette a disposizione due nuovi attributi: about e property.

- about soggetto
- property verbo
- pagina complemento oggetto

Usando RDF, quindi usando grafi per la rappresentazione dell'informazione, funziona molto bene l'aggregazione. Inoltre funziona bene anche:

- reificazione: mettere le virgolette per parlare a più livelli
- contenitori: aggiungere le congiunzioni per gli oggetti
- variabili: un oggetto non specificato anonimo

Una proprietà importante è la monotonicità, cioè prendete dell'informazione espressa in RDF, e supponente che sia vera, allora, ogni passo di questa informazione è vero.
RDF definisce i mattoni di base ma serve di più. Classificazione degli oggetti, quindi l'uso di classi.

Più struttura permette maggiore integrità e deduzioni(ragionamenti), infatti per esempio consideriamo la classe vini rossi che contiene l'oggetto Cannonau, si può dedurre che Cannonau è un vino rosso, anche se esplicitamente non è scritto.

8.1.2 RDF-schema

RDF-schema è lo standard che permette il supporto di ontologie, formato come elementi di base da:

- class
- rdfs: subclassof
- individual: concetto di individuo

Inoltre si può definire la struttura informativa come DTD fa con XML.
Proprietà degli archi definite sempre dallo schema:

- RDF: property
- RDFS: subpropertyof
- RDFS: domain tipo di relazione che delinea il verbo
- RDFS: range

RDFS è il namespace di RDF.

Esempio
Property = mangiare
Subproperty = agire
Domain = animali
Range = cibi

Con i strumenti sopra elencati possiamo dare informazione e catalogare, ma basta XML schema per avere uno strumento completo a tutte le esigenze? No è solo l'antipasto, chiamato tassonomico.
Si può notare che RDF funziona meglio se sviluppato in oper-world, invece XML funziona meglio se in close-world. RDF inoltre posta a 0 la complessità di aggregazione delle informazioni.

8.2 URI e Nomi

Il nome **lato social** impatta il 10-20% di gradimento complessivo, è necessario scegliere un nome che funzioni.

1. I nomi **corti** funzionano meglio
2. Il nome dovrebbe essere **unico** e non confondersi con altri. Non prendere nemmeno un plurale se il singolare è occupato.
3. Prendere il **.com**, impatto del 4,5%.
4. Deve essere **facile** da scrivere e da memorizzare.
5. Non inventare parole. Al massimo unire parole vecchie. (1,5% ; -4,5%).
6. Attenti al suono, deve **suonare bene**. Un sito che inizia con le vocali guadagna il 3,7%, [r j y w] 2,9%, [f v s z] 3,3%, [p k t] 1,9%. Suoni che sono associati a parole brutte creano un danno -44%. In altri contesti, ad esempio un pubblico adulto, genera vantaggi fino al 7%.
7. Niente trattini -3%.
8. Usare i **numeri** 8.2%
0. Attenzione a come controllare se un nome è libero o no perché potrebbero rubarlo. Suggerimento internic.

Gli **URI** sono i nomi nel web **lato tecnico**.
Le connessioni dei grafi sono rappresentate come indirizzi web, infatti i nomi nel web sono proprio gli indirizzi web stessi. Si fa molta confusione tra URL e URI.
URI(Uniform Resource Identifier): sono i nomi con cui si identificano le risorse nel web (è un sovrainsieme degli indirizzi web), sono stati creati nel 1998 nella normativa RFC2396, l'ideatore è Tim Berners-Lee. Nel documento sono contenuti degli assiomi da rispettare:

- **assioma 0**: universality 1: ad ogni risorsa e ovunque si può assegnare un URI

- **assioma 0a**: universality 2: ogni risorsa nel mondo web che ha importanza deve possedere un URI

Per esempio www.libero.it non è un sito web ma il browser lo interpreta e lo traduce.
Esempio di URI:
http://www.libero.it
news:it.cultura
Tel:+358-123432426
Gli URI sono gli identificatori del web, si dividono in URL e URN.

URL: Uniform Resource Locator, sono URI che identificano la risorsa tramite una rappresentazione del loro meccanismo primario di accesso, dice come si trova una risorsa.
URN: Uniform Resource Name, è il nome vero e proprio, resta univoco e permanente anche quando la risorsa non è disponibile o viene eliminata.
Gli URI possono essere assoluti o relativi, sono formati: Schema:parte-dipendente-dallo-schema.
Lo schema definisce la semantica ad esempio:
Schema: http://
Parte-dip: corsi.math.unipd.it/tecweb2
Gli URI inoltre possono essere Gerarchici o Opachi:

Gerarchico: schema://authority/path?query
Riprendendo l'esempio di prima:
Authority: corsi.math.unipd.it, indica che il resto dell'URI è sotto il controllo di un'autorità.
Path: /tecweb (formato da 0 a più segmenti) non sono directory ma separatori gerarchici.
Query: informazione interpretata dalla risorsa usata tipicamente per interrogare la base dati.

Opaco: schema:opaque-part
Esempio:
Schema: mailto:
Opaque-part: massimo@W3.org, come path ma senza '/'.

URN, usa la seguente sintassi: urn:NID:… (NID è un namespace simile allo schema). Ad esempio gli ISBN: urn:ISBN:0-123-45678-89
La nuova zelanda ha acquistato un suo URN(URN::=ZLN). IETF, è l'organo che registra tutti gli URN di tutto il mondo e ne tiene traccia.
Inoltre negli URN ci sono caratteri consentiti, caratteri non consentiti, caratteri non consigliati, usabili con il corrispondente %ASCII.
Esempio:
http://www.unipd.it/a/b/c/d
http://www.unipd.it/a/b/c/%2Fd
Anche se L'ASCII corrispondente è d non sono due indirizzi uguali, ma vengono considerati due URI diversi.
Si deve prestare molta attenzione per esempio ad usare le lettere accentate che sono considerate un carattere escape, quindi si devono rappresentare con ASCII. Molto spesso gli URI sono creati e gestiti male e creano molti problemi ai motori di ricerca, dato che confrontano gli indirizzi delle varie pagine per capire se sono dello stesso URI, molto spesso essendo scritti in maniera non identica(key sensitive o uso di accenti), causano un riconoscimento sbagliato.

URI **variant problem**: molti URI rappresentano lo stesso concetto, l'utilità degli URI decresce esponenzialmente con il numero di varianti. Con gli assiomi sono nati dei problemi molto seri, vari nomi per ogni livello.
URI **variant law**: l'utilità degli URI decresce esponenzialmente con il numero di varianti.

Assioma 1: Global scope: Non importa a dove assegni un nome web ma avrà lo stesso significato (problemi di URI **Variant Meanign**).

Per evitare tutti questi problemi è stato inventato OWL.

8.3 OWL

OWL(Web Ontology Language Overview), aggiunge altre funzionalità per una maggiore informazione. In OWL c'è la gestione di uguaglianza o disuguaglianza tra oggetti. Permette di migliorare la piaga dell'URI Varian Law.
Equivalenze:

- equivalentClass
- equivalentProperty
- sameAs
- differentFrom
- AllDifferent
- distinctMembers

per esempio specifica maggiori caratteristiche ai verbi:

- inverseOf
- TransitiveProperty
- SymmetricProperty
- FunctionalProperty
- InverseFunctionalProperty

restrizioni sui tipi:

- allValuesFrom
- someValuesFrom
- minCardinality (only 0 or 1)
- maxCardinality (only 0 or 1)
- cardinality (only 0 or 1)

Prolog: è una interfaccia PNL(Pseudo Natural Language) per visualizzare le potenzialità di questa tecnologia. Questo sistema è molto più potente di OWL.

Tramite relazioni stiamo definendo una logica, ma poi dobbiamo essere in grado di fare calcoli automatici e quindi la logica va resa eseguibile. MA già la logica di primo ordine non è decidibile.

8.4 SPARQL

Ognuno può inserire una frase nel web ma il rischio è l'aumento eccessivo della complessità. Si rischia che una ricerca non termini mai, quindi non visualizzi i risultati! Si attua una scelta diversa, si decide che la ricerca sia decidibile, come per esempio SQL venne creato proprio per avere un linguaggio decidibile su una grande mole di dati.
SPARQL è l'equivalente dell'SQL per il web semantico, quindi per RDF e non per OWL.
Il mattone fondamentale è il graph pattern matching: *?x verbo oggetto.*
Le query sono molto simili alle SQL classiche, utilizza però vari tipi a seconda del contesto, per esempio nel web sociale utilizza FOAF(Friend of a friend) o utilizza le classiche triple di RDF.

DC –Dublin Core: è lo standard per descrivere le proprietà di base dei documetni ed è formato da 15 elementi informativi (titolo, oggetto, ecc…).

FOAF – Friend Of A Friend
Modella info sociali. Al primo posto c'è la persona:

- Nome, ecc…
- myersBriggs -> classificazione in 16 tipi di personalità.

```
<foaf:Person>
   <foaf:name> Nome Cognome </foaf:nome>
</foaf:Persona>
```

```
PREFIX foaf:<http://xmlns.com/foaf>
SELECT ?url
FROM <http://planetdf/blog.rdf>
WHERE { ?contributer foaf:name="Mario Rossi". }
```

SPARQL permette inoltre la gestione di dati opzionali usando la key OPTIONAL (ad esempio l'avatar):

```
WHERE{
  OPTIONAL{…}.
}
```

SPARQL è decidibile con complessità PSPACE (come sql).

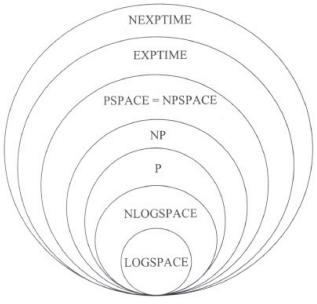

Figura 12: Diagramma delle complessità

Come si comporta con OWL?
Fissiamo una logica decidibile e sacrifichiamo l'espressività o diamo una logica molto espressiva e sacrifichiamo la computabilità? Sono state attuate **entrambe**.

Si è deciso di dividere in 3 OWL:

- OWL Lite: molte restrizioni, è decidibile e rispetta una logica leggera (SHIF), ha una complessità EXPTIME (esponenziale).
- OWL DL: è decidibile e rispetta una logica(SHOIN), è meno flessibile, possiede alcune restrizioni e ha una complessità NEXPTIME.
- OWL Full: non ha restrizioni ma può non essere decidibile.

Le classi di complessità in verità sono il caso peggiore, quello che conta realmente è la complessità media, i tempi di risposta dei motori di ricerca sformano esponenzialmente i tempi massimi, per questo avviene troncamento dei risultati. SPARQL ha una complessità PSPACE, quindi complessità bassa, ma se togliamo la possibilità di opzioni, si scende alla complessità CO-NP. La maggior parte del web usa logiche con complessità media bassa, ci sono rari picchi in PSPACE.
Se si pone una domanda(per esempio X è vera?), la risposta è formata da una prova di essa e la sua dimostrazione. Tra chi pone la domanda (noi) e chi elabora la risposta (google, web,...) esistono delle complessità: noi abbiamo una complessità polinomiale, cioè non siamo infallibili ma possiamo però lanciare un dato.
Con questo sistema possiamo risolvere tutti i problemi NP usando un sistema più potente di noi. (sistema Arthur Merlin).

8.5 IP (Non trattato)

Se al posto di ricevere una risposta ad una domanda ad un motore di ricerca, avvenisse un dialogo tra i due? Si potrebbe interagire con un sistema intelligente. IP(Interactive proof system), questo sistema aumenta di molto la potenza computazionale e si passa da una complessità NP a PSPACE, se invece di un sistema intelligente se ne usano due in parallelo la potenza aumenta ancora e si passa da PSPACE a NEXPTIME!
Il prezzo per questa tecnologia è:

- non è sempre garantita la risposta al 100%
- il tempo di risposta è maggiore

Il vantaggio principale è l'elaborazione di risposte molto più precise. Quelle citate fino ad ora sono logiche precise:

- OWA open world association
- CWA closed world association

Il mondo informatico che usiamo normalmente è tutto CWA(programmazione, database,...), se non si trova qualche cosa, significa che non c'è, questa filosofia domina da anni, usa un gestore unico.
Invece OWA implica che la mancanza di conoscenza non porti certamente ad una falsità! Usa logiche diverse dalla logica classica. OWL è codificato in tripplete e ricaviamo informazione a seconda "degli occhiali che usiamo".
Altri modi di trattare il web:

8.6 XQUERY (Non trattato)

E' più potente di XSLT, la sua potenza è come quella tra SPARQL e RDF, inoltre è funzionale e strettamente tipato.
RDF è XML, quindi possiamo usare XML per parlare dei dati e possiamo addirittura vedere i dati di tutte le pagine web, questo comporta che il nostro database è il web!
Xquery da la possibilità ad una sintassi multipla:

- SQL: sintassi ibrida(SQL like) non è XML
- XQUERYX sintassi pura in XML

Il cuore di XQUERY è FLWOR:(For, Get, Where, Order by, Return.
Come si costruiscono le query:

- almeno uno tra FOR e LET, per il caricamento dei dati, anche multipli e combinati tra di loro
- Where e Order by sono opzionali
- return è obbligatorio

Esempio

```
for $v in //video return $v
```

ritorna tutti i video del DB
Con xpath invece:

```
//video[years=1999]/title
```

Esistono i cicli e tutte le sequenze sono piatte, quindi è uguale scrivere (1,(2,3),5,4) e (1,2,3,5,4). Altri comandi disponibili sono CONT e AVG.
XQUERY è dichiarativo e funzionale, quindi una funzione può essere ricombinata usando altre funzioni.
si può prendere l'esempio precedente e:

```
avg( for $v in //video return $v)
```

ed in Xpath:

```
avg(//video[years=1999]/title)
```

XSLT non è funzionale, quindi non componibile!
Inoltre ogni sequenza si può vedere come un array e quindi si può utilizzare l'operatore [ ] per selezionare parti di essa!
Non sempre le query scritte nei due modi possibili(XPATH e XML) sono identiche per risultati, infatti, XPATH elimina i duplicati, invece XML no!!

FLWOR: LET
Dichiara una variabile e gli assegna un valore definitivo, infatti in un sistema dichiarativo ogni variabile è una costante che non può più cambiare il suo valore.

```
let   $numvideo:=5
```

Esempio sbagliato di uso di una variabile:

```
let   $i:=$i+1
```

NON è possibile!
Questa proprietà è detta referential transparency, questa proprietà è importantissima, infatti:

```
let   $x:= f (x)+f (x)
```

se su un sistema normale questa istruzione non può essere semplificata perché non conosco se i valori d'uscita delle due funzioni sono identici, invece in un sistema dichiarativo dato che x è sicuramente uguale in entrambe posso sostituire con:

```
let   $x:=2□ f (x)
```

Per calcolare per esempio la durata di tutti i video del database:

```
let $sum:=sum(// video/runtime )
```

WHERE
Filtra i dati, esempio:

```
for   $genre  in   //genre/choice
for   $video  in   //video
for   $actorRefs  in
$video/actorRef
for   $actor  in   //actor
where   $video/genre   = $genre
and   $actor/@id =   $actorRefs
return   concat($genre ,   "|   ",$actor)
```

soluzione ibrida:

```
for $genre in //genre/choice
for $video in //video[genre=$genre]
for $actorRefs in $video/actorRef
for $actor in //actor [ @id=$actorRefs ]
return concat($genre ,": ",$actor)
```

ORDER BY
In SQL standard se non impongo un ordinamento, l'output ha un ordine, invece in XQUERY l'ordine segue i cicli for annidati. L'esecuzione è libera, solo durante la stampa si allinea all'ordine imposto dai cicli for:

```
for $x in //video
order by $x/ years ascending
```

XML annidato dove le parentesi annidate attivano XQUERY e non viene considerato come testo:

```
for $v in //video [ genre="comedy"]
return
<actor video ="{$v/ t i t l e }">
{// actor [ @id= $v/actorRef ]}
</ a c t o r s >
```

DOC(URI)
Per identificare i dati da usare in XQUERY si usa il comando DOC(URI) che riceve in ingresso un URI. DOC può identificare i dati da qualsiasi posto nella pagina, però identifica i dati da un'unica grande sorgente che può essere virtuale, composta da più fonti fisiche, tramite il comando COLLECTION.

COLLECTION(URI)
Viene usato per reperire i vari documenti fisici.
Il prezzo pagato è implementation-dependent, cioè dipende dal tipo di implementazione!

Commenti
(:Commento:)

Si può inoltre utilizzare XQUERY su RDF emulando SPARQL
XQUERY utilizza due caratteristiche un pò diverse dalla normale implementazione SQL:

- sequenze piatte
- rotta la retrocompatibilità per XPATH (eliminazione dei duplicati)

Per capire queste scelte, bisogna fare una premessa, il web è grande, molto molto grande. Problemi non comuni con la normale programmazione, per la

grande mole di dati, quindi non salvabili in un'unica macchina. Se è impossibile salvarli, si rende impossibile anche elaborarli. La soluzione adottata è di utilizzare più macchine in parallelo, applicando un sistema distribuito, questo causa un cambio della logica classica di programmazione. La soluzione più usata e più sviluppata è HADOOP, viene usata da tutti i maggiori gestori di dati (Google, Facebook, Twitter).

8.7 HADOOP (Non trattato)

HDFS(HADOOP Distributed File System) è la componente di base, impone una taglia massima ai dati dividendoli in blocchi da 64 MB a 128 MB. Inoltre è fault tollerant che può causare errori di lettura scrittura, per questo si usa una tecnica simile al RAID, con una mole molto maggiore di dati. Si è scelto di usare una politica di performance, a scapito dello spazio. Molto simile a RAID 1, ogni dato viene copiato 3 volte, però il sistema è reso intelligente ed a conoscenza di una porzione di sistema intorno a se. Usa il concetto di Rack, cioè ogni gruppo di macchine è un Rack, i dati vengono suddivisi in 2 copie nello stesso Rack e una in un'altro.
Vantaggi:

- se un Rack ha dei problemi non vengono persi i dati, perchè c'è sempre una copia esterna
- se una copia dei dati funziona male, si può utilizzare la copia nello stesso rack con notevoli vantaggi di velocità
- si può leggere e scrivere i dati in parallelo aumentando la velocità

Questa complessa architettura(di tipo WEB3) viene gestita in maniera centralizzata, tramite il master node. Questa architettura ha un unico tallone di Achille, se la macchina del master node cade, cade tutto il sistema. Per questo negli ultimi sistemi, viene creata una macchina di backup del master node.

HADOOP agisce sul dato con map reduce: è tecnica funzionale e parallela, l'ordine delle operazioni non conta, obbliga la programmazione già predisposta al parallelismo.
Flusso tipico:

- leggo i dati
- MAP(estraggo quello che mi serve)
- shu□e and sort
- reduce(aggrego, sintetizzo e filtro)

La potenza del seguente metodo è anche se diverso dalla programmazione usuale, touring completa.
Il dato

Ogni singolo dato è formato da una coppia chiave, valore.
La trasformazione dei dati avviene come segue:
Elabora: map(chiave,valore) crea lista(nuova chiave,valore intermedio)
reduce:(chiave, lista valori intermedi) crea lista valori
Quando avviene l'elaborazione map può essere anche ripetuta e ricorsiva.
In codice:

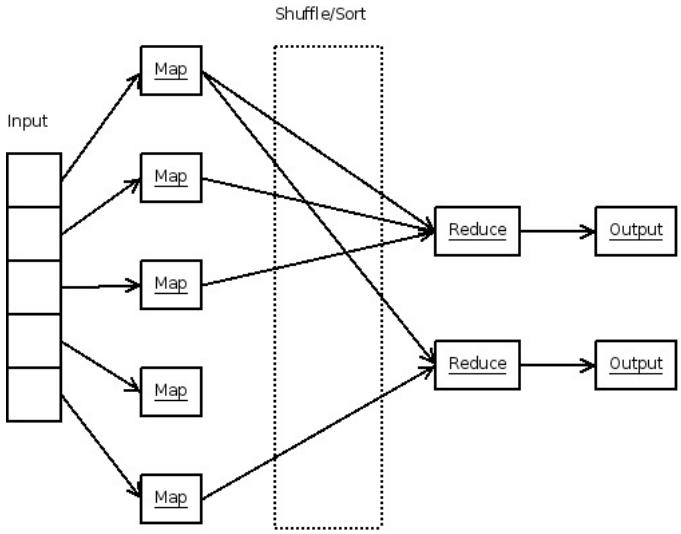

Figura 13: Schema di map reduce

```
let map(k ,v)= emit(k. toUpper () , v. toUpper ()) la
```

stampa è:

```
(”luca      rossi”,”senior”)
```

stampa
(”LUCA ROSSI” ,”SENIOR”)

Parallelismo
Si eseguono tutti i map in parallelo
stop, si aspetta che finiscano tutti i map si
riparte in parallelo con la reduce
Tutto basato su data locatily: i map sono proferenzialmente su macchine vicine, per minimizzare il tra□co di rete che crea molto spesso colli di bottiglia.
Contare quanto una parola è nel database:

```
Map( key , v a l u e )
```

```
document
for  eath word w in  value:
            risultato=0
            for  eath cont v in  values:
                        risultato+=v
emit(key,risultato)
```

Come spegato precedentemente tutte le elaborazione sono fault tollerant, se per qualche motivo una map non riesce a produrre un risultato in un tempo utile, viene buttata e rilanciata. se dopo un certo numero di tentativi non funziona viene eliminato (best efford).
Queste appena descritte sono il nucleo di HADOOP. Degli applicativi che si basano su di essi sono:

8.7.1 FLUME(e FLUME NG) (Non trattato)

è un sistema per il trasporto e l'aggregazione di big-data in real time(Weblog,...) e si basa su flussi di dati. Ogni flusso può avere caratteristiche molto diverse, che viene successivamente aggregato in nodi, ogni nodo ha un sorgente e una uscita.
Esempio di inserimento dati:

```
source  tail ("/var/log/HTTPD/Access log ")
```

Prende il log del server web e ne invia l'ultimo pezzo al nodo.
Il controllo di FLUME è anch'esso centralizzato con un master. Parla con le map come con i cellulari. Il controllo esterno avviene mediante API gestita da thrift I dati realtime arrivano dalle varie fonti a intervalli a seconda delle esigenze.
Utilità di questo sistema:

- analisi commerciali
- personalizzazione realtime
- trends
- monitoraggio del carico e delle performance
- controllo di errori

Usa anch'esso il fault tollerant, se avviene qualche problema si attiva il buffering e se eccessivo si usa il load shedding, cioè i vecchi dati vengono eliminati(milk).

8.8 HBASE (Non trattato)

Fa parte anch'esso del tipo no-sql, l'evoluzione del concetto di base di dati viene sviluppato sulla base dei big data. Il tipo no-sql sono soluzioni distribuite e fault-tollerant e sopratutto non sono ACID. ACID significa:

- Atomicity: ogni transizione è atomica, quindi quando una transizione si attiva deve finire e non può essere interrotta
- Consistency: ogni transizione lascia il DB consistente, cioè le modifiche fatte vengono subito aggiornate
- Isolation: più transizioni non possono interferire tra di loro
- Durability: quando completo una transizione dura nel tempo

Con i DB no-sql, quindi non usando ACID si ha molta più flessibilità. Questa nuova tecnologia viene usata dai più grandi DB, come per esempio Google, Facebook, Amazon, ecc... Essi inoltre usano i dati cacheati, quindi le persone vedono i dati non in realtime ma di qualche minuto o ora prima, a seconda dei tempi di aggiornamento delle basi di dati. Amazon per esempio quando due utenti provano ad acquistare l'ultimo libro, permette l'acquisto ad entrambi! Poi cercherà di far arrivare il prima possibile il riodine del libro!

8.8.1 Teorema di Brewer

Non esiste il pasto gratis: se vogliamo Consistency, Availability e fault tolerance si deve rinunciare ad almeno una di esse. Impossibile quindi riportare la vecchia generazione di DB nella nuova. L'alternativa ad ACID è BASE!
Basic Availability: il sistema è disponibile, ma non sempre.
Soft State: l'informazione può morire, se non viene costantemente rinfrescata piò ritornare alla forma di partenza.
Eventual Consistency: Se per un certo timer il DB non viene letto o modficato, il sistema si aggiorna! Ma non sempre!

HBASE è una base di dati distribuita open source modellata su BigTable di Google e scritta in Java. Fu sviluppato come parte del progetto Hadoop. La struttura dati infatti è quella di map reduce, quindi formata da coppie chiave-valore, ordinate per chiave.
Una tabella è una collezione di righe-colonne:
una riga contiene una coppia chiave valore ogni
chiave è primaria e unica
ogni valore di riga contiene n colonne
le colonne possono essere compresse o definite per stare in memoria il
valore di ogni colonna è definito cella
famiglie di colonne: le colonne sono raggruppate in famiglie ed esse si dichiarano alla creazione della tabella. Le colonne e le famiglie stanno in un

unico file chiamato HFILE.
Il nome delle colonne è detto qualificatore.

Uno dei problemi del vecchio DB sql era l'obbligo di dichiarazione dei valori NULL. Il nuovo tipo di DB può essere visto come il Web che si adatta al suo contenuto. Il confronto con il vecchio DB è simile alle differenze tra matrice e grafi. Nel HBASE semplicemente i valori null non ci sono e non vengono scritti! Dato che scriviamo le relazione e non i dati in una tabella.

Ogni cella ha una parte di versione, aggiornata automaticamente dal sistema. Il sistema fa versioning automatico! Ogni dato viene inserito inoltre un time step. Il cambiamento di un dato non avviene, per i vari problemi distribuiti. Un dato quindi non si cancella, ma se aggiorno un dato si aggiunge solo che ho un nuovo dato in quella cella con il time step aggiornato. Il vecchio dato resta sempre in memoria, finchè non viene cancellato dopo un certo arco temporale.
Come si accede ai dati:
(table, rowkey, family, column) = value
in generale: (table, rowkey, family, column,timestamp) = value

A livello di struttura dati:
sortedArray(Rowkey,list(sortedArray(column, list(value,timestamp))))

Consistenza: l'accesso ad una singola riga è atomico, invece l'accesso a più righe contemporaneamente non ha nessuna garanzia.
Quando il sistem cresce sia auto adatta, il sistema all'avvio inserisce tutte le righe in una regione, se la regione cresce troppo il sistema fa split, cioè taglia la regione in due parti inserendo ogni parte in una nuova regione. La stessa cosa, ma al contrario, se le regioni diventano troppo piccole, le unisce. Le operazioni avvengono usando WAL(Write AheadLog), accumula tutte le operazione e quando può inizia a farle, per cui questo è un punto di fallimento, se crolla il WAL crolla tutto.
Le tabelle vengono "disegnate" alte e strette, perchè per fare split usa le chiavi delle righe. Per non avere problemi di crescita eccessiva di una riga usano righe piccole contenenti solo un singolo messaggio, anche se molto più numerose, ma non possono causare problemi di lunghezza. Gli split fuzionano bene.
Il passaggio ai big data cambia anche le logiche algoritmiche, si stà passando all'uso di algoritmi probabilistici, per esempio una struttura probabilistica viene usata per sapere se esiste una certa chiave nel DB. Se c'è la chiave la trova, ma può sbagliare!! I vantaggi però sono maggiori della probabilità di sbagliare, ha una costante di lettura e scrittura, inoltre ha un bisogno di spazio molto più basso dei classici algoritmi.

8.9 Linked data

Linked data descrivono un metodo di pubblicazione di dati strutturati in modo che possono essere interconnessi e più utili rispetto alle query del web semantico. Con linked data, in altre parole, ci si riferisce a dati pubblicati sul web in una modalità leggibile e interpretabile da una macchina, il cui significato sia esplicitamente definito tramite una stringa costituita da parole e marcatori. Si costruisce così un reticolo di dati collegati (linked data, appunto) appartenenti a un dominio (che costituisce il contesto di partenza), collegato a sua volta ad altri set di dati esterni, ovvero fuori dal dominio, in un contesto di relazioni sempre più estese.

LOD – Linked Open Data

Sono un tipo particolare di open data e raccoglie i dataset open messi a disposizione sulla rete. Seguono una classificazione da 1 a 5 stelle:

1) dati disponibili sul web con licenza open.

2) dati disponibili sul web in un formato machine readable.

3) come sopra, usando un formato dati non proprietario.

4) come sopra, formato semantic web.

5) come sopra, con dati linkati ai dati di altri per dare il contesto.

Per fare mashup e unire il mondo xml, xhtml e html con il web semantico sono necessari strumenti come il **lifting** (passaggio da web non semantico a RDF) e il **lowering** (l'altro verso).

Da sql a RDF: d2rq. Ci si collega al database tramite jdbc e si convertono i dati. Funziona anche da server, fornendo subito accesso via web ai dati visti come RDF. Un comportamento simile lo fa Triplify.

Da pagine web a RDF: Tecniche nlp, natural language processing, analizziamo l'informazione testuale o ipertestuale e fanno automaticamente il lifting. Esempi: Open calais, spotlight, alchemy, ecc.

Per ottenere i dati bisogna accedere agli endpoints, ciò avviene tramite protocollo SPARQL. Di solito si fa get http di sparql query.
Query: la query sparql che viene eseguita modificata facendo il classico %encoding.

DBPedia: Wikipedia in versione semantica. Ha una sua ontologia. Fornisce degli endpoints per il pubblico. Si appoggia alle ontologie standard.

Schema.org definisce le ontologie più importanti per i dati web.

Esempi di LOD:

- linkedmdb.org
- datahub.io
- publicdata.eu
- it.dbpedia.org
- dati.gov.it

Per trovare i LOD si usa CKAN che è il catalogo di riferimento per i LOD o Swoogle (motore di ricerca per il web semantico).

Graph Of Thing: è un tool che ci permette di creare, manipolare e studiare grafi. Questi grafi sono oggetti che descrivono la relaziono tra vari gruppi.
Google ha pieno supporto per i dati semantici.

- Comprende RDF;
- Supporta tutte le antologie Schema.org;
- Capisce ed incorpora tutta la semantica che usa internamente;
- Espone alcune delle proprietà del **grafo della conoscenza** (le info mostrate a destra durante una ricerca -ora anche tra la lista delle ricerche-).

Ad esempio i **Rich Snippests**, le informazioni aggiuntive nel risultato della ricerca grazie all'utilizzo di informazioni semantiche.
I tipi supportati dai Rich Snippests:

- Autori: nome, foto (chi aveva la foto aveva anche più link quindi c'era uno sfasamento della top10) e altre varie info;
- Business & Organizzazioni: ad esempio posizione nella mappa;
- Eventi;
- Musica;
- Persone;
- Prodotti: rating, ad esempion, di Amazon;
- Ricette;
- Recensioni: info e rating ad esempio con Tripadvisor;
- Video: immaginina del video, durata, ecc.

Grazie al web semantico e al grafo della conoscenza è possibile dare direttamente la risposta alle domande dell'utente (oltre ai vari link che contengono la risposta).
Anche l'esistenza di concetti stessi sono ottenuti con il grafo della conoscenza. Stessa cosa vale per il raffinamento delle info e per le ricerche correlate. Ne consegue che un utente passa più **tempo** su Google.
LOD di Google:

- Google.com/publicdata (in modo piccolo)
- Freebase (in modo grande): è il super contenitore della base semantica che Google usa.

9 Mobile e Social

9.1 Mobile

Le app non sono web ma sono intercorrelate. Il 52% dei top 1000 siti non hanno un sito mobile e il 25% sfora lo schema. Tutti hanno problemi di usabilità.
L'evoluzione è quella di arrivare alla convergenza tra desktop e mobile. Steve jobs voleva le web applications e non app. Ma per un motivo economico si è preferito non farlo.
Con HTML5 si è arrivati a hybrid app favorendo tale convergenza.

Le **app** minimizzano il tempo di accesso al servizio che mi interessa.
¼ degli utenti usano le app più di 60 volte al giorno. La fascia di età che utilizza meno le app è quella tra i 25 e 35 anni.
L'utente medio usa per l'86% app e per il 14% il web.
Cosa fanno gli utenti con le app:

- Giocano 32%
- Siti sociali 28% (17% da Facebook)

A parti i giochi, molto del resto è solo uso del web tramite apps.
Le app vanno di moda, perciò è lecito che molti sviluppatori si dedichino a queste. Di conseguenza c'è molta competizione.

Il 26% delle app sono aperte una sola volta, il 13% 2 volte, il 9% 3 volte e via così. Hanno inoltre una vita media bassissima: dai 4 mesi a 1 anno. I videogiochi hanno paradossalmente una vita media di 4 mesi.
Se la crescita dura più di tre mesi, allora molto probabilmente vivrà molto più a lungo, altrimenti morirà a breve.
Proprio per questo resta il classico problema di essere trovati.
Come si viene trovati? Primariamente, lo **store**, cioè un motore di ricerca. Ma cambiano le regole per essere nella top 10.

ASO – App Search Optimization
È il SEO delle app. funziona sempre a parole chiave che vanno inserite nell'app:

- Dentro la descrizione dell'app;
- Eventuali keywords;
- Il nome stesso dell'app (fattore più importante).

Per il resto, i motori Google e Apple usano tutta l'informazione data dal SIS, il sistema sociale complessivo. Downloads, tempo di utilizzo, ratings e reviews, unistalls, brand (eredità giudizi successo app delle stessa società), e le parti positive e vegeti è che vengono dagli altri SIS (web ed email).

9.2 Usabilità Mobile

Queste regole sono valide sia per app che per web.
Google's mobile compatibilty test.
Webmaster spam: paura, incertezza, dubbio, se non si segue queste regole si scende dal ranking.
Tre componenti di base del mobile:

1. Essere mobile
2. Taglia dello schermo
3. Modalità di iterazione: dita. (Nel mondo esistono ancora cellulari non touch).

Facebook: target mondiale, tre versioni mobile. m.facebook.com touch.facebook.com 0.facebook.com

1. Essere MOBILE
Il problema principale è la **rete**. Si passa al 3G e sono il 40% più lente di quelle desktop. Le reti 4G sono in media il 12% più lente di quelle desktop. Quindi ogni sito paga un corrisponde dazio in termini di velocità ed aumento di timer. Non è solo una questione di timer di sessione o globali, questo è un ritardo locale per ogni pagina. E quindi succede che l'utente lo confronta con i tempi medi propri. Nel caso desktop il tempo massimo di attesa è di 2 secondi. Se sfora si percepisce una sensazione di ritardo. Nel caso mobile il **timer resta invariato**. Bisogna perciò stare attenti a non sforare questi limiti. Gli stessi limiti si applicano alle app. È essenziale essere responsiveness!
Soluzioni:
Mettere una **progress bar** o un cosiddetto spinner (nel caso desktop) provoca frustrazione all'utente. L'utente percepisce il periodo di tempo in modo più lungo. Non andrebbe fatto. Perciò, si usano altre tecniche.

- **Transitioning**, tenere l'utente impegnato con animazioni o altro.
- Caso particolare di transitioning, **skeleton screen**. Se so il layout finale dell'azione, posso cominciare a disegnarlo anche se non ho ancora i dati. Primo Instagram.

- **Preemptivenes**: cerco di prevenire alcune azioni. Esempio app che carica foto, invece di far scegliere una foto, chiedere una descrizione e poi fare l'upload, carico la foto appena è stata scelta.
- 0.facebook.com versione a super banda ridotta. Velocissima ed usa solamente testo, però cambia la navigazione del sito, rendendo quindi l'utente da un lato contento per la velocità ma dall'altro scontento. La versione 0 viene offerta gratis in tutti quei posti del mondo dove le connessioni sono lente.

2. Taglia dello schermo

Una pagina normale farò fatica ad essere visualizzata senza scroll.
Quanto grave è lo scroll? Lo scroll orizzontale è molto peggiore di quello verticale. Lo scroll verticale spesso non è così grave. Lo sforzo fisico e mentale è minore perché si usano le **gesture**. È però deleterio per gli utenti quando viene usato per offrire scelte (l'utente deve tenera mente la parte nascosta). Minimizzare lo scroll verticale e quindi ad esempio evitare le immagini. Si giustificano nelle liste di scelta solo quando si trattano di prodotti finali. Non funziona neanche l'effetto: testo e immagini piccole, crea l'effetto vedo non vedo.
È possibile usare icone anche se l'utente preferisce ancora il testo.
Se però si vogliono utilizzare senza testo vicino, occorre rispettare due principi:

- Explainability: tenendo premuta l'icona si ottiene l'informazione testuale su alla sua azione.
- Escapability: poter comunque evitare l'azione spostandomi fuori dell'area con il dito.

Inoltre avendo taglie piccole dello schermo devono essere ridefiniti certi elementi invasivi come la pubblicità.
IAB: interactive advertising bureau, ente che gestisce la dimensione dei banner.
Interstitial ads: prende tutto lo schermo ed ha un tasto per la chiusura.
Smart banners: la taglia si adatta in base all'ampiezza dello schermo. Sono i migliori però funzionano molto male se sono implementati in posizione fissa (crea limitazioni e disagio agli utenti). Ancora peggio se il banner cambia.
Lo Smart app banner, cioè pubblicizzare la versione app del sito, ha lo stesso effetto negativo dei popup, meglio scriverlo nel contenuto della pagina.

3. Mezzo di interazione: le dita

Il **drag** non crea sforzi muscolari, per questo facilita l'uso delle gestures, da qui deriva l'uso degli **swype**.
Si deve rispettare però l principio dell'**Untiming**, azioni derivanti da pressione non dovrebbero dipendere dalla durata, ma solo distingue il tap e il drag. Non rispettare l'untiming provoca stress.
Il problema principale è che le dita sono molto grandi (**fat finger**) e non riusciamo a cliccare in modo preciso (il dito medio è largo in media: 11mm, 8mm per i bambini e può arrivare fino a 19mm). Morale: un'area cliccabile deve essere molto grande. La **taglia minima** di un'area cliccabile dev'essere di 7x7 mm (massimo 5x5 a discapito di perdere il 20% di precisione). Ed attorno almeno una zona di **padding** di 2mm. L'ideale è 9x9. Più dell'83% dei siti top 1000 non rispetta queste regole.
Reversibility: ogni azione che si compie dovrebbe essere reversibile, tanto più in situazioni di potenziale errore dovuto.

La **legge di Fitts** cambia nel senso che la taglia dell'oggetto conta ma ora conta anche l'imprecisione del dito e la presa del dispositivo (ci sono 5 fasi e ognuna ha una termografia diversa).
Gran parte usano i pollici, il che peggiora la precisione, bisogna aumentare di 2 mm le taglie.
Su mobile la finestra coincide con il bordo del dispositivo quindi ci sono delle zone magiche come i bordi. Possiamo posizionarci i **fan menù**. I pie menù non sono invece il massimo (il dito copre alcune parti della torta).

9.3 Social Network

7-13 marzo 2010 Facebook supera Google.
Tre aspetti fondamentali:

- TIME
- TRUST
- TRANSMISSION

I social network hanno un **tempo** di assorbimento altissimo, Facebook da solo assorbe tutti gli altri social (FB 53%, Yahoo 17.2%, Google 12.5%). Un'informazione dentro un social è più efficace rispetto ad un sito web normale (**trust**). È più efficace di ordini di grandezza dai 10x ai 100x.
Astroturfing: creo commenti o informazioni sui social che sembrano vere ma in realtà non lo so. Nel 2013 il traffico bot ha superato quello umano.
La **viralità** è un mezzo potente ma ha bisogno delle giuste condizioni:

- Social Currency: rende chi la condivide speciale, gli dà una luce positiva;
- Practical Value: valore pratico associato al contenuto;
- Storytelling: creare curiosità con una storia che attiri;
- Emotion: un messaggio deve portare emozioni. Le emozioni positive sono più virali di quelle negative (FB non ha il tasto “non mi piace”).
    10) Eccitazione
    9) Affetto
    8) Speranza
    7) Gioia
    6) Piacere
    5) Delizia
    4) Felicità
    3) Sorpresa
    2) Interesse
    1) Divertimento