Technologies Web 2

1 First lesson: Introduction to the web

In 1945,
birth of**MEMEX**, is an analog computer equipped with a filing system, designed by the American scientist and technologist Vannevar Bush. It also made it possible to view the data entered.
The innovative fulcrum of MEMEX is that it manages to connect more information, so from the first you can get to the second, through a primitive link, for example from the Indian page you can get to the description page of the arch. It is the primitive concept of hyperlink.
In 1960,
Douglas Engelbart invents the on Line System (**NLS**), was a first computer consisting of a mouse, keyboard and a device for fast writing.
Ted Nelson, in the same years he invents the definition of hypertext and its applications, and also extends the definition of hypertext to hypermedia. In 1967 he launches**XANADU**, is the first draft of the Web where he applied hypertext, the basic concept was point and click. It was a very innovative project, it also included a form of micropayments for the licenses of other users' programs or tools, allowing you to expand your "site" quickly and easily, this technology was not successful and disappeared. The address scheme was more powerful than the current one and the links were equipped with BIVISIBILITY and BIFOLLOWABILITY. Project never built.
A very similar technology has reappeared in recent years with amazon, apple store, ...
In addition, XANADU allowed you to connect individual strings with a link, which is not yet possible nowadays. Finally it kept track of the various versions of the site so that it could never happen that a link was broken but was linked to the previous version if not present. The only problem is that he has never been able to fully realize it, it has always been an idea that has never been fully implemented! Another very large obstacle to its realization is that the idea had been patented and therefore no one could work on it freely, if it had been free maybe some other group

would attempt its implementation.

In**1980**,

Tim Berners-LeeI recycle all the previous ideas and freewheel he put ideas on paper. In 1989 he wrote a document: information management and proposal. In this document he describes the**first idea of the Web**, represented in the following image.

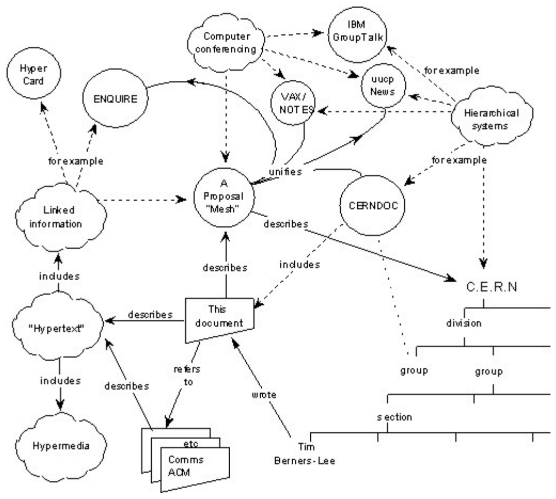

Figure 1: 1980 The first idea of the Web

In 1990 he received a grant from CERN in Geneva and was sent the best computer on the market, the Next (created by Steve Jobs).
After several attempts, he chooses the definitive name for his project: World Wide Web.
In addition, the browser is also created to use it, always called WorldWideWeb! This browser is a bit particular, in fact it allows not only navigation, it allows the editing of pages and includes a server side within the browser itself!
In 1991 at the international hyperlink conference he presented his project, but was refused, because it was defined as too simple!
The project included:
Archie: it is the first search engine in the world, but it does not work on the web, but via FTP.
WAIS: it is a search engine inside a text document (inside the databases scattered on the internet).
Gopher: it is a network protocol based on a server module that rationalizes information and manages it in a tree structure accessible to the client. The web pages loaded in Gopher could be of two types, either menu or text.
In 1992 it was invented**Veronica**, a search engine for Gopher. Given Gophie's steadily increasing popularity, Tim decides to make him pay.

chin. In a short time it lost much of its popularity and disappeared.

In 1993,
the WWW doubles its users, because it's free! The reference browser is **Mosaic**, compared to other browsers it displayed images within the page and not in a separate link. It was not specified in any standard that the images should be displayed on the page but sometimes it is the graphics that count and not the system!
The servers present in the WWW go from 50 to about 200!

In 1994,
the traffic managed by CERN increases by 2000%, it goes from 200 servers to 2200, the WWW has an exponential growth.
CERN decides not to finance it anymore !!! Then by popular demand the entire project was moved to MIT where it was founded**W3C**.
The WEB was born in Europe, but we gave it to the Americans !!
The browser of the moment becomes**Nascape**Navigator, allows you to view pages incrementally, the change of browser makes it clear that waiting time is fundamental.

2 Second lesson: website usability

A site must be considered as a house or a shop, you look at the window and then you decide whether to enter or not. The homepage can be seen as the shop window, the visitor wants information and therefore our home must provide a summary of everything we offer. The homepage is the heart of the site, it must be managed like a journalistic text, so it must respect the 6W:

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

When writing a site, the same rules apply, adapted a little to the context:

- Where: what kind of site I arrived at, the content
- Who: who represents the site
- Why: site benefits
- What: site offers
- When: latest news, the news of the site
- How: how to get to the main sections of the site

A user's main problem is time. The site therefore, in addition to respecting the 6W, must also be able to summarize the topics in a small number of words per page.
Users have expectations but also limited time. An average user stays on the homepage for about 31 seconds, so you have 31 seconds to convince a user to stay on the site they are viewing. Furthermore, it must be taken into account that an above average user can read 180 words per minute, so doing the calculations in 31 seconds can read about 93 words! Actually you have to take into account the time he wastes looking at the visual layout, so really the words he reads are even less!

Furthermore, another minimum requirement that must be set is that not only the user remains on the site, but that he returns to it. When he returns, the rules change, in fact the questions Who, Where and Why must no longer be considered, but the questions What, When and How must be answered. Furthermore, a visit after the first implies a greater need and less time available !!

Average length of stay:

- 2◦time: 25 seconds
- 3◦time: 22 seconds
- 4◦time: 19 seconds

From 4◦time on and on stabilizes. If we aim for a site that recalls rente means we have 19 seconds to answer the What, When and How! So about 54 words.

After the user enters the site, then leaves the homepage, the aces are no longer needed, because a "bond" is created which makes leaving less difficult! In fact, on the internal pages it dedicates an average of 51 seconds, therefore it allows more specific pages, however we must not exaggerate in the contents, you have about 150 words available. If the information exceeds, it must be divided into modules of the exact size and separated into several pages.

Another very important constraint is the**global time**to stay on a site, so that the user is satisfied, that is, he has found what he was looking for. There are two main categories:

2.1 Preliminary time

The time he takes to get an idea of the site and whether to stay or not is called choice time.
The average choice time for a generic site is**1:49 seconds**, if after this time the user is not satisfied he leaves the site even if the contents were there. So you manage to visit the homepage (31 sec), and two internal pages (53 sec each)! But if the user leaves the site what is the probability that he will return? In 88% a user never returns!
It is the most important time as it is the overall time to convince the user to stay with us.

2.2 Time to success

Or satisfaction time, is the time we have to satisfy the user. The user has decided to stay, so he wants the information he was looking for, the average time he takes before being dissatisfied is**3:49 seconds**. The balance between homepage, internal pages and paths to get to the information is very important. In fact, on the first access, the user visits, on average, the homepage and an internal page and then decides whether to stay! Looking at the site tree, a user who starts from the home in 3 clicks should get to the information he was looking for.
Unless...
In recent times, navigation has changed, once you always started from the home to browse a site and then entered the internal pages ...

Now, on the other hand, with search engines, most of the time you enter directly into an internal page containing the information you were looking for, skipping the home page. For users who arrive for the first time on a site and directly on an internal page, they mess up all the accounts made previously! This phenomenon is said**Deep Linking**("Deep connection"). Each page must therefore be a small homepage !! The 6W are valid again with some modifications: Optional axes:

- When: what kind of site I arrived at, the content
- Why: brief description of the benefits of the site
- How: a search at the top right is sufficient

Mandatory axes:

- Who: typically displays the logo on the top left and a brief description of who we are
- What: direct link to the home
- Where: the what is not enough because it obliges the user to make an extra link to understand which site he is on and therefore a waste of time.

It is also necessary to display the position, with respect to the home page or the path made by the user, on the site on each internal page.
The said technique is used**Breadcrumb**, which literally means bread crumbs and refers to the fairy tale of Hänsel and Gretel leaving a trail to find their way home!
There are 3 types of Breadcrumbs:

- **location**: shows we are based on the site hierarchy, giving a level structure: home> news> news;
- **attribute**: shows the attributes of the given page, so it does not necessarily correspond to the location, but contains the path from the main type and its subtypes (for example a component sales site may have HW / graphics cards / nvidia / geforce). Each page has tags that help understand the categorization;
- **path**: shows the physical path taken by the user to reach the page. It will be of a dynamic type to avoid problems deriving from the predetermined one (it is not certain that a user arrives at the page following the predetermined path).

The classic separators for Breadcrumbs are: either the major (>) or the slash (/).

3 Third lesson: usability of websites part 2

I can encounter two types of usability problems on a website:

- Persistent
- Not persistent

The first problem**persistent**is:**Lost in navigation**,
the user does not know in which position he is during navigation, therefore the Where axis is missing or missing. Not helping the user to remember where he has been increases the computational effort.
One solution to this problem, which was seen in the previous chapter, are breadcrumbs, even if they don't fully meet a user's expectations. In fact, they do not describe in detail the path taken by the user, a common solution is to color the links already visited in a different color than the links to be visited, allowing the user an immediate glance on the pages already viewed. If, on the other hand, I leave the user with the task of remembering where he has already been, it creates discontent and stress in the visitor, risking his abandonment.
The colored links were introduced by Nescape Navigator and are not a standard, so it is not written in any rule that the color change must be applied, but it is a good rule that it is recommended to respect to ensure greater usability.
74% of websites use link color change.

Another user need is the fast movement between pages, in fact to return to the previous page almost all users use the back button even if there is a return link. They minimize the computational effort even if to return, for example, to the home you need several clicks and instead there is a direct link to it! An average user is willing to click up to 7 times to go home instead of using the direct address key! They don't have to spend any effort searching for the return link! This common use is called backtracking, that is, the continuous visit of the links with return via back.

A second persistent problem is opening more than one browser window, rather than always using the same one. First of all, the use of the back button no longer works, moreover, it creates a lot of confusion for the average user. Two types of windows can be opened, either in the form of tabs or new windows proper. The worst choice is opening a new window, in fact the user may not know how to go back and if the window does not take up the whole screen, he leaves it open and clicks on the previous window, putting it in the background. If at a later time the user clicks on the same link as the background window, the browser will update it, showing no changes on the screen. This brings a lot of inconvenience to the user and therefore absolutely to be avoided!

Another problem are the pop-ups, which create navigation difficulties and disturb due to the unwanted appearance. They are new windows that appear without the user's permission.

Another problem is the non-compliance with conventions, such as the color of the links which are not mandatory standards but which help a lot navigation.

Other issues:

- Don't respect the what axis;
- Use of empty language or with little content;
- If the content is important then the form is also important.

3.1 Jakob's law

Users spend most of their time on other websites and not yours!

This law summarizes the concept of not creating sites that are too original, which can create problems in navigation and discovery of the site! But stick to conventions as much as possible.

4 Fourth, Fifth, Sixth Lessons: The Content ...

Problems concerning the content are of the type**not persistent**. An average user expects that the text contained in a site is rich in content, not a meaningless text such as "polychase". Another very important point is the form of the text, it must not be too complicated or too elementary. Finally, as explained above, the timers must always be respected. Some sites may not respect the timers, because maybe they are unique of the kind, for example Government sites.
Web text must be even more readable than other texts, for various screen reading reasons.

4.1 Web Text Rules

If a common text, on paper or other medium, we use 100 words, a web text to be of equal difficulty in reading, comprehension and learning speed must be 50 words. If it is to be understood by everyone, it must be 25 words compared to the initial 100.
A good way to write web text is to start from the conclusion. Non-persistent problems:

- **splash pages**: it is the initial welcome page, it is to be avoided at all costs, 80% of users hate it, moreover the timers run and it only creates a waste of time even worse if they are animated. A further downside to splash pages is the initial registration page. It creates computational effort to remember username and password, furthermore, if registration is requested on the splash page it can cause abandonment of the site because it requires a trust in releasing one's data.

- **scrolling**: an average user is willing to use the scroll for 1.3 screens, so 2.3 total screens. On the first page 23% scrolls, while on the inside pages 42%. The second visit to the home, shake 14%! The reference size for a site is 1024x768, but the minimum size to consider is 800x600. Often, especially in netbooks, given their small size or using a fixed layout (**frozen layout**), you focus too much on the reference size and forget about the larger ones. Or we focus too much on the vertical axis and create horizontal scrolls, creating a lot of computational difficulty. In fact, the horizontal scroll changes the classic information mode, in which the X axis is fixed. Very often the visible information after horizontally scrolling is lost forever!

4.2 Site creator problems, users

Bloated design: you define a site design when it is too thorough! Very nice but exaggerated, difficult to use and which creates a high computational difficulty.

The browser war has led to the introduction of new commands, spectacular but excessive for the understanding of the average user. For example

blink (blinking text), rotating text, moving text, etc.
An example of an exaggerated design is a menu design in an egg tree. Six out of ten users do not recognize it and therefore leave the site. In addition to visual abuse, there are**multimedia abuse**. An example is the**audio**on the site, hated by almost all users.
Another example is the site with**3D effects**, which is also hated by many users, it is preferable to use classic 2D (implies less computational effort). Another problem is the use of**plugin**. Their main problem is that they are not standard and therefore require an installation for the correct functioning of the site and therefore computational effort. A plugin should wait at least a year from its release to be used (TRUST factor, pulgin does not seem a safe thing), even if it still creates problems, in fact 90% of users who encounter a site with plugins do not install them but abandon them. the site.

An example of multimedia to be used in a moderate tone is the inclusion of**video**. It can work, has little computational effort but does not alter users' timers. The main problem is that it takes a lot of bandwidth to see it in a reasonable time even if it doesn't enter the timers. The average time of a video should not exceed 2 minutes, the ideal time is 1 minute! Obviously with the exception of sites dedicated to videos, for example youtube, here the timers are not valid.
Visual metaphors: certain symbols or signs that imply a thing, on the other hand, are disappointed. For example, a fake link, that is, a non-clickable link!
Metaphysical metaphors: certain objects that in reality function in a certain way and in the site change their functioning. For example, a button is actually clickable at any point, while some sites can only be clicked on the word written inside it. Create usability problems.
Another problem is to bring gods**desktop aspects in the web**. For some time now, the menu object has been brought to the web. The problem is that they are not exactly identical, in fact the desktop menu contains commands, while the web menu contains information. This can lead to a menu explosion.
Another problem related to the menus is the reference size, the menu is too large on the screen, creating viewing difficulties. Also to visit the menu 83% of users fail to center the boxes! And 54% erroneously exit, this happens because the human mind always thinks of the minimum path from A to B and if the menu is cascaded and closes when exiting the menu area with the mouse, following the minimum path. it will surely come out creating frustration and computational effort. A menu without closing timer used by a user for the first time, causes 92% of wrong closings!

The maximum number of sub-levels recommended in the menus is 2 and have fault tolerant (i.e. if I exit the menu it does not close immediately).

In the text, in addition to all the rules mentioned above, one must pay attention to the size of the text itself and the contrast with the background.

Text rules:

- Readable text, minimum size 10pt
- Give options for the resize of the text, the best way is with the various sizes already clickable.
- Text is text, the user must recognize that it is text and therefore can use it as such, with the most common operations (copy, past, cut, etc.).
- do not use too many fonts and too many effects, the golden rule is to use only one font for the whole page, maximum two. Favorite font Verdana.
- Keep a very high contrast between text and background.
- Uppercase text reads harder even if it carries more emphasis, an average user reading uppercase text takes 10% longer.
- avoid the use of images containing text instead of common text, it reacts badly to resize, increases the weight of the page, does not allow common operations, interacts very badly with search engines.

4.3 Curse of LOREM IPSUM

The limits of good and evil.
Normally a design first creates the layout and then inserts the text, to test the layout with text it inserts random words called lorem ipsum. Then it later replaces them with the real text.
It therefore leads to divide the layout from the content, giving priority to the first and adapting the second. When a user views a web page for the first time, he does not start reading it but his mind does a quick scan of the entire page and then later starts reading. A good design takes into account the scanning, which is done continuously and minimizes the effort. Large blocks of text increase the slowness of scanning, it is necessary to give a structure to the text to increase its speed. So lorem ipsum creates problems, in fact first I have to know the content and then create the layout.
How to**structure**:

- text divided into blocks
- structure the blocks with gods**descriptive titles**, helping the primary cataloging done by scanning
- You can use them**keywords**highlighted (bold) to help the virtual map, without exaggerating otherwise the opposite effect is created. They must be short and to the point. An average user memorizes 6/7 keywords.
- I**link**they must not be too long and contain the title of the section because it creates confusion.

• do not use links with names that are too similar, even worse use the physical address of the page

• never use click here as the name of a link, it only increases the computational effort, it is obvious that a link must be clicked!

• use**lists**in the text, creates order, helps memorization and increases user satisfaction by 47%, they must be used with at least 4 elements. Avoid using too many of them, they decrease their storage linearly to the number of lists present (3 lists make 1/3 compared to 1 alone) vertically. The number of horizontal lists, on the other hand, decreases the memorization effect exponentially!

• extremes of LOREM IPSUM, the inserted text is too large compared to the layout created and therefore the guillotine effect is created, the text is cut, or a scroll inside the paragraph is created, to be absolutely avoided.

• avoid the**blonde effect**, a paragraph with a very interesting title or image but then the text does not contain substance, or is inconsistent with the title or image that represent it.

4.4 E-commerce

It is commonly thought of a commercial site that the most important thing is the product, but that is not entirely true. In truth it is the product and the price! Users want the price and the next product visible together. The price, as in a real shop, should be placed next to the product. Problem of**hyper-association**: a summary of the products is shown without info and price and if a user wants to know you have to click on it, creating computational effort. Consequently, the user does not know how to quickly create the product-price association.
This problem creates**gambling click**(bet): they are the clicks to bet and it is not therefore certain that the action is completed.
This problem creates 40% less approval for the site. 30% of users never click on product images, which usually contain the price! So to avoid and bring the price back to the same plan as the product.

If we look at the site of a company that produces a product sector but does not sell retail, it usually does not display prices. The user is not satisfied even if the reasons are clear. You must always display the price if you present a product, you can give an approximate price, or a range of prices.

4.5 Classic advertising and web advertising

In classic advertising, the user reads the advertisement for a short time, so he is impressed and impressed. There are various ways to entice, including entice on price. In most cases, two methods are used:

- **fishing price**, bait price, a "false" price is displayed to entice the user;
- **net price**, price without taxes or mandatory additional costs on the final price.

In both cases the final price is not the one presented. Many web advertising campaigns use these two techniques, wrongly. To understand the reason for this error we must start from the brain, it is made up of two parts, short-term memory and long-term memory. the advertising takes advantage of the short memory, it makes it clear that there is a bargain in that article and in that shop, everything else is canceled, including the fake price!
This works because the time lag between reading the advertisement and entering the shop is enough to erase almost everything from the short memory, but on the web as in a shop the temporal and physical distance is so short that nothing is forgotten, so creates the opposite effect. A site that uses for example the fishing price is abandoned by 90% of users and the remaining 10% remains but with a significant loss of trust in the site.

The net price, on the other hand, 85% of users leaves, in this trick, in addition to taxes, shipping costs and any product insurance are not included.
A middle ground is specifying the final price and displaying it in the cart, this creates gambling clicks, because the user hopes that the expenses are specified, a not optimal method.
Another thing to value that is often not specified with adequate emphasis are the free products. Like the catalog of a site that is sent home free of charge, the term FREE must be spelled out in large letters, as it causes pleasure!

4.6 Product

The product has the same importance as the price, the very common mistake that is made is to think that a user already knows the product, giving him the minimum information about it. The average user believes a detailed description of it as well as the price is essential. If he does not find the description, he typically searches other e-commerce sites to get the description and the site is also considered unprofessional. 99% of users look at another e-commerce for the description and do not compare the prices, up to 20% savings remain in the site with the satisfactory description !!
In addition to the description, the vision of the product is important. When the user wants to see a product in detail, the timers turn off! The detailed view must be optional, if forced, the timers remain!
If a user is interested in the product he wants to see it in full screen too, if instead he is not interested the full screen only creates annoyance.
A trick to keep in mind is to never use filling images for the product but always use real images and perhaps in proportion. The idea is to give as many angles as possible, then give a 3D view with 2D images (Amazon).

Always respect the main rules of usability, so never open a new window for details. It must be remembered that there is always a need for detail!

5 Seventh lesson: Study of attention on text and images

A behavioral study on reading and focusing attention in newspapers and magazines was conducted in 1990-1991. It emerged that there are very specific classes of behavior, the most attractive part are first of all the photos and then the color used on the page, a page with colored parts and colored images gives the idea of   a greater wealth of information, compared to a page in black and white!
Between text and images in a newspaper, images prevail, 80% compared to 20%. When combined together, articles with images stand out more than a text-only article.
On the first page of a newspaper, the initial image is first looked at and then the text of the title is read, even if written in large letters, however, the image is first looked at. Another feature of a newspaper that creates a lot of pleasure and that has not yet been reproduced on the web is the ability to view two pages at the same time, opening the newspaper in half and making it a single large page (two open pages are perceived as one page) !

5.1 The Web

Subsequently, a study similar to the previous one was conducted in the web environment.
It was thought that the rules described above were valid, but it is not true, we must consider the web as another means of communication. The point of greatest interest and first viewing is the content in the upper left corner (no longer the larger image like in newspapers). Usually the logo representing the company of the site without text is positioned right in the upper left, this creates disorientation and discontent in the user.

Analyzing a page with a**eye tracker**, that is a gaze reader, it was possible to have the focus points of a generic page of an average user: an F-shaped or ice cream cone thermographic is created!
This in the first view of a page, but then with the scroll? The scroll is almost exclusively done by screen, when you scroll, a blind spot is created in the point where you scroll and it depends on the size of the screen, in fact you can see from the thermographs that where you scroll the text is hardly looked at .

Between text and images on the web, text wins!
How to improve a web page:
Multi-column layout is to be avoided because it creates a lot of difficulty in scanning.
Keyword: if you have more keywords on the same line it creates a tiring effect for the scanning phase (max one per line), if you use only the bold for the keywords it becomes heavy, it is advisable to use the underlining and the

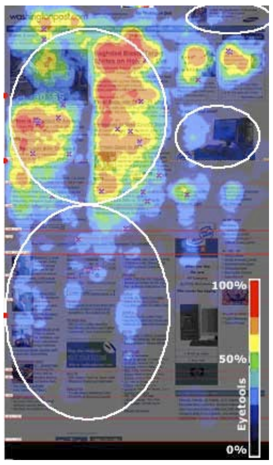

Figure 2: An example of cone thermography

bold or enlarge the text merged with the bold or separate the keyword line by itself, making it a kind of mini-title.
Paragraphs: short paragraphs attract twice as much as a single paragraph with the same text. Many small paragraphs relax the timers and entice you to read, 100% more!
Title: a short title is more effective than a long one. Another title enhancement is to add under the table of contents (blurb). There is therefore the possibility of having a shorter title and in the blurb expand its explanation. THE**Blurb**the points of focus on the thermographic map increase, lead to a greater digestion of information, but do not increase the desire to continue navigating the site. However, they increase the rate of return (+ 20%). Each blurb has its own micro-zone: being asymmetrical, the leftmost part is the first read, therefore more important, than the right part.
The layout must be**compact or not?**A user who has just accessed a page prefers the wide, but then the compact wins in the scroll. So a compromise must be found between the two, in fact:
Separation=faster scanning, recommended in the pages for navigation, full of links.
Compact=better learning of the content, if there was separation it would be created**diluited design**which would only bring annoyance.

Images

They are always of series B with respect to the text but also important. The user wants to see the images in a fairly viewable format, minimum 210x230 px, if the user is interested in the image.
Image vs text, always prefer text, even if an image is placed in the hot zone! The images, however, attract clicks, even if little seen 20% of users click on the images even if it is not specified that there is

an action. It is therefore advisable to always make the images clickable. Another important feature is the escape route from a page which must always be clearly visible so as not to create difficulties for the user.

The heat map is an important tool but it doesn't take timers into account, therefore time!

Fitts law:

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

This law is widely used, in fact it has a 98% match! Indicates the time it takes a user to move the mouse from point A to point B.

a = start / stop, ignition time
b = co-speed (slowness) - how fast we are to move the mouse D = travel distance
W = width, size of the destination

So based on the previous law is a point and click or a drag and drop to be preferred?
Definitely the**point and click**, in fact, drag and drop has a slower execution speed due to muscular tension, moreover it increases the distance between a and b in a linear way.

Good rules to respect are:
Minimizing the distance between A and B and making the destination large enough, you should generally try to create a middle ground between the two.

With these rules you can understand even better why an average user doesn't like menus on a site and why they frustrate using them. When we saw that Fitts's law works, we tried to respect it by creating a balanced menu, each submenu is centered, allowing a direct path to it.

Furthermore, the**target size rule**: a more used button is displayed of a larger size than a less used button (for example office). Another ploy to comply with Fitts' law is to use screen borders. Fitts in fact sees the edges as an infinite button, therefore very efficient. With this law it is more efficient for example to use a button on the edge on the other side of the screen than to use a simple button a few pixels away. The use of buttons on board is greatly exploited by the Mac interface, it has been calculated that using the Mac menu is 5 times faster than a Windows menu.

The touchpad can also take advantage of Fitts's law, using the side edges for special operations, such as scrolling.

Corners are even better, they have a better access time because it takes even less time to reach them.
For ease of use, from most important to least:

- bottom right
- top left
- top right
- lower left

As far as right-handers are concerned, for left-handers it is symmetrical.
Even more important and that fully respects Fitts' law is where you are!
Then use the right button to view the popup menus as they have the top accessibility.

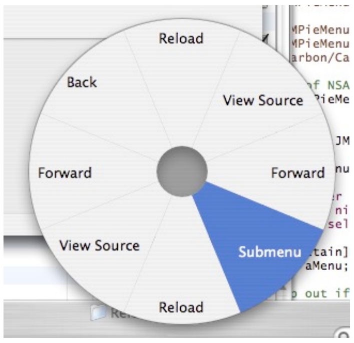

Figure 3: An example of a Pie Menu

The best menu is the**Pie Menus**I right click and a pie appears around the clicked point, with the various fields at the same distance. It reduces the computational effort.

A variant of the Pie Menus are the**Fun Menu**, are very similar to the Pie, but are found not in the focus point but in the corners or edges of the application.
Obviously the Pie Menu is better than the classic menu. A variation is the combination of the two.

The menus**Linear**instead they work well if they have a lot of entries or when their description requires a lot of text.

The application of Fitts and its rules is widely used in video games, because they must be as usable as possible.

6 Eighth action: Advertising

Users hate advertising, only 0.4% click on it!
How to increase the click-through rate, first of all it must be beautiful and attractive, and have a good positioning,**better places**:

- left column

- top of the page
- right column

The worst place is at the bottom of the page.
Advertising is important because it is part of the classic business model of web applications. It allows you to make a service for free, which leads to an increase in the number of users and the revenues are obtained through it. Advertising on our site usually comes from external sites.

Alternatively, if an advertisement is placed among interesting text, it gets noticed more. The shape and size of the text itself also positively influence.

To be avoided absolutely:

- 10◦sounds automatically, gives an unpleasant message 79%
- 10◦on the go 79%
- 9◦flashing 87%
- 8◦takes up all or most of the page 90%
- 7◦moves across the screen 92%
- 6◦does not say what it is 92%
- 5◦covers the text you are reading 93%
- 4◦it is not clear how to remove it 93%
- 3◦tries to get clicked 94%
- 2◦loads slowly 94%
- 1◦pop up 95%

So you have to use other tricks to attract the gaze of users, in real life beautiful people or bright colors attract, but on the web? The images, as already mentioned, are of series B, because the**zapping effect**, one thinks already seeing an image that it is an advertisement and discards it. You can avoid the zapping effect by proposing ordinary people who behave strangely, or who despise the product itself! (an example of the opposite effect is the big brother).
If I insert an advertisement in the middle of interesting text, but the image has a border, I create separation and thus activate the zapping effect. A positive example is the effect**Blending**, normal content is mixed with advertising, the training set is confused. The web games are a good way to attract users and have a high effectiveness even if trivial! Google, for example, uses Blending by inserting advertising messages in each search result, instead of the first three results, separated from the real results by an almost invisible light yellow border (required by law). Other effects to attract attention:

- If there is someone in the image looking at something, users focus on that thing.
- In images with people displayed in full, the focus is more on the private parts.

Care must be taken not to insert advertising that is too disconnected to the content, it is created**distraction effect**, the timers go down by -40%, it creates frustration and the desire to return to the site goes down by -80%! For this reason, there is an increasing attempt to use not generalist advertising but**Behavioral Advertising**, that is advertising adapts to the content and to the user who views it, tracing a profile, then recording the interests and hobbies. This is possible because users can be diversified on the web, this type of advertising has an effectiveness up to 100 times higher than normal and an advantage factor of at least 10 times! Furthermore, this technique has no negative effects on the timers and increases the desire to return by 100 times!

6.1 Research

Critical thresholds for differentiating research:

- up to 100 information items, it is not mandatory to provide search tools
- from 100 to 1000 it is crucial to give a good search tool
- over 1000 it is essential to provide an excellent search tool that covers the site 100%.

On average, a search tool is used by 100% of users.
Since the user is used to search engines, he expects to find a search that is almost the same also on the site. Furthermore, since it can arrive on any page of the site,**Deep Linking**, 60% of users take advantage of the search immediately.
Solutions:
Simulating a search engine, for example the google tools, locally for the site, is a no-cost solution. However, the search does not work well, because it is not designed to work at a low scale, if the user reuses the search on the site and has arrived with a search engine, it means that the search engine has not provided him with the information he is looking for and therefore the search made with the same method will display the same result. Another negative point is that search engines cut off 50% of the indexing and therefore it will also be done locally with loss of half of the information. The site map is useless for indexing.
Solution 2:
Create a local search, but paying attention to respect the habits of an average user, being used to search engines, it must have a very similar interface, with a text box and a search button, the button must have the precise name search or search or search, in order not to confuse the user and must be placed on the right (the button on the left creates a delay of 2 seconds).

Less is more.
Putting too many things creates a negative effect, a simple search with a box next to it is the most immediate and the most appreciated. Google removed the advanced search button from the main page because just reading the button made it difficult for the user!
You have to be as minimal as possible, being careful not to overdo it.

Never postpone a local search to a global one, sending the user to the google results page!

Constrained search

A constrained search can be useful when combined with the classic search, which is never displayed alone. It is efficient and users like it, but care must be taken whether to make it static (enter the data and then click on search) or dynamic (each selection of a feature displays the results). In **static search**the user applies constraints and works only on explicit user action. It creates confusion, the user seeing a search button looks for the box to enter the text, but does not find it !! There**dynamic search**instead it takes a few seconds each time to display the results. In dynamic search, as soon as I fill in a field, the search starts, it is worth using it when there are few constraints so as not to waste time for the user.
Many users think that static search is dynamic, they select the search and wait for the results, then they realize that they had to click on the search button, creating frustration.
Static with many constraints is used, while it is better to use dynamics with few constraints. A union of the two is the**hybrid search**, if all the constraints have been filled in, the search starts by itself, otherwise it is static and starts by clicking on the button.

Search results

In the search results it is very useful to allow the sort on all results and in both directions (price from + high to + low and vice versa).
If the result contains 0 data the best solution is to display a no result message.**God is in the details**, so in the 0 results for example!

6.2 links

Links on a page can break. You can choose three ways of behavior, or set to be automatically rendered on the home page, or display a page with the error, or leave the default display of the broken link to the browser. The last option is the most wrong. The best thing is to see a clear and simple message that the page has been removed.

A case similar to the 0 results can occur with the**effect 404**. Never do not manage non-existent pages. Help the user as much as possible by giving as much information as possible, for example by adding a search bar. The 404 page management site b3ta.com took all the funniest 404s to create theirs. This solution is very welcome, the average user spends 2 minutes and 30 to look at the images.

6.3 exposure

A grid view of products is more compact and displays more products on the same page than a list view. The problem is the search for a product, the time to find it is very variable, the so-called is used**random walks**because the degree of freedom to search is too great, a dynamic explosion occurs in the brain. For this reason, search engines use a list view of results, because the grid creates a great waste of time in micronavigation.

6.4 search box

At the beginning of the use of the web, when users used a search box, in most cases they entered single keywords, with the passage of time users use more and more words. The recommended average length for a search box is 30 characters, this measure satisfies 90% of users. (Google 62 characters, Bing 42 characters).

- Google: 64 characters -> 100%
- Yahoo: 36 characters -> 95% (but search is in the background)
- Bing: 50 characters -> 97% (design choice)

If a box is too small, scroll occurs, creates stress in the users, l1% for each scroll character, furthermore a small box induces the user to create shorter queries and therefore less specific, therefore less precise and poorer results. You can also use a hybrid view, when the box is not used you can view it small, once you click on it it enlarges (Volunia).

6.5 Research 2.0

Increase your satisfaction by -43% !!! Create damage!
Just remember the less is more, the user sees the high level of iteration, so the expectations grow a lot, but is unable to meet them, hence frustration. If the humanoid is more of a cartoon, then we see that it is virtual, the expectation drops and therefore the satisfaction goes up. The ikea site gives 2.0 research as an alternative to the classic search and with a basic 2D interface made of cardboard style, which makes expectations very low, the satisfaction is +70% !!! 10% use it.

Voice XML

The web seen as a visual medium is very limiting, the voice is becoming a medium.
VoiceXML was born to go from the screen to the phone, the voice is translated into text and you always interact with only text directly with the computer.
VoiceXML uses various technologies:

- Speech synthesis marchup language (SSML), passes from enriched text to voice, then gives the possibility to give accents and all tonal enrichments to the computer voice, using special characters.
- Pronuciation Lexicon Specification (PLS),
- Call Control XML (CCXML), manages the human-machine interaction, is the orchestra conductor!
- State Chart XML (SCXML), is an evolution of CCXML, it is more general
- Speech Recognition Grammar (SRGS), defines the action grammar, the context in which it is speaking
- After applying SRGS, I interpret the resulting text.

The SRGS in voiceXML are defined with a file called <name> .gram Speech recognition takes place not with VoiceXML but through an engine

external. For higher quality the voice recognition of siri takes place in
a dedicated server and not inside the mobile phone. In
1966 Eliza (first virtual psychologist)
In 1975 the first adventure video game, you could freely write that-
what you wanted to do.
Nowadays, pre-made sentences are provided to the user, the degree of freedom has dropped a lot because very often it was not understood and therefore it created strong frustration in the user.

It has also been noted that making the characters move makes unwanted movements, called fractal noises, more pleasant by an order of magnitude.
The noise is in fact the key, a curve is created that starts with a large amplitude and little vitality and all pieces are created continuing to increase the vitality and decreasing the amplitude. Eventually they overlap and create an effect of anything very real, for example waves or ground or the movement of grass. These techniques make everything more real!

6.5 Interfaces 2.0

There**noise component**creates the feeling of verisimilitude. It has a fractal behavior and if the noise is out of phase it gives us a strange feeling. An interface made with hand lines, full of noise, is warmer to the user than a classic one with straight lines.
THE**wearable**for now they don't work, they interfere with our reality. THE **Google glass**they have too invasive and stressful input. The output as a popup is negative and annoying, and the physical media is heavy on both a physical and psychological level. The**smartwatch**have greater potential than GG.

7 Search Engines

SERP: is the ordered list of the results of a search, which is displayed by a search engine.
More than 95% of the clicks are absorbed by the top ten, in detail:

- 1◦position takes 51% of clicks, therefore more than all the other 9 on the page;
- 2◦position takes 16% of clicks;
- 3◦position takes 6% of clicks;
- 4◦position takes 6% of clicks;
- 5◦position takes 5% of clicks;
- 6◦position takes 4% of clicks;
- 7◦position takes 2% of clicks;
- 8◦position takes 1% of clicks;
- 9◦position takes 1% of clicks;

The tenth position ...
Black Jersey effect(in Italian effect**malabrocca**)
in the last place of the top ten the click rate doubles! in fact the clicks are at 2%, the visibility is higher than in the two previous posts.

An essential feature for a site is to be seen outside. Typically, users come to a site via search engines. It is therefore necessary to be as high as possible in the ranking of search engines to have visibility.

Does mixing images and text in the results create differences in reading percentages? No, when the user enters the section dedicated to images, the percentages stall, once he returns to the results page he picks up where he left off.

7.1 Spam Index

Also said**SEO**o SEP, it calculates the score of a page, that is the score that the search engine assigns to position it. It consists of a textual component (written + code) and a hypertextual component.

TFIDF(Term Frequency Inverse Document Frequency)
It gives the weight of how important a word is to a page. It is formed:
TFIDF = TF * IDF

TF
Example: given a web page made up of 100 words, "foo" is contained 5 times in it, so TF = 5%
Giving a more formal definition is the percentage of occurrence of a word on a page. The main problem using only TF, that with common words like for example an article, the TF is very high. It must be borne in mind that if a word appears too much it is a link and not a content one.

IDF
It is the inverse of the frequency of the word, on a logarithmic scale.
Example: given a web page of 1000 words, the article appears 980 times, so 98% frequency, log (1 / 0.98) = 0.008.

To calculate the final value of TFIDF = TF * IDF

By giving a simplified definition, a search engine takes all the pages containing the searched word and calculates the TFIDF of each (usually already pre-calculated), if more than one word adds the TFIDFs. Raising the TFIDF too much on some words makes it lower on others, so you have to try to focus on one group of words and leave the others alone.

7.2 Increase positioning

To increase the ranking, one must try to increase the TFIDF of the site's keywords. There are various techniques:

- **Body spam**, you put the words in the body of the HTML page, it is effective but disadvantages the reading for the user.
- **Title spam**, keywords are inserted into the page titles, the content is changed much less.
- **Mega spam tag**(<meta name = ”keywords” content = ”...”>), the main advantage is that the content is not modified, but in today's search engines it creates a disadvantage!
- **Anchor text spam**, insert keywords in the text of the links, the score assigned compared to the common words is higher, moreover, even the landing page receives benefits in terms of score, increases the link targets.
- **Spam URL**, insert keywords in the address of the page itself, they too receive bonuses over common words in the text.

Keyword insertion techniques:

- **Repetition**: the word is repeated several times, paying attention to the TFIDF, which if too repetitive creates damage. This technique is easy to spot by search engines.
- **Dumping**: inclusion of many terms that are little used even if not inherent, if they have low TFIDF they will be present in very few places and therefore the score will be high!
- **Weaving**: I insert pieces of other websites into the site, creating more interesting content, seen by search engines as an advantage for a higher score.
- **Stitching**: you copy and paste other web pages and assemble them together with the site, you create interesting content to populate the site, since search engines consider the percentage of content, this technique increases the overall bonuses.
- **Broadening**: I insert not only keywords but also synonyms, similar words and related phrases. Not only to cover user requests, but search engines use the keyword similarity technique, words of the same context give additional bonus pages. To determine if two words are in the same context they use the matching percentage of the words in the other sites.

There are various ways to choose keywords, the simplest is to use Google Adwords, which displays the recommended words and the percentage of searches with those words by users.

Using spam index (text spamming), however, is not very pleasant for the user and creates disapproval as it changes the content of the page. There are various techniques to overcome this problem:

Hiding(hide), divided into various subgroups:

- content hiding, put spam text in the same color as the background
- insert 1x1px images.
- technique of**redirection**(technique 302), from the page with the spam to the clean one using the refresh. It is not very effective and it is easy to intercept, it is much more effective if done with javascript, because search engines find it much more difficult to interpret a javascript code.
- **cloaking**, it is a redirection to the nth degree, it does not use a redirect, but it checks if the page is requested by a user or a search engine and, depending on the response received, it displays different pages. If the search engine finds that a site uses cloaking it will be banned for a period.

A large part of pagerank is given by hypertext links.

7.2 Pagerank

A large part of pagerank is given by hypertext links.

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

Calculated recursively, it counts the number of incoming links to determine the score. Pi (w) is the number of links pointing to v, a (w) is the number of links on the page.

This pagerank calculation was easy to manipulate with the tricks seen above, so we thought of a reformulation, it was decided to apply the **Markov chains**And**random walks**. Markov chains manage complex systems, in general they say that to build the next moment you can approximate the story to the moment before. So given what the web metaphor is**random surfer**, i.e. wandering around the web at random, it was decided to take this random factor into account:

Calculation**iterative**

Given the following simplified web: A
<–> B <–> C

A = C = 1/4
B is therefore double 1/2

One must try to avoid causing the so-called**spider traps**, that is the infinite navigation in a spider site (it is the Google bot that checks the scores) of a search engine, for example: a calendar that if I continue to enter the next day I can go to infinity. This problem happens often!

Effect**Island**:a site with only incoming links is trapped for sure, the "liquid" will slowly be brought to the site, so pagerank 1. For example Microsoft used this trick to get high pagerank. If I have these two web clips:

A–> B–> C–> (A) Q–> P
-> R–> (Q)
Towards the liquid not evenly, but a little more in the first island than in the second, the liquid will always remain in abundance in the first compared to the second given the absence of connection. The pagerank must be changed.

Calculation with**Teleportattion**
$\pi_v = (1-\varepsilon)\,(\Sigma_{(w,v)}\pi_w/a_w) + (\varepsilon/No.)$
N is the number of pages on the web.
The bot does not just browse but can go anywhere on the web. It is used to avoid spider tracks, because even if I have many pages, sooner or later I will come out.
It was decided to insert artificial links to connect the disconnected parts, it is taken and teleported to a random page of the web. If teleportation is disabled, the pagerank is the original one, otherwise it is the modified one (flag teleport = 0 or = 1).

Calculation with**Total rank**
$T. = \int_0^1 r(to)\,from$
It makes an integral average of all possible choices. Average between teleport = 0 and teleport = 1. The computational cost is the same as the initial one!

Example
A–> B <-> C
B and C = 1/2
A = 0

There always remains the problem of**dead ends**(Road closed)
A page without outgoing links accumulates most of the liquid even if teleportation is applied, the solution adopted is the penalty! It was decided to heavily penalize the pages without outgoing links so that the liquid is lowered!
Search engines to start the problem add links, changing the structure.

7.3 Rules for increasing the pagerank

Add In-link , the pagerank always increases!
Techniques:

- **infiltration**: you infiltrate other sites to insert links to us, for example in blogs, bulletins, wikis, etc.
- **honey pot**(jar of honey), I create palatable content and thus attract links! You can create it by copying and pasting other sites.
- **link exchange**, exchange of links by agreeing with another site.
- **resurrection**, a domain dies, you decide to take it and put a link to your page, surely it will have had some incoming links and therefore bring liquid to ours.

The million dollar page worked, subconsciously because having a link meant them being on a page with a lot of links and so you were getting a lot of liquid.

Add Out-link
For symmetry the pagerank is lowered.
Techniques:

Socidity, by itself adding out-link does not increase pagerank but in reality it does! Because the page analysis can change positively!

Spam farm

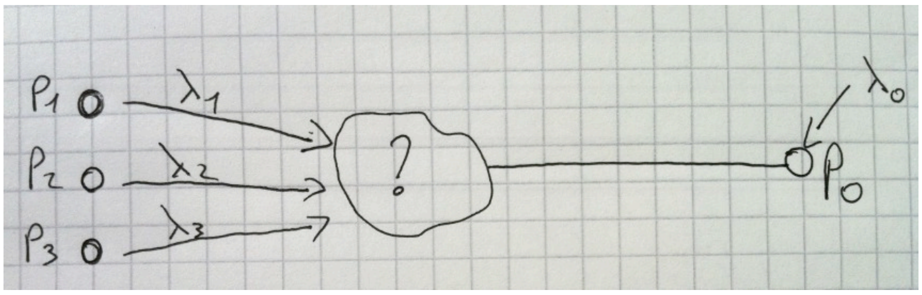

Figure 4: An example of spamfarm

Take pages under our control and point them to the page we want to give importance to.
As you can see, the**reachability** (wherever I am I can get to any other point), that is, all the pages are connected, so the spiders can visit them all! To make it true, just insert infiltration links, i.e. artificial ones on nodes P1, P2 and P3.

Optimal Spamfarmis:

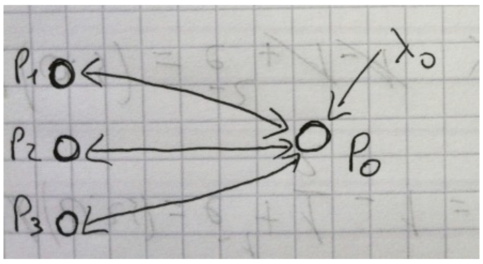

Figure 5: An example of an optimal spamfarm

Also giving the return link. This spamfarm uses the fewest links and maintains reachability!

7.4 Alliances between sites

In this section we see how the alliances between two or more sites can lead to an improvement and greater stability of the pagerank of both. There are several ways to link two or more sites:

Deep alliance

The pagerank in this case is the<u>average pagerank of each site</u> , the advantage

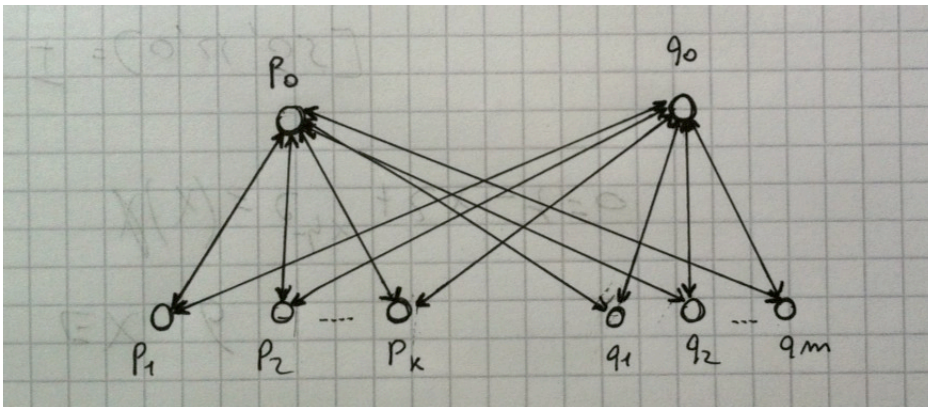

Figure 6: An example of a deep alliance between two sites

is that if one of the two sites loses pagerank, with the average it remains stable.
Our secondary pages point to our target page and that of our ally and vice versa.

Higher Alliance

The pagerank is calculated as the maximum between the two pageranks of the individual sites.

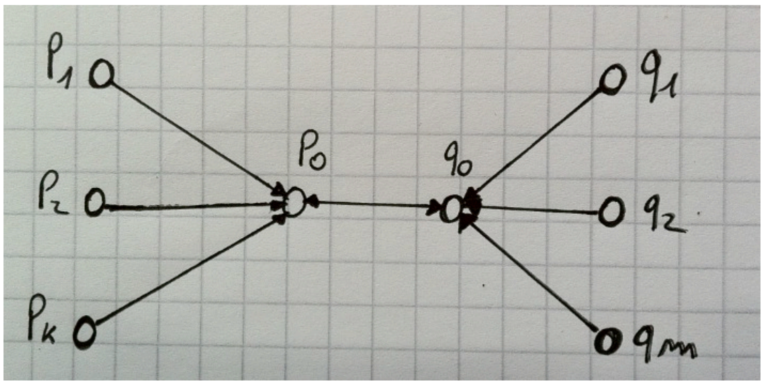

Figure 7: An example of a superior alliance between two sites

Ring structure

The ring structure allows a high pagerank, but if one of the internal links

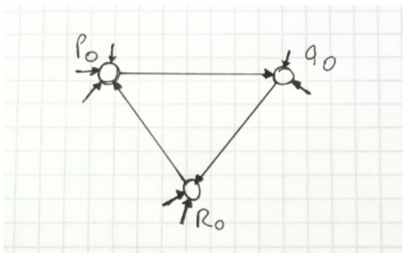

Figure 8: An example of a three-site ring structure

falls, many connection problems are created and therefore pagerank balancing problems.

Complete came

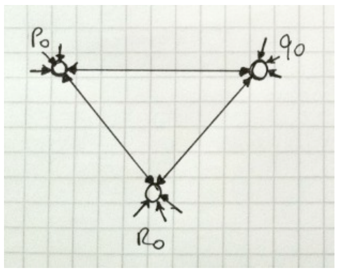

Figure 9: An example of a ring structure as between three sites

7.5 Engine countermeasures

Search engines obviously have to find countermeasures to limit the tricks described above. On complete structures such as rings, they identify them and penalize all sites that are part of them. To identify them, just find all the focus pages and the strong connections between them. The main problem is the quantity of connections to check, think that for: N = 3 sites, I have to check about 18 links
N = 4, 1606 connections
N = 5, 565080 connections
and so on, so with 10 sites I will have an indescribable amount of links. Cannot find spam farms.
It was therefore necessary to carry out checks in a different way, we started from the statement that the web has a shape, we then take the website and analyze its shape, if it differs a lot from the "standard" form then it means that in it uses shortcuts and is penalized! In addition, incoming and outgoing links are analyzed and all links that deviate from the average are penalized.
Another countermeasure is the use of the two types of pagerank, the original one and the one with teleportation, both are calculated and their difference is called **relative spam mass**. If the spam mass is very high, the site is suspicious and therefore penalized. The success rate with this technique is 95% -100% in finding spam.

Looking at a graph on the comparison between spam and positioning, fairly marked classes are denoted, in the first place there is a minority of sites that do spam and are not discovered so the positioning is excellent, the second place instead there are sites that do not apply techniques of spam, the third place is practically empty and in the middle of the graph there are most of the sites that make spam and that manage to obtain an average positioning.

Pagerank can be seen as: (1 - e) 1^{T}.N + and L with
matrices instead: (1 - e) 1^{T}.N + PL
where P is a matrix that allows the choice of a personalized pagerank, the nodes are not all the same but they have preferences.
Google uses the personalized pagerank, in fact it decides through a specially created team to favor or disadvantage sites with teleportation according to their personal judgment.
For example searchKing was a competitor of Google and was manually penalized !! SearchKing sued him but lost it since pagerank is an opinion and therefore can be subjective!
Google can then do what it wants.

It would be even better to give the user his personalization, something closer to him, his personal ranking, rather than just the opinions of one (Google).
Each personal user profile requires the calculation of an optimized pagerank to display customized rankings, also considering the fact that the profile and the queries it makes are dynamic (the pagerank calculation must also be dynamic).

There**complexity**of a custom calculation is: if there are N pages in the web, even if we restrict ourselves to two customizations per page, we have 2 ^ N possible customizations. There are some**Good news**: the customized pagerank is linearly modular. I can calculate the separate pageranks and then add them, without recalculating them. So we will have N personalized pageranks (for example: I visit 1000 pages, then just add the 1000 mini pageranks of those 1000 pages). Scale to a complexity N, which is however a very large number (number of web pages). In practice:

- **Topic type pagerank**: we select a certain number of keywords and we calculate the customized pagerank on the pre-selected ones. Then, depending on the user profile, we can build its dynamic private pagerank by adding the customized pageranks of the individual keywords.

Care must be taken that each variant of the pagerank that is used is compatible with the countermeasures. The custom pagerank is compatible with all techniques based on the web structure.

7.6 Janus graph

How to determine that a site is good?
Until recently the rank of the sites was considered only the good part. ranking = measure of goodness. Pagerank in the first version was fine until sites started to adapt. There has been a change of perspective as the positive and negative aspects are on the same level. For this reason it was decided to use a pagerank with two associated measures:
W_+good part
W_-bad part
So the old pagerank is calculated with the combination of the positive and negative parts.
$R._{GOOD}(G._+)$
$R._{BAD}(G._-)$
G is a part of the web graph containing many W.
Given an old style pagerank, to calculate the new one**dual pagerank**and since the positive part of G reflects the negative part of G I can focus on calculating only one of the two and then refracted. So: $A._J$= R (G_+) - R (($G_{R.}$)$_-$)
where $G_{R.}$it is the reflected web.

We can now focus on calculating positive W and negative W.
They can be calculated by taking as many information sources as possible, for example emails are a very rich information space. Emails are already a network structure, so just integrate them. Whenever a URL is mentioned, a link is created from a mail object to a web object and vice versa. The popularity of the link, in addition to being calculated on the number of links in the emails, also determines the negative or positive judgment used in the email for that link. Furthermore, the popularity of the link is determined according to the number of replies that the email containing the URL receives.
Other information that can be obtained from the emails are: the age of the email, the history measured on the temporal quantity of emails, etc ...

Even the bad part can be influenced by the emails, in fact every time an email is considered spam and contains a URL, the bad part is transmitted to the pagerank of the site mentioned. So the work initially assigned to a paid team to give weight to various sites is now done directly by millions of users every day, simply by using the spam button!

Lately search engines are trying to connect more and more the social parts of the web to personalize searches, the introduction of emails for example has led to a higher quality of personalization, because emails are much more truthful, as they use many security systems to avoid being generated automatically (captcha), moreover users can be tracked and their profile can be created (cookies, logs, etc.), but web pages cannot.

Still being tested is the SIS (social information system) each object is associated with a VID (user identifier), if two objects have a different VID then it means that they are from different subjects, instead if the same, it is the same subject and therefore you only take one of the two.

Since today's rank is based only on paths and does not consider the social part (**social ranking**) you are trying to insert this variable in the pagerank calculation. For example, if a path passes through social networks, it certainly means that it is of different subjects, therefore more value. Google has already begun to integrate these systems.

If I have a URL that is contained in several pages but with the same domain, it means that it is of the same subject so socially it is equal to 1. Therefore the unity of the web page falls and the paths can be broken. We are trying to go no longer towards a single information system but a socio-information sphere that contains all the means to obtain useful information (EMAIL 2.0).

8 tower of babel

10 years ago it was thought that the evolution of the web would be using automatic agents, which locally interfaced with the web and carried out user requests.
Example, the user asks the agent: send a rose to my girlfriend. The main problem is that 10 years ago only HTML was used which is simple but limited, so the agent could not understand the requests 100%, for example without having a precise explanation of the term rosa, as flower could search the web for any what was called pink, and send the one with the lowest price (for example a sadomasochistic kit !!). If XML had been used, the problem did not exist.
The advantage / problem of XML is that it is flexible, so there are many "dialects".

Since the web is distributed, to work well with automatic agents you have to be able to create aggregation of the various parts, the main problem is that if XML uses all the same dialect, it works well, otherwise it doesn't. In this case, the philosophy of the open world fails.
Since XML works in trees, it is very difficult to aggregate multiple trees together, so the information doesn't work well. In that, when you join two trees, you no longer have a single tree, but a forest of xml documents.

8.1 Semantic Web

An attempt was therefore made to implement the automatic aggregation of information through dedicated tools.

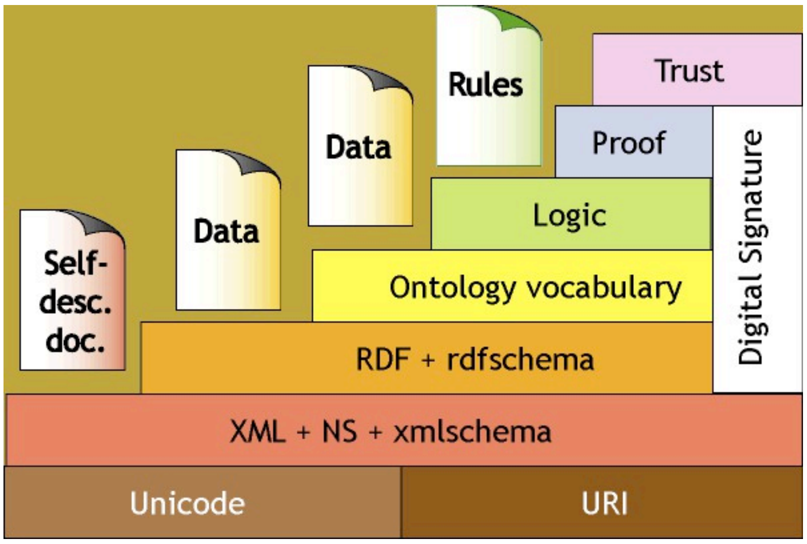

Figure 10: Scheme of the semantic web

8.1.1 RDF

The base layer represents the universal language to use, RDF (Resource Description Framework) describes resources and is the cornerstone of Web 2.0, it describes relationships, creates metadata, data about data, descriptions and interoperable concepts. RDF uses the principle of simplicity, in fact if something is not simple on the web it is already dead! It is an Enriched Entity Relationship model. Use the following basic grammar: subject, predicate, object complement. RDF uses graphs:

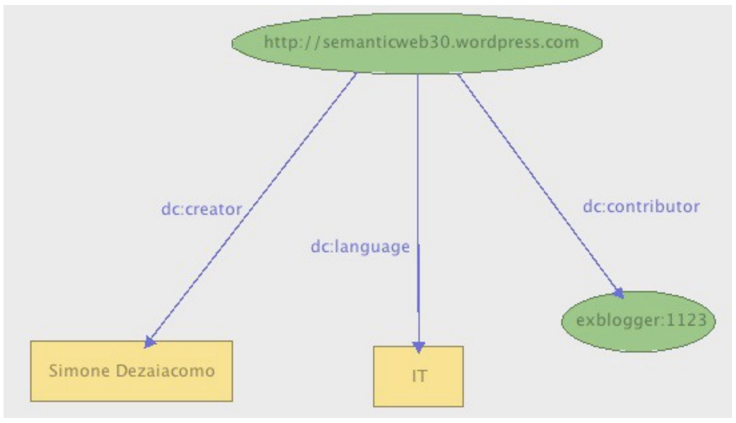

Figure 11: Example of an RDF graph

RDF also implements two writing modes:

- XML
- N-triples

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: D escription>

<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: D escription>



</ rdf: RDF>
```

N-triple
This language, on the other hand, is much simpler as it was created specifically for RDF.
Example:
How to**integrate RDF into the web**? XHTML2 provides two new attributes: about and property.

- about subject
- property verb
- object complement page

Using RDF, then using graphs for information representation, aggregation works very well. It also works well:

- reification: put quotation marks to speak at multiple levels
- containers: add conjunctions for objects
- variables: an anonymous unspecified object

An important property is monotonicity, that is, take some information expressed in RDF, and supposing that it is true, then, every step of this information is true.
RDF defines the basic building blocks but more is needed. Classification of objects, hence the use of classes.

More structure allows greater integrity and deductions (reasoning), in fact for example we consider the red wines class that contains the Cannonau object, it can be deduced that Cannonau is a red wine, even if it is not explicitly written.

8.1.2 RDF-schema

RDF-schema is the standard that allows the support of ontologies, formed as basic elements by:

- class
- rdfs: subclassof
- individual: concept of the individual

Also you can define the information structure as DTD does with XML.
Properties of the arcs always defined by the scheme:
- RDF: property
- RDFS: subpropertyof
- RDFS: domain type of relationship that outlines the verb
- RDFS: range

RDFS is the namespace of RDF.

Example
Property = eat
Subproperty = act
Domain = animals
Range = foods

With the tools listed above we can give information and catalog, but is XML schema enough to have a complete tool for all needs? No it's just the appetizer, called taxonomic.
It can be seen that RDF works better if developed in oper-world, whereas XML works better if in close-world. RDF also sets the complexity of information aggregation to 0.

8.2 URIs and Names

The name**social side**impacts 10-20% overall satisfaction, you need to choose a name that works.

1. The names**short**work better
2. The name should be**one of a kind**and don't get confused with others. Don't even take a plural if the singular is busy.
3. Take the**.com**, impact of 4.5%.
4. It must be**easy**to write and memorize.
5. Don't make up words. At best, join old words. (1.5%; -4.5%).
6. Beware of the sound, it must**sound good**. A site that starts with vowels earns 3.7%, [rjyw] 2.9%, [fvsz] 3.3%, [pkt] 1.9%. Sounds that are associated with bad words create -44% damage. In other contexts, such as an adult audience, it generates benefits of up to 7%.
7. No dashes -3%.
8. Use the**numbers**8.2%

0. Pay attention to how to check if a name is free or not because they could steal it. Internic tip.

The**URI**are the names on the web**technical side**.
The connections of the graphs are represented as web addresses, in fact the names on the web are the web addresses themselves. There is a lot of confusion between URLs and URIs.

URI(Uniform Resource Identifier): they are the names with which the resources on the web are identified (it is a superset of web addresses), they were created in 1998 in the RFC2396 regulation, the creator is Tim Berners-Lee. The document contains axioms to be respected:

- **axiom 0**: universality 1: a URI can be assigned to any resource and anywhere
- **axiom 0a**: universality 2: every resource in the web world that matters must have a URI

For example www.libero.it is not a website but the browser interprets and translates it.
URI example:
http://www.libero.it
news: it.cultura
Tel: + 358-123432426
URIs are the identifiers of the web, they are divided into URL and URN.

URL: Uniform Resource Locator, are URIs that identify the resource through a representation of their primary access mechanism, tells how a resource is found.
URN: Uniform Resource Name, is the real name, remains unique and permanent even when the resource is not available or is deleted. URIs can be absolute or relative, they are formed: Schema: part-dependent-on-the-scheme.
The schema defines the semantics for example:
Schema: http: //
Part-dip: Corsi.math.unipd.it/tecweb2
URIs can also be Hierarchical or Opaque:

Hierarchical: schema: // authority / path? query
Taking the example above again:
Authority: corso.math.unipd.it, indicates that the rest of the URI is under the control of an authority.
Path: / tecweb (formed from 0 to more segments) are not directories but hierarchical separators.
Query: information interpreted by the resource typically used to query the database.

Opaque: schema: opaque-part
Example:
Scheme: mailto:
Opaque-part:massimo@W3.org , like path but without '/'.

URN, use the following syntax: urn: NID:... (NID is a namespace similar to the schema). For example the ISBNs: urn: ISBN: 0-123-45678-89
New Zealand has acquired its own URN (URN :: = ZLN). IETF, is the body that registers all URNs around the world and keeps track of them.

In addition, in URNs there are characters allowed, characters not allowed, characters not recommended, usable with the corresponding% ASCII.
Example:
http://www.unipd.it/a/b/c/d http://
www.unipd.it/a/b/c/%2Fd
Although the corresponding ASCII is d they are not two identical addresses, but are considered two different URIs.
For example, you must be very careful to use accented letters which are considered an escape character, so they must be represented with ASCII. Very often URIs are created and managed badly and create many problems for search engines, since they compare the addresses of the various pages to understand if they are of the same URI, very often being written in a non-identical way (key sensitive or use of accents) , cause wrong recognition.

URI**variant problem**: many URIs represent the same concept, the usefulness of URIs decreases exponentially with the number of variants. Very serious problems have arisen with the axioms, various names for each level.
URI**variant law**: the usefulness of URIs decreases exponentially with the number of variants.

Axiom 1: Global scope: It doesn't matter where you assign a web name but it will have the same meaning (URI problems**Variant Meanign**).

To avoid all these problems, OWL was invented.

8.3 OWL

OWL (Web Ontology Language Overview), adds other features for more information. In OWL there is the management of equality or inequality between objects. Helps improve the Varian Law URI plague.

Equivalences:

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

for example it specifies more characteristics to verbs:

- inverseOf
- TransitiveProperty
- SymmetricProperty
- FunctionalProperty
- InverseFunctionalProperty

restrictions on types:

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

Prolog: is a PNL (Pseudo Natural Language) interface to display the potential of this technology. This system is much more powerful than OWL.

Through relationships we are defining a logic, but then we must be able to do automatic calculations and therefore the logic must be made executable. BUT first order logic is not decidable.

8.4 SPARQL

Anyone can insert a sentence on the web but the risk is an excessive increase in complexity. There is a risk that a search will never end, so you won't see the results! A different choice is made, it is decided that the search is decidable, as for example SQL was created precisely to have a decidable language on a large amount of data.
SPARQL is the equivalent of SQL for the semantic web, therefore for RDF and not for OWL.
The fundamental building block is graph pattern matching:*?x verb object.* The queries are very similar to classic SQL, but it uses various types depending on the context, for example in the social web it uses FOAF (Friend of a friend) or uses the classic RDF triples.

DC - Dublin Core:it is the standard for describing the basic properties of documents and is made up of 15 information elements (title, subject, etc...).

FOAF - Friend Of A Friend

Social info model. In the first place there is the person:

- Name, etc ...
- myersBriggs -> classification into 16 personality types.

```
<foaf: Person>
   <foaf: name> Name Surname </ foaf: name> </
foaf: Person>
```

```
PREFIX foaf: <http://xmlns.com/foaf>
SELECT? Url
FROM <http: //planetdf/blog.rdf>
WHERE {? Contributer foaf: name = ”John Smith”. }
```

SPARQL also allows the management of optional data using the OPTIONAL key (for example the avatar):

```
WHERE {
  OPTIONAL {...}.
}
```

SPARQL is decidable with PSPACE complexity (like sql).

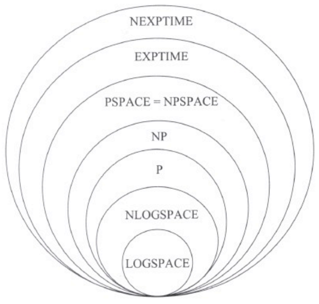

Figure 12: Complexity diagram

How does it behave with OWL?
Do we fix a decidable logic and sacrifice expressiveness or do we give a very expressive logic and sacrifice computability? They have been implemented **both**.

It was decided to divide into 3 OWLs:

- OWL Lite: many restrictions, is decidable and respects a light logic (SHIF), has an EXPTIME (exponential) complexity.
- OWL DL: it is decidable and it respects a logic (SHOIN), it is less flexible, it has some restrictions and it has a NEXPTIME complexity.
- OWL Full: has no restrictions but may not be decidable.

The complexity classes are actually the worst case, what really matters is the average complexity, the response times of the search engines exponentially distort the maximum times, which is why the results are truncated. SPARQL has a PSPACE complexity, therefore low complexity, but if we remove the possibility of options, it comes down to the CO-NP complexity. Most of the web uses low to medium complexity logic, there are rare spikes in PSPACE.
If a question is asked (for example is X true?), The answer is formed by a proof of it and its proof. There are complexities between those who ask the question (us) and those who elaborate the answer (google, web, ...): we have a polynomial complexity, that is, we are not infallible but we can however launch a datum.
With this system we can solve all NP problems using a more powerful system than us. (Arthur Merlin system).

8.5 IP(Untreated)

What if instead of receiving an answer to a question from a search engine, there was a dialogue between the two? It could interact with an intelligent system. IP (Interactive proof system), this system greatly increases the computational power and you pass from an NP complexity to PSPACE, if instead of an intelligent system you use two in parallel the power increases again and you pass from PSPACE to NEXPTIME!
The price for this technology is:

- 100% response is not always guaranteed
- the response time is longer

The main advantage is the processing of much more precise answers. Those mentioned up to now are precise logics:

- OWA open world association
- CWA closed world association

The computer world we normally use is all CWA (programming, database, ...), if you can't find something, it means it's not there, this philosophy has dominated for years, it uses a single manager.
Instead OWA implies that lack of knowledge certainly does not lead to a falsehood! Use different logics from classical logic. OWL is coded in tripplete and we obtain information according to "the glasses we use". Other ways of dealing with the web:

8.6 XQUERY(Untreated)

It is more powerful than XSLT, its power is like that between SPARQL and RDF, besides it is functional and strictly typed.
RDF is XML, so we can use XML to talk about the data and we can even see the data of all web pages, this means that our database is the web!

Xquery gives the possibility to a multiple syntax:

- SQL: hybrid syntax (SQL like) is not XML
- XQUERYX pure syntax in XML

The heart of XQUERY is FLWOR: (For, Get, Where, Order by, Return.
How queries are built:

- at least one between FOR and LET, for loading data, even multiple and combined with each other
- Where and Order by are optional
- return is required

Example

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

returns all the videos of the DB
With xpath instead:

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

There are loops and all sequences are flat, so it is the same to write (1, (2,3), 5,4) and (1,2,3,5,4). Other available commands are CONT and AVG. XQUERY is declarative and functional, so a function can be recombined using other functions.
you can take the previous example and:

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

and in Xpath:

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

XSLT is not functional, therefore not modular!
Furthermore, each sequence can be seen as an array and therefore the operator [] can be used to select parts of it!
Queries written in the two possible ways (XPATH and XML) are not always identical in terms of results, in fact, XPATH eliminates duplicates, instead XML is not !!

FLWOR: LET
It declares a variable and assigns it a definitive value, in fact in a declarative system each variable is a constant that can no longer change its value.

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

Wrong example of using a variable:

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

It is not possible!
This property is called referential transparency, this property is very important, in fact:

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

if on a normal system this instruction cannot be simplified because I don't know if the output values of the two functions are identical, instead in a declarative system since x is certainly the same in both I can replace with:

```
let $ x: = 2 f (x)
```

To calculate for example the duration of all the videos in the database:

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

WHERE
Filter data, example:

```
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)
```

hybrid solution:

```
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 standard SQL if I don't force a sort, the output has an order, instead in XQUERY the order follows nested for loops. Execution is free, only during printing does it align with the order imposed by the for loops:

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

Nested XML where nested brackets activate XQUERY and is not treated as text:

```
for $ v in // video [genre = ”comedy”] return

<actor video = "{$ v / title}"> {// actor
[@ id = $ v / actorRef]} </ actors>
```

DOC (URI)
To identify the data to be used in XQUERY, the DOC (URI) command is used, which receives a URI as input. DOC can identify data from anywhere on the page, but it identifies data from a single large source that can be virtual, made up of multiple physical sources, via the COLLECTION command.

COLLECTION (URI)
It is used to find various physical documents.
The price paid is implementation-dependent, i.e. it depends on the type of implementation!

Comments
(:Comment:)

It is also possible to use XQUERY on RDF by emulating SPARQL XQUERY uses two slightly different characteristics from the normal SQL implementation:

- flat sequences
- broken backwards compatibility for XPATH (elimination of duplicates)

To understand these choices, we need to make a premise, the web is big, very very big. Uncommon problems with normal programming, for the large amount of data, therefore not saving in a single machine. If it is impossible to save them, it is also impossible to process them. The solution adopted is to use several machines in parallel, applying a distributed system, this causes a change in the classical programming logic. The most used and most developed solution is HADOOP, it is used by all the major data managers (Google, Facebook, Twitter).

8.7 HADOOP(Untreated)

HDFS (HADOOP Distributed File System) is the basic component, it imposes a maximum size on the data by dividing it into blocks from 64 MB to 128 MB. It is also fault tolerant that can cause read / write errors, which is why a technique similar to RAID is used, with a much larger amount of data. We have chosen to use a performance policy, at the expense of space. Much like RAID 1, each data is copied 3 times, but the system is made intelligent and aware of a portion of the system around it. It uses the concept of Rack, i.e. each group of machines is a Rack, the data is divided into 2 copies in the same Rack and one in another.
Benefits:

- if a Rack has problems, the data is not lost, because there is always an external copy
- If a copy of the data fails, the copy in the same rack can be used with significant speed advantages
- you can read and write data in parallel by increasing the speed

This complex architecture (WEB3 type) is managed centrally, through the master node. This architecture has a single Achilles heel, if the master node machine falls, the whole system falls. For this in the latest systems, a backup machine of the master node is created.

HADOOP acts on the data with map reduce: it is a functional and parallel technique, the order of the operations does not count, it forces the programming already predisposed to parallelism.
Typical flow:

- I read the data
- MAP (I extract what I need)
- shue and sort
- reduce (aggregate, synthesize and filter)

The power of the following method is while different from the usual programming, full touring.

The data

Each single data is made up of a key, value pair. Data transformation occurs as follows:

Process: map (key, value) create list (new key, intermediate value) reduce: (key, intermediate value list) create value list

When the map processing occurs it can also be repeated and recursive.

In code:

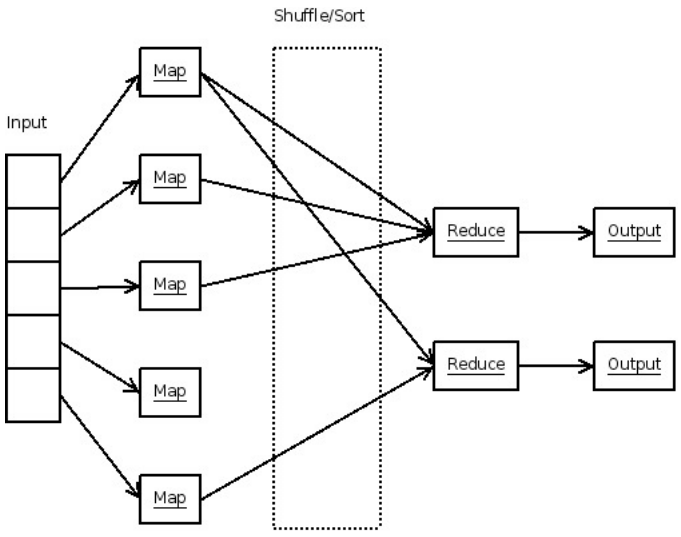

Figure 13: Scheme of map reduce

let map (k, v) = emit (k. toUpper (), v. toUpper ()) the print is:

("Luca rossi "," senior ")
Press
("LUCA ROSSI", "SENIOR")

Parallelism

All the maps are executed in parallel. Stop, all the maps are expected to end, and all the maps are restarted in parallel with the veteran

All based on data locatily: the maps are mainly on nearby machines, to minimize the network trace that very often creates bottlenecks.

Count how much a word is in the database:

Map (key, value)

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

As explained above, all processing is fault tolerant, if for some reason a map fails to produce a result in a timely manner, it is thrown and relaunched. if after a certain number of attempts it does not work, it is eliminated (best efford).
These just described are the core of HADOOP. Applications based on them are:

8.7.1 FLUME (and FLUME NG)(Untreated)

is a system for transporting and aggregating big-data in real time (Weblog, ...) and is based on data flows. Each stream can have very different characteristics, which is subsequently aggregated into nodes, each node has a source and an output.

Example of data entry:

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

It takes the web server log and sends the last piece of it to the node.
The control of FLUME is also centralized with a master. Talk to maps like mobile phones. The external control is done through API managed by thrift
The realtime data arrives from the various sources at intervals according to the needs.
Utility of this system:

- commercial analysis
- realtime customization
- trends
- load and performance monitoring
- error checking

It also uses the fault tolerant, if some problem occurs buffering is activated and if excessive, load shedding is used, i.e. the old data is eliminated (milk).

8.8 HBASE(Untreated)

Also part of the no-sql type, the evolution of the database concept is developed on the basis of big data. The no-sql type are distributed and fault-tolerant solutions and above all they are not ACID. ACID means:

- Atomicity: every transition is atomic, so when a transition is activated it must end and cannot be interrupted
- Consistency: each transition leaves the DB consistent, ie the changes made are immediately updated
- Isolation: multiple transitions cannot interfere with each other
- Durability: When I complete a transition that lasts over time

With no-sql DBs, so not using ACID you have much more flexibility. This new technology is used by the largest DBs, such as Google, Facebook, Amazon, etc ... They also use cached data, so people see the data not in realtime but a few minutes or hours earlier, depending on the databases update times. Amazon for example, when two users try to buy the last book, it allows both of them to buy! Then he will try to get the riodine of the book to arrive as soon as possible!

8.8.1 Brewer's theorem

There is no free meal: if we want Consistency, Availability and fault tolerance we must give up at least one of them. It is therefore impossible to bring the old generation of DB back to the new one. The alternative to ACID is BASE! Basic Availability: the system is available, but not always.
Soft State: information can die, if it is not constantly refreshed, it can return to the original form.
Eventual Consistency: If for a certain timer the DB is not read or modified, the system is updated! But not always!

HBASE is an open source distributed database modeled on Google's BigTable and written in Java. It was developed as part of the Hadoop project. The data structure is in fact that of map reduce, therefore formed by key-value pairs, sorted by key.
A table is a collection of rows-columns: a row
contains a key-value pair each key is primary
and unique
each row value contains n columns
columns can be compressed or defined to stay in memory the value
of each column is called a cell
families of columns: the columns are grouped into families and they are declared when the table is created. Columns and families stand in one

single file called HFILE.
The name of the columns is called the qualifier.

One of the problems of the old sql DB was the obligation to declare NULL values. The new DB type can be seen as the Web adapting to its content. The comparison with the old DB is similar to the differences between matrix and graphs. In HBASE, null values are simply not there and they are not written! Since we write the relationship and not the data in a table.

Each cell has a version part, automatically updated by the system. The system does automatic versioning! Each data is also inserted a time step. The change of a datum does not happen, due to the various distributed problems. A datum therefore is not deleted, but if I update a datum it only adds that I have a new datum in that cell with the updated time step. The old data always remains in memory, until it is deleted after a certain period of time.
How to access the data:
(table, rowkey, family, column) = value
in general: (table, rowkey, family, column, timestamp) = value

At the data structure level: sortedArray (Rowkey, list (sortedArray (column, list (value, timestamp))))

Consistency: access to a single line is atomic, while access to multiple lines at the same time has no guarantee.
When the system grows is auto adaptable, the system at startup inserts all the rows in a region, if the region grows too much the system splits, that is, it cuts the region in two parts by inserting each part in a new region. The same thing, but on the contrary, if the regions get too small, it merges them. The operations take place using WAL (Write AheadLog), it accumulates all the operations and when it can it starts to do them, so this is a point of failure, if the WAL collapses everything collapses.
The tables are "drawn" tall and narrow, because to split it uses the row keys. In order not to have problems with excessive growth of a line, they use small lines containing only a single message, even if much more numerous, but they cannot cause problems of length. The splits work well.

The transition to big data also changes the algorithmic logics, we are moving on to the use of probabilistic algorithms, for example a probabilistic structure is used to know if a certain key exists in the DB. If the key is there, he finds it, but he can make a mistake !! The advantages, however, are greater than the probability of making mistakes, it has a constant reading and writing, moreover it has a much lower need for space than the classic algorithms.

8.9 Linked data

Linked data describe a method of publishing structured data so that it can be interconnected and more useful than semantic web queries. In other words, linked data refers to data published on the web in a machine-readable and interpretable way, the meaning of which is explicitly defined by means of a string consisting of words and markers. In this way, a network of linked data (linked data, in fact) belonging to a domain (which constitutes the starting context) is built, connected in turn to other sets of external data, i.e. outside the domain, in a context of increasingly more relationships. extended.

LOD - Linked Open Data

They are a particular type of open data and collects the open datasets made available on the network. A classification from 1 to 5 stars follows:

1) data available on the web with an open license.
2) data available on the web in a machine readable format.
3) as above, using a non-proprietary data format.
4) as above, semantic web format.
5) as above, with data linked to the data of others to give context.

To mashup and merge the xml, xhtml and html world with the semantic web you need tools like**lifting**(transition from non-semantic web to RDF) and the **lowering**(the other verse).

From sql to RDF:d2rq. We connect to the database via jdbc and convert the data. It also works as a server, providing immediate web access to data seen as RDF. Triplify does similar behavior.

From web pages to RDF: Nlp techniques, natural language processing, we analyze textual or hypertext information and automatically perform the facelift. Examples: Open calais, spotlight, alchemy, etc.

To get the data you need to access the endpoints, this is done through the SPARQL protocol. Usually you do get http by sparql query.
Query: the sparql query that is executed modified by doing the classic% encoding.

DBPedia: Wikipedia in semantic version. It has its own ontology. Provides endpoints for the public. It relies on standard ontologies.

Schema.orgdefines the most important ontologies for web data.

Examples of LODs:

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

To find the LODs we use CKAN which is the reference catalog for the LODs or Swoogle (search engine for the semantic web).

Graph Of Thing: is a tool that allows us to create, manipulate and study graphs. These graphs are objects that describe the relationship between various groups. Google has full support for semantic data.

- Includes RDF;
- Supports all Schema.org anthologies;
- Understands and incorporates all the semantics it uses internally;
- It exposes some of the properties of the**knowledge graph**(the info shown on the right during a search - now also in the search list -).

For example i**Rich Snippests**, the additional information in the search result thanks to the use of semantic information.
Types supported by Rich Snippests:

- Authors: name, photo (who had the photo also had more links so there was a shift in the top 10) and other various info;
- Business & Organizations: for example position on the map;
- Events;
- Music;
- People;
- Products: rating, for example, by Amazon;
- Recipes;
- Reviews: info and rating for example with Tripadvisor;
- Video: image of the video, duration, etc.

Thanks to the semantic web and the knowledge graph it is possible to give the answer directly to the user's questions (in addition to the various links that contain the answer).
Even the existence of concepts themselves are obtained with the knowledge graph. Same thing goes for the refinement of the info and for the related searches. It follows that a user passes more**weather**on Google. Google LOD:

- Google.com/publicdata (in a small way)
- Freebase (in a big way): It's the super container of the semantic base that Google uses.

9 Mobile and Social

9.1 Mobile

The apps are not web but they are interrelated. 52% of the top 1000 sites do not have a mobile site and 25% breach the pattern. Everyone has usability problems.
The evolution is to arrive at the convergence between desktop and mobile. Steve jobs wanted web applications and not apps. But for an economic reason he preferred not to do it.
With HTML5 we arrived at a hybrid app favoring this convergence.

The**app**minimize the access time to the service that interests me.
¼ of users use apps more than 60 times a day. The age group that uses apps the least is between 25 and 35.
The average user uses apps for 86% and the web for
14%. What users do with apps:

- They play 32%
- Social sites 28% (17% from Facebook)

Apart from the games, much of the rest is just using the web via apps.
Apps are all the rage, so it's fair for many developers to devote themselves to them. As a result, there is a lot of competition.

26% of apps are opened only once, 13% 2 times, 9% 3 times and so on. They also have a very low average life: from 4 months to 1 year. Video games paradoxically have an average life of 4 months.
If the growth lasts more than three months, then it will most likely live much longer, otherwise it will die shortly.
Precisely for this reason the classic problem of being found remains.
How are you found? Primarily, it**store**, that is, a search engine. But the rules change to be in the top 10.

ASO - App Search Optimization
It's the SEO of apps. always works on keywords that must be entered in the app:

- Inside the app description;
- Any keywords;
- The name of the app itself (most important factor).

For the rest, the Google and Apple engines use all the information given by the SIS, the overall social system. Downloads, usage time, ratings and reviews, unistalls, brands (inheritance app success ratings of the same company), and the good and good parts that come from other SISs (web and email).

9.2 Mobile Usability

These rules apply to both apps and the web.
Google's mobile compatibilty test.
Spam webmaster: fear, uncertainty, doubt, if you do not follow these rules you will go down from the ranking.
Three basic furniture components:

1. Be mobile
2. Screen size
3. Iteration mode: fingers. (Non-touch cell phones still exist in the world).

Facebook: worldwide target, three mobile versions. m.facebook.com touch.facebook.com 0.facebook.com

1. Be MOBILE
The main problem is the**net**. Switch to 3G and they are 40% slower than desktops. 4G networks are on average 12% slower than desktop ones. So each site pays a corresponding duty in terms of speed and timer increase. It's not just a matter of session or global timers, this is a local delay for each page. And so it happens that the user compares it with the own average times. In the desktop case, the maximum waiting time is 2 seconds. If it goes over, there is a feeling of delay. In the mobile case the **timer remains unchanged**. We must therefore be careful not to exceed these limits. The same limits apply to apps. Responsiveness is essential!

Solutions:
Put a**progress bar**or a so-called spinner (in the desktop case) frustrates the user. The user perceives the time period in a longer way. It shouldn't be done. Therefore, other techniques are used.

- **Transitioning**, keep the user busy with animations or other things.
- Special case of transitioning,**skeleton screen**. If I know the final layout of the action, I can start drawing it even if I don't have the data yet. First Instagram.

- **Preemptivenes**: I try to prevent some actions. Example app that uploads photos, instead of choosing a photo, asking for a description and then uploading it, I upload the photo as soon as it has been chosen.
- 0.facebook.com super low bandwidth version. Very fast and uses only text, however the navigation of the site changes, thus making the user on the one hand happy with the speed but on the other dissatisfied. Version 0 is offered for free in all those places in the world where connections are slow.

2. Screen size
A normal page will struggle to be displayed without scrolling. How bad is the scroll? Horizontal scrolling is much worse than vertical scrolling. Vertical scrolling is often not that bad. The physical and mental effort is less because they are used**gesture**. However, it is detrimental to users when used to offer choices (the user must bear in mind the hidden part). Minimize vertical scrolling and thus avoid images, for example. They justify themselves in the selection lists only when it comes to final products. The effect does not work either: text and small images, creates the effect I see I do not see.
Icons can be used even if the user still prefers text.
However, if you want to use them without adjacent text, two principles must be respected:

- Explainability: holding down the icon you get the textual information about its action.
- Escapability: still being able to avoid the action by moving out of the area with your finger.

Furthermore, having small screen sizes, certain invasive elements such as advertising must be redefined.
IAB: interactive advertising bureau, body that manages the size of banners.
Interstitial ads: takes up the whole screen and has a button to close. **Smart banners**: The size fits according to the width of the screen. They are the best but they work very badly if they are implemented in a fixed position (creates limitations and inconvenience to users). Even worse if the banner changes.

The Smart app banner, i.e. advertising the app version of the site, has the same negative effect as popups, better write it in the content of the page.

3. Means of interaction: the fingers
The**drag**it does not create muscular efforts, which is why it facilitates the use of gestures, hence the use of**swype**.
However, the principle of**Untiming**, actions resulting from pressure should not depend on the duration, but only distinguishes the tap and drag. Not respecting untiming causes stress.
The main problem is that the fingers are very large (**fat finger**) and we cannot click precisely (the middle finger is wide on average: 11mm, 8mm for children and can go up to 19mm). Moral: a clickable area must be very large. There**minimum size**of a clickable area must be 7x7 mm (maximum 5x5 at the expense of losing 20% precision). And around at least one area of**padding**of 2mm. The ideal is 9x9. More than 83% of the top 1000 sites do not respect these rules.
Reversibility: every action that takes place should be reversible, especially in situations of potential error due.

There**Fitts law**changes in the sense that the size of the object matters but now the imprecision of the finger and the grip of the device also count (there are 5 phases and each has a different thermography).
Most of them use thumbs, which makes accuracy worse, you have to increase the sizes by 2 mm.
On mobile the window coincides with the edge of the device so there are magical areas such as the edges. We can position ourselves i**fan menu**. The pie menus are not the best (the finger covers some parts of the cake).

9.3 Social Network

March 7-13, 2010 Facebook overtakes Google.
Three fundamental aspects:

- TIME
- TRUST
- TRANSMISSION

Social networks have a**weather**of very high absorption, Facebook alone absorbs all the other social networks (FB 53%, Yahoo 17.2%, Google 12.5%). Information on a social network is more effective than on a normal website (**trust**). It is more effective by orders of magnitude 10x to 100x. **Astroturfing**: I create comments or information on social networks that seem true but in reality I don't know. In 2013, bot traffic surpassed human traffic.
There**virality**it is a powerful tool but it needs the right conditions:

- Social Currency: makes those who share it special, gives them a positive light;
- Practical Value: practical value associated with the content;
- Storytelling: creating curiosity with a story that attracts;
- Emotion: a message must bring emotions. Positive emotions are more viral than negative ones (FB doesn't have the "I don't like" button).

10) Excitement
9) Affection
8) Hope
7) Joy
6) Pleasure
5) Delight
4) Happiness
3) Surprise
2) Interest
1) Fun