UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Accessibilità delle applicazioni web che utilizzano Ajax

Candidato: *Valentina Casciola*

Relatore: *Fabio Paternò*

Correlatore: *Mirko Tavosanis*

Anno Accademico 2008-2009

Sommario

Introduzione

Il seguente lavoro di tesi si occupa dello studio dell'accessibilità delle applicazioni web che utilizzano la tecnica di sviluppo Ajax.

Nella prima parte della relazione viene studiato il funzionamento di Ajax e il suo ruolo all'interno del web ed, in particolare, nel web 2.0.

Nella seconda parte viene definito il concetto di accessibilità, vengono presentate le leggi che la riguardano e le principali linee guida da seguire.

La terza parte affronta più nello specifico il tema dell'Ajax accessibile in quanto vengono presentate tutte le soluzioni per progettare un Ajax accessibile, tenendo in considerazione soprattutto l'uso di WAI-ARIA.

In relazione agli argomenti trattati è stato riprogettato e implementato un sito web museale: il sito del museo del marmo e dei beni culturali di Carrara. Nell'applicazione sono stati inseriti script Ajax per migliorare e arricchire le funzionalità, cercando di mantenere l'accessibilità utilizzando gli attributi WAI-ARIA.

1 Prima parte: Ajax

1.1 Web 2.0

> *"Sul Web dovremmo essere in grado non solo di trovare ogni tipo di documento, ma anche di crearne, e facilmente. Non solo di seguire i link, ma di crearli, tra ogni genere di media. Non solo di interagire con gli altri, ma di creare con gli altri. L'intercreatività vuol dire fare insieme cose o risolvere insieme problemi.*
> *Se l'interattività non significa soltanto stare seduti passivamente davanti a uno schermo, allora l'intercreatività non significa solo starsene seduti di fronte a qualcosa di interattivo."*
> Tim Berners-Lee,
> L'architettura del nuovo Web

1.1.1 Aspetti del Web 2.0

La prima cosa da chiarire è che Ajax e Web 2.0 non sono due termini con lo stesso significato. Ajax è una tecnica di programmazione, il Web 2.0 è l'utilizzo di tutte le migliori tecnologie esistenti per creare esperienze nuove. L'Ajax è quindi solo una delle tecnologie del web 2.0.

Figura 1: piramide del web 2.0

Il web 2.0 è un concetto molto complesso che quindi non è facile definire. Conviene analizzarlo tenendo conto di una serie di tratti comuni che possono essere categorizzati in tre dimensioni: sociale, contenutistica e tecnologica.

Si parla di aspetto sociale in quanto le applicazioni web 2.0 offrono uno strumento di relazione attraverso la condivisione di contenuti e proprio a seconda della natura di questa relazione possiamo valutare la "socialità" del servizio. Per avere successo online serve creare, mantenere e stimolare la partecipazione. I principi fondanti sono dunque la fiducia, la credibilità e il controllo sociale.

Riguardo ai contenuti si può notare che le applicazioni web 2.0 sono caratterizzate da un processo di riuso e aggregazione delle informazioni. Criteri fondamentali sono quindi l'ordine e l'aggregazione, che è la raccolta dei contenuti attraverso dei particolari servizi – gli "aggregatori" – in grado di assemblare in uno spazio unico materiale che proviene da fonti diverse. I tag sono appunto le etichette che vengono apposte ai contenuti, caratterizzandoli per categorie e per parole chiave, per fare in modo che i contenuti siano correlabili e utilizzabili per parametri semantici qualitativi e non quantitativi.

Dal punto di vista tecnico l'evoluzione si concentra sullo sviluppo di interfacce sempre più simili a quelle dei programmi desktop, sull'utilizzo di tecnologie innovative quali Ajax e Rss e sull'impiego di API. Il web sta diventando un ambiente sempre più dinamico, in cui le informazioni non sono più statiche ma tendono a mutare continuamente. [19]

1.1.2 Come nasce il termine web 2.0

Figura 2: Tim O'Reilly

Il concetto di web 2.0 nasce nel 2004 in una sessione di brainstorming durante una conferenza tra l'editore O'Reilly e MediaLive International. Qui il vice-presidente Dale Dougherty ha fatto notare quanto fosse diventata importante la rete e quante applicazioni nuove e interessanti dotate di caratteristiche comuni si stavano sviluppando. Questa svolta prende il nome di web 2.0 ed è così che da questa analisi sono nate una serie di conferenze sul web 2.0. Da questo momento in poi il termine si è diffuso enormemente ma c'è ancora disaccordo sulla sua definizione precisa. [1]

1.1.3 Dal web 1.0 al web 2.0

Il web è stato concepito per visualizzare documenti ipertestuali statici. E' questa la definizione del web 1.0. Il 6 agosto 1991 viene pubblicata la prima pagina web della storia all'indirizzo http://info.cern.ch/hypertext/WWW/TheProject.html .

Un'altra data importante è il 14 gennaio 1997, giorno in cui viene pubblicata la specifica del W3C HTML 3.2. Creata da una collaborazione tra aziende leader del settore (Microsoft, IBM, Netscape

Corporation, SUN Microsystems) aggiunge al web elementi fondamentali come le tabelle, gli applet e il testo attorno alle immagini. Ma è solo l'inizio della rivoluzione.

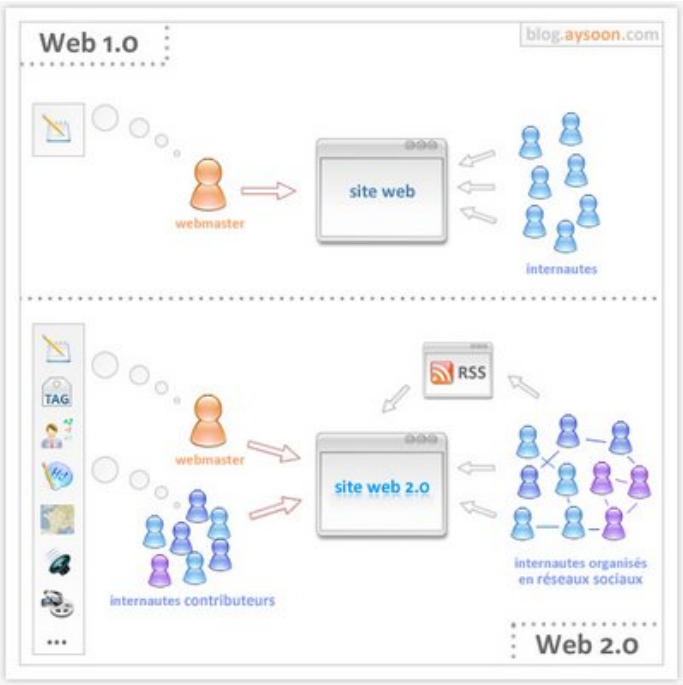


Figura 3 differenze visuali tra web 1.0 e web 2.0

Si parla di web 1.5 quando il web comincia a diventare più dinamico. L'utilizzo di database e di linguaggi lato server permette di creare applicazioni che garantiscono una grande interattività con l'utente, arrivando a creare delle vere e proprie applicazioni web.

Si parla di web 2.0 quando il web diventa un vero e proprio ambiente operativo. Nel corso di questa evoluzione comunque la struttura è rimasta invariata: alla base del web c'era e rimane sempre l'ipertesto. Quello che cambia è il differente approccio al web degli utenti. [2]

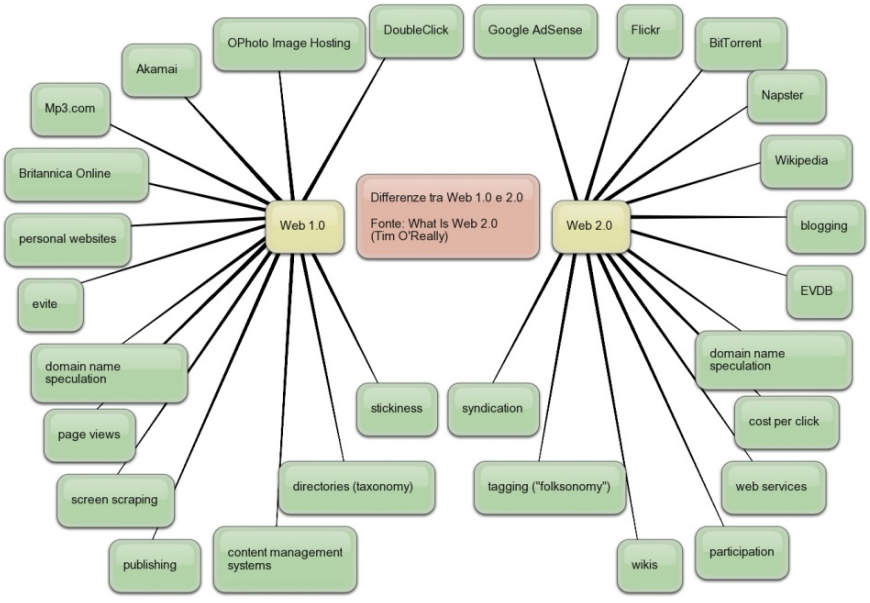


Figura 4 differenze tra web 1.0 e web 2.0

1.1.4 Verso il web 3.0

Esistono molte discussioni in corso sulla definizione del web 3.0, la nuova fase di evoluzione del web. Il termine è quindi materia di grande interesse e dibattito.

L’idea è che in questa fase le informazioni prodotte dai nuovi utenti devono poter interagire tra loro, automaticamente. Per fare in modo che le informazioni pubblicate online possano essere interpretate e condivise dai computer, esse devono essere presentate con una nuova struttura di formattazione per le informazioni. E questa nuova forma può essere quella del web semantico proposta da Tim Berners-Lee. Sfruttando le costruzioni semantiche, infatti, è possibile stabilire relazioni significative fra risorse differenti. [4]

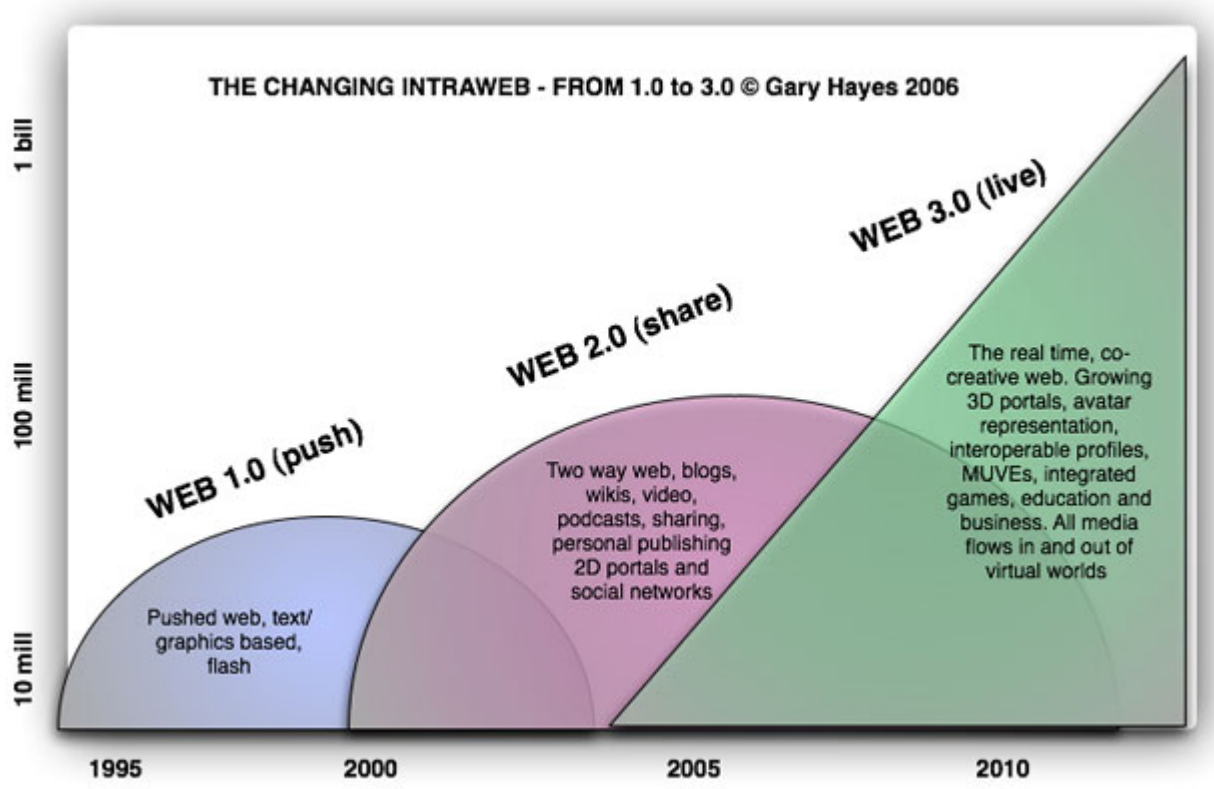


Figura 5: il passaggio dal web 1.0 al web 3.0

1.1.5 Web 2.0 come piattaforma

L’inventore del termine, O’Reilly, ha tentato di dare una definizione compatta del termine:

“Web 2.0 is the business devolution in the computer industry caused by the move to the internet as platform, and an attempt to understand the rules for success on the new platform. Chief among those rules is this: Build applications that harness network effects to get better the more people use them.”

Il web è visto come un piattaforma da sfruttare il più possibile. L'uso di applicazioni web porta ad una sempre maggiore presenza continuativa online degli utenti. Cambia il loro atteggiamento, non è più passivo. Sono gli utenti che fanno il web, non si limitano ad usarlo. [3]

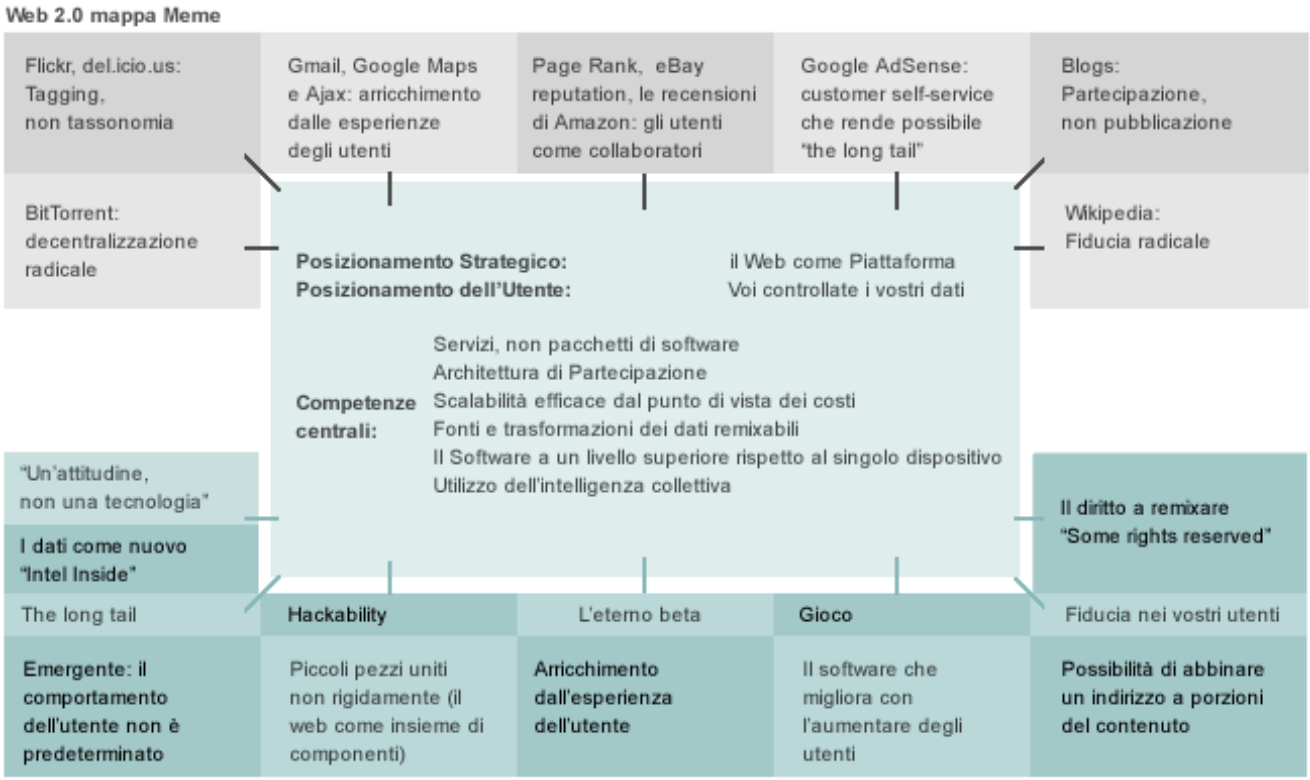


Figura 6: mappa Meme del web 2.0 [3]

1.1.6 Mappa del web 2.0

Varie mappe visuali del web 2.0 sono state realizzate per rendere più chiari tutti gli elementi che caratterizzano questo stato evolutivo del web. Sono tanti i termini presenti , a dimostrazione del fatto che il concetto del web 2.0 è molto complesso e pieno di sfaccettature. I punti cardine sono partecipazione, remixability e usabilità. [2]

Figura 7: esempi di mappe web 2.0 [2]

1.1.7 Partecipazione (Blog, WIKI, Folksonomia)

L'idea di partenza è che il web deve essere lo strumento per permettere agli utenti di partecipare e condividere informazioni tra loro. Gli utenti diventano dei co-sviluppatori che operano in un ambiente aperto, aggiornato di continuo, in uno stato di "beta perpetuo". Un monitoraggio continuo del comportamento degli utenti è necessario per verificare quali sono le funzioni più utilizzate. Per facilitare la collaborazione bisogna aiutare l'utente rendendo l'applicazione web 2.0 usabile e immediata, dando strutture predefinite da modificare e fornendo meccanismi di valutazione.

Uno degli aspetti più pubblicizzati del web 2.0 è lo sviluppo dei blog, siti che consistono in diari online dove gli interventi sono visualizzati in ordine cronologico inverso e dove è possibile inserire commenti. Chiunque può creare un blog con pochissimi passi. E' evidente quindi che i blogger, essendo i linker più prolifici e tempestivi, hanno un ruolo sproporzionato nel dare forma ai risultati dei motori di ricerca. Poi la comunità dei blogger è auto-referenziale: quelli che prestano attenzione agli altri blogger aumentano la loro visibilità e quindi il loro potere. Il blog quindi sfrutta l'intelligenza collettiva non semplicemente come un amplificatore ma come una specie di filtro. Questa attenzione collettiva della blogosfera decide quindi cosa è importante e cosa no creando una "saggezza del folle" dove ogni individuo aggiunge o seleziona il valore.

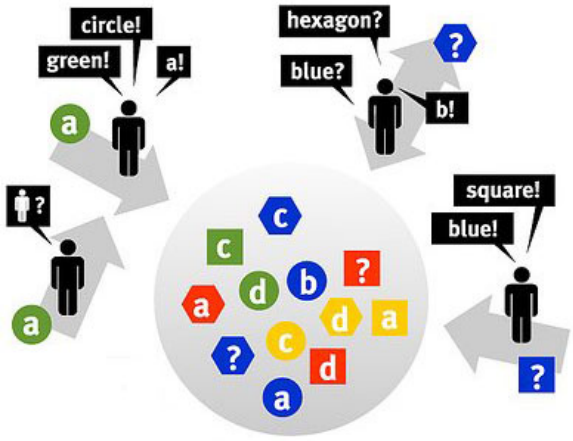
Il Wiki è un software che permette agli utilizzatori di aggiungere contenuti ma anche di modificare contenuti inseriti da altri utenti. E' chiaro quindi che si tratta di un sistema altamente collaborativo. Il successo di questo sistema è dovuto quindi a vari fattori:

- facilità con cui una informazione può essere modificata o cancellata;
- gli utenti più validi lasciano una traccia più evidente rispetto agli altri;

- il sistema non è in tempo reale quindi l'utente ha tempo di ragionare e informarsi per bene prima di apportare delle modifiche;
- è accessibile ovunque in quanto non necessita l'istallazione di alcun software.

Il caso più famoso è Wikipedia, enciclopedia libera con contenuti utilizzabili e ridistribuibili liberamente, realizzata con il contributo di tutti quelli che desiderano migliorarla (utenti registrati e non) e che utilizza software libero e formati aperti.

Il termine folksonomia è un neologismo creato da Thomas Vander Wal ed è formato dall'unione di due parole: folks (gente) e tassonomia (categorizzazione). La folksonomia è quindi una tassonomia creata da chi la usa. Nel web 2.0 si definisce in questo modo la metodologia utilizzata da un gruppo di persone che collaborano spontaneamente per organizzare in categorie tutte le informazioni disponibili. In un contesto altamente collaborativo come quello del web 2.0 una struttura di classificazione predefinita rischia di limitare troppo. La folksonomia permette invece di fornire un'informazione che si adatta al modello concettuale degli utenti in quanto sono gli utenti stessi di solito ad organizzare l'informazione. I problemi principali sono dunque l'ambiguità dei tag utilizzati e la mancanza di un insieme comune di parole chiave a fronte di un sistema di distribuzione rapido ed efficace nel momento in cui la pratica è condivisa da comunità in crescita.



Flickr e Delicious sono esempi di applicazioni web che puntano molto sulla folksonomia. Si tratta di servizi che stanno rivoluzionando il modo di usare il web in quanto la classificazione popolare è vista da molti come la futura concretizzazione del web semantico. In un sistema in crescita costante infatti la folksonomia sembra al momento il metodo migliore per ordinare i dati disponibili. La folksonomia ci aiuta a condividere con gli altri la nostra visione del mondo senza costringersi entro una categorizzazione assoluta, cosa che in un contesto collaborativo vuol dire avere la possibilità di scoprire significati e differenze.

[19] [2]

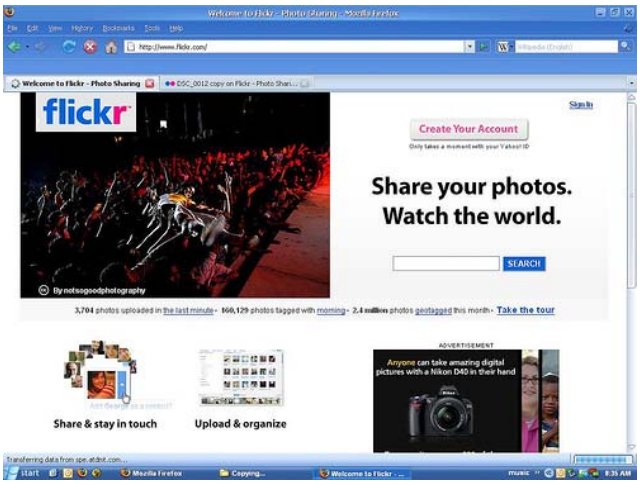


Figura 7: pagina di Flickr

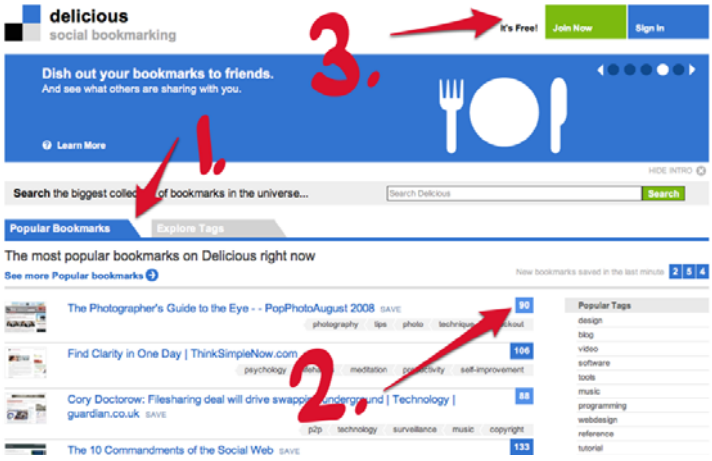


Figura 8: pagina di delicio.us

1.1.8 Remixability (RSS, API)

La remixability nasce dalla volontà dell'utente e degli sviluppatori di poter condividere ed utilizzare le informazioni per poi ricomporle, elaborarle e modificare per dare vita a nuovi concetti e idee.

Figura 9: icona RSS

RSS, acronimo di Really Simple Syndication, è uno dei più popolari formati per l'esportazione di contenuti sul web. E' uno standard de facto che permette di distribuire con grande facilità i contenuti di un sito e presentarli sotto altre forme. Il formato viene usato per la prima volta da Netscape per pubblicare una serie di notizie e poi viene adoperato anche dalle comunità dei blogger, così da permettere l'esportazione dei post. E' la popolarità dei blog dunque a segnare il successo di RSS: i weblog cominciano a produrre contenuti in RSS e iniziano a proliferare i blog aggregatori, siti che raccolgono una selezione di post dei blog più letti, e gli RSS reader, programmi

per fruire i contenuti di un blog tramite desktop o altri dispositivi. Il formato RSS si basa su XML, da cui ha ereditato la semplicità, l'estensibilità e la flessibilità, ed è quindi un suo "sottolinguaggio" studiato con lo scopo preciso di rappresentare una forma semplificata di diffusione di contenuti nel web. Il vantaggio di RSS è che ci si può aggiornare rapidamente da più fonti senza dover navigare da sito a sito ma, semplicemente, verificando la lista dei contenuti pubblicati, il feed RSS. RSS definisce una struttura adatta a contenere un insieme di notizie, organizzate in vari campi. Quando si pubblicano notizie in formato RSS, la struttura viene aggiornata con i nuovi dati. Per leggere un feed serve un lettore di feed, strumento in grado di tradurre la struttura in contenuti fruibili da parte dell'utente. Un formato di syndication alternativo al RSS è Atom, anche questo basato su XML.

Si definiscono API (Application Programming Interface) ogni insieme di procedure disponibili al programmatore per portare a termine un determinato compito. Sul web sono tantissimi i vantaggi che comportano le API: utilizzando le API di siti di terze parti è possibile creare servizi innovativi anche da parte di persone che non sono programmatori di professione. Il risultato si definisce mash-up, applicazione che usa contenuto da più sorgenti per creare un servizio completamente nuovo. [2]

1.1.9 Usabilità e web 2.0

L'usabilità è definita come l'efficacia, l'efficienza e la soddisfazione con le quali un certo gruppo di utenti raggiunge degli obiettivi in determinati contesti.

Figura 10: Jacob Nielsen

Jacob Nielsen ha criticato il web 2.0, colpevole di concentrarsi solamente sulla grafica, tralasciando accessibilità e usabilità. Le nuove applicazioni si preoccupano troppo di fornire strumenti di personalizzazione e condivisione senza curarsi tanto delle regole base del web-design. I nuovi concetti di comunità e di condivisione di contenuti non sono sbagliati ma vanno sviluppati mantenendo il sito usabile.

Per altri, invece, l'usabilità è un concetto che prescinde dalla versione del web, ma è in stretta relazione al sito che stiamo esaminando. Il web facile da usare, leggibile, comprensibile e sobrio non è che un buon web 2.0. [2]

1.1.10 Progettazione continua

Le applicazioni del web 2.0 non necessitano di istallazione ma sono tipicamente ospitate nel server di chi le offre. Questo rende possibile un lavoro continuo di evoluzione e di revisione di queste

applicazioni attraverso l'analisi dei problemi e le richieste degli utenti. Tutto questo avviene in maniera graduale, senza rivoluzionare l'ambiente. E' un metodo non invasivo (l'utente infatti può continuare ad utilizzare la vecchia funzionalità, fino a che non decide di passare alla nuova) ma che ha il difetto di non rendere sempre l'utente pienamente consapevole delle funzionalità presenti nell'applicazione. Le modifiche di solito non alterano il layout e sono integrate nella grafica per mezzo di gradienti molto leggeri: sono innovazioni discrete che vengono introdotte gradualmente quando si rendono necessarie. E' il cosiddetto modello progettuale del perpetual beta, cioè quello per cui le applicazioni rimangono perennemente in versione provvisoria, ed è sicuramente l'approccio migliore per lo sviluppo di interfacce innovative in quanto permette un miglioramento graduale dell'interfaccia , in base anche a test o suggerimenti dati dagli utenti stessi, velocizzando il processo. Non avremo più un'applicazione 1.0 che passa dopo molti mesi alla versione 2.0 in seguito alle segnalazioni degli utenti. L'applicazione ora diventa una sorta di cantiere aperto revisionato quotidianamente, sempre garantendo la piena funzionalità.

1.1.11 Rivoluzione grafica del web 2.0

Il web 2.0 ha portato anche ad un'innovazione dello stile grafico delle applicazioni web. Un aspetto fondamentale è la semplicità, intesa non come minimalismo ma come mettere a disposizione dell'utente tutto il necessario, senza eccedere, allo scopo di catturare meglio la sua attenzione e aiutarlo nella ricerca delle informazioni. La distinzione tra la testata della pagina e il contenuto principale è netta e il logo dell'applicazione è facilmente riconoscibile, in quanto tutti i loghi utilizzati dalle varie applicazioni web 2.0 hanno denominatori comuni come grandezza, spessore e colore. Rispetto al passato c'è un utilizzo di font più grandi in modo da dare più enfasi all'identità del sito e i colori forti vengono sfruttati per dividere le varie sezioni. L'utilizzo di superfici arricchite con effetti 3D e gradienti è importante in quanto rende possibile l'associazione degli oggetti presenti nel sito a qualcosa di reale. [2]

Figura 11: icone web 2.0

1.1.12 Web 2.0 e Accessibilità

Spesso le applicazioni web 2.0 non sono pensate per l'accessibilità. Infatti non possiamo dire semplicemente che un sito web è accessibile quando ogni gruppo di utenti può accedervi ma serve che, le persone adulte in particolare, abbiano la volontà di accedere al sito. Il web 2.0 nasce per essere usato nella vita di tutti i giorni. L'accessibilità fisica non è che il punto di partenza, e il web 2.0 è accessibile solo se i gruppi potenzialmente esclusi vogliono usare e traggono beneficio da esso. [1]

1.1.13 Web 2.0 e Ajax

L'Ajax è una componente tecnologia fondamentale del web 2.0. Con Ajax infatti possiamo dare alle applicazioni web la velocità e l'interattività delle tradizionali applicazioni desktop. Non si tratta di una novità ma l'acronimo è stato coniato per riassumere l'utilizzo congiunto di tecniche già esistenti e ampliamente utilizzate. L'Ajax mette dunque in evidenza due caratteristiche molto potenti, in circolazione da tempo, ma poco utilizzate: fare richieste al server senza ricaricare la pagina e analizzare e lavorare con documenti XML. [1]

1.2 Cosa è Ajax?

> *"The biggest challenges in creating Ajax applications are not technical. The core Ajax technologies are mature, stable, and well understood. Instead, the challenges are for the designers of these applications: to forget what we think we know about the limitations of the Web, and begin to imagine a wider, richer range of possibilities."*
>
> *Jesse James Garrett, Ajax: A new approach to Web Applications.*

1.2.1 Il significato dell'acronimo Ajax

Figura 12: Jesse James Garrett

Il termine Ajax compare per la prima nel febbraio del 2005. In quel periodo Adaptive Path stava investendo molto sullo sviluppo delle applicazioni web. Jesse James Garrett, presidente e cofondatore della Adaptive Path, infatti scrive un articolo dove spiega questo nuovo concetto che sta cambiando il modo di percepire il web. Ajax non è nuova tecnologia. In realtà è un insieme di tecnologie (già esistenti) che, unendosi ad altre, creano dei sistemi nuovi e potenti. Il nome è l'abbreviazione di Asynchronous Javascript XML , termine coniato per mettere in evidenza le caratteristiche principali di questa tecnica di sviluppo:

- **A** come **Asincronia**. Le richieste Ajax tra server e client sono asincrone: il browser invia una richiesta ad un server web e non attende la risposta. Nel modello tradizionale di comunicazione fra un browser ed un server web l'utente invia una richiesta al server ed attende. Il server recupera le informazioni ed invia una riposta. Questo modello di interazione si dice quindi sincrono. Nel modello asincrono, invece, l'utente invia una richiesta e non attende la risposta. Il server quindi può gestire più richieste contemporaneamente e il browser gestisce le risposte non appena arrivano dal server web.

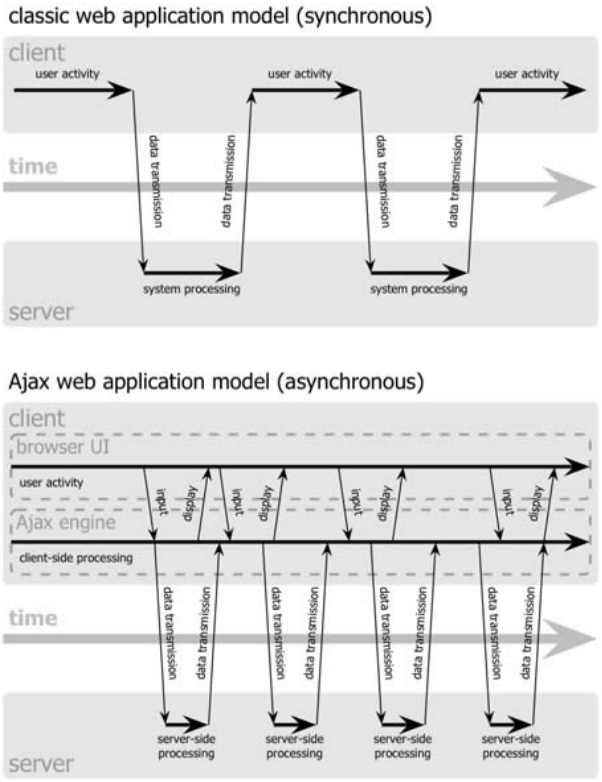


Figura 13 immagine del modello di un'applicazione web [12]

- **J** come **JavaScript**. JavaScript è un linguaggio di scripting orientato agli oggetti e comunemente usato nei siti web. Sviluppato inizialmente da Netscape Communications è stato poi formalizzato con una sintassi più vicina a quella del linguaggio Java di Sun Microsystems. Ajax usa l'oggetto JavaScript XMLHttpRequest per gestire il flusso di dati tra client e server senza dover ricaricare l'intera pagina, indirizzando la risposta del server su uno o più elementi della stessa pagina.

- **X** come **XML**. XML, acronimo di eXtensible Markup Language, è un metalinguaggio creato e gestito dal W3C. Nato nel 1998 come adattamento dell'SGM (Standard Generalized Markup Language), permette di definire la grammatica di diversi linguaggi specifici derivati. I browser hanno modi incorporati per trattare le informazioni che sono formattate in XML e quindi è molto vantaggioso usare l'XML come formato per la condivisione delle informazioni fra browser e server web.

  Quello che fa il termine Ajax non è altro che descrivere due funzionalità presenti da anni nei browser ma da sempre sottovalutate:

- Eseguire richieste al server senza ricaricare la pagina;

- Interpretare e lavorare con documenti XML.

Grazie alla tecnica di sviluppo Ajax possiamo quindi creare pagine web che rispondono in maniera più rapida, grazie allo scambio in background di piccoli pacchetti di dati con il server, così che l'intera pagina non deve essere ricaricata ogni volta che l'utente effettua una modifica. E' una tecnica che migliora l'interattività, la velocità e l'usabilità di una pagina web.

Utilizzare questa tecnica significa utilizzare una combinazione di:

- HTML (o XHTML) e CSS per il markup e lo stile;
- DOM (Document Object Model) manipolato attraverso un linguaggio ECMAScript come JavaScript, per mostrare le informazioni ed interagirvi;
- Oggetto XMLHttpRequest per l'interscambio asincrono dei dati tra il browser dell'utente e il web server;
- XML (di solito usato come formato di scambio dei dati, anche se qualunque formato può essere utilizzato, inclusi testo semplice e HTML preformattato)

Le applicazioni web che utilizzano Ajax richiedono browser che supportano le tecnologie necessarie. Questi browser includono Mozilla, Firefox, Opera, Konqueror, Safari e Internet Explorer. Tuttavia, per specifica, Opera non supporta la formattazione di oggetti XLS. [11]

1.2.2 La storia: da DHTML a XHR

La tecnica che permette di interagire dinamicamente con il documento attraverso la manipolazione del DOM è già presente nel 1998 ed era chiamata DHTML (Dynamic HTML). L'obiettivo di questo insieme di tecnologie è quello di rendere dinamica una pagina statica, sfruttando lo scripting lato client, cioè alterando dinamicamente un elemento della pagina senza interagire con il server.

Il passo successivo è stato negli anni 1996 e 1997 con l'introduzione, rispettivamente, del tag IFRAME in Internet Explorer 3 e del tag LAYER da parte di Netscape in Navigator 4, che permettono l'inserimento di pagine o JavaScript esterni.

Microsoft ha continuato a lavorare su questa tecnica fino alla realizzazione di un oggetto ActiveX chiamato XMLHTTP, presente la prima volta in Internet Explorer 4.

L'esempio di Microsoft è stato seguito in altri browser, tra cui Mozilla e Safari, che hanno messo a disposizione un oggetto JavaScript con le stesse funzionalità e la stessa interfaccia (API). [11]

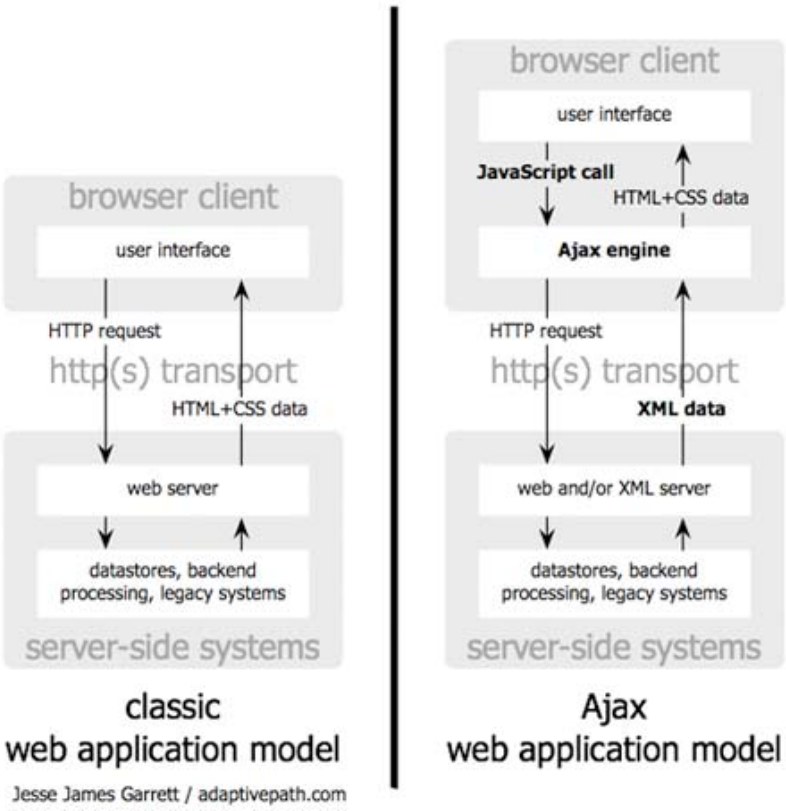
1.2.3 Ajax: un nuovo approccio alle applicazioni web

"Ajax: a new approach to web applications" è il titolo del post scritto e pubblicato da Jesse James Garrett di Adaptive Path per che introduce per la prima volta il termine Ajax. Vediamo quali sono i punti salienti di questa pubblicazione.

"Le applicazioni desktop hanno una ricchezza e una reattività che sembravano fuori portata del web. La stessa semplicità che ha permesso la proliferazione rapida del Web crea anche un gap tra le esperienze che possiamo fornire e le esperienze che gli utilizzatori possono ottenere dall'applicazione desktop." Una tecnica di progettazione innovativa come Ajax diventa necessaria per fornire alle applicazioni web la velocità e l'interattività delle applicazioni desktop e questo comporta dei vantaggi enormi e garantisce una maggior usabilità delle applicazioni web. E' da questa necessità di "imitare" le applicazioni desktop che dobbiamo partire per ogni discorso su Ajax e sul perché è utile questa tecnica.

"Ajax non è una tecnologia. In realtà è più tecnologie, ognuna si sviluppa per conto proprio, ma unendosi con le altre crea nuovi e potenti sistemi." Un'altra cosa che Garrett chiarisce subito è che Ajax non è una tecnologia nuova ma solo un nuovo modo di utilizzare tecnologie già esistenti. Quello che è nuovo è il modo di progettare e di usare l'interfaccia web.

"Un'applicazione Ajax elimina la natura start-stop-start-stop dell'interazione sul web introducendo un intermediario, un motore Ajax, tra l'utente ed il server." Con Ajax la comunicazione tra client e server diventa asincrona. L'utente non deve più aspettare la risposta del server per fare una nuova richiesta: l'interazione con l'applicazione può continuare tranquillamente. Una volta che il server è pronto ad inviare i dati, questi vengono aggiornati senza la necessità di ricaricare l'intera pagina.



"Le più grandi sfide nella creazione delle applicazioni Ajax non sono tecniche. Le tecnologie principali di Ajax sono mature, stabili e ben strutturate. Invece, le sfide sono per coloro che progettano queste applicazioni: per dimenticare ciò che pensiamo di sapere sui limiti del web, e iniziare ad immaginare una serie più ampia e ricca di possibilità." Garrett conclude dicendo che la tecnologia che sta dietro ad Ajax è abbastanza matura. Quello che c'è da fare è quindi capire quali sono le applicazioni per cui Ajax è più adatto, non è un caso se importanti società, come Google, stanno investendo moltissimo su questa tecnologia. [12]

1.2.4 Come nasce l'oggetto XMLHttpRequest

XMLHTTP è un set di API usate da linguaggi di scripting come JavaScript per trasferire dati come XML da ed a un web server tramite http ed è quindi un componente fondamentale nella tecnica di sviluppo web Ajax. Inventato nel 1999 da Microsoft e usato da Internet Explorer 5.0 come oggetto ActiveX, è stato poi implementato con una versione compatibile nel 2002 da Mozilla, poi seguita a ruota dagli altri produttori di browser come Opera e Safari. [11]

1.2.4.1 Metodi dell'oggetto XMLHttpRequest

abort()	Cancella la richiesta corrente.

getAllResponseHeaders()	Restituisce sotto forma di stringa tutte le intestazioni di risposta http.
getResponseHeader(header)	Restituisce solo il valore dell'intestazione specificata.
open(method, URL, asynch)	La richiesta http ha tre parametri: Method specifica il tipo della richiesta (GET quando si richiedono dei dati e POST quando si inviano dei dati); URL può essere un parametro sia relativo che assoluto e specifica l'indirizzo URL; Asynch è un valore booleano che specifica se la richiesta deve essere gestita in modo asincrono (true) o in modo sincrono (false).
send(content)	Iniva la richiesta al server. E' il metodo che di fatto avvia l'operazione. Se il metodo è di tipo GET il parametro content è impostato a null, altrimenti contiene i dati che sono inviati al server.
setRequestHeader(name,value)	Aggiunge una coppia di valori chiave/valore alle intestazioni HTTP da inviare.

1.2.4.2 Proprietà dell'oggetto XMLHttpRequest

readyState	Quando viene creato l'oggetto request il valore di questa proprietà cambia. Gli stati di questa proprietà sono 4: Uninitialized (0) →non inizializzato: l'oggetto è stato creato ma non gli è stata comunicata la richiesta: open() non è stata chiamata; Loading(1) → aperto: l'oggetto sa della richiesta ma non è stata ancora inviata: send() non è stata chiamata; Loaded(2) → richiesta inviata:la richiesta è stata inviata e sono disponibili informazioni di base sulla risposta; Interactive(3) → risposta in ricezione: la risposta è stata caricata nell'oggetto request ;

	Completed(4) → risposta ricevuta: l'intera risposta è stata caricata nell'oggetto request ed ora è disponibile.
responseText	Restituisce una stringa che contiene la risposta dal server.
responseXML	Restituisce un oggetto documento DOM che contiene la risposta del server in formato XML che può essere interpretato con i metodi e le proprietà degli oggetti dell'albero DOM, standard del W3C.
status	E' il codice secondo lo standard http relativo all'esito dell'operazione. Il codice 200 indica che l'operazione http si è conclusa correttamente.
statusText	Restituisce una stringa contenente lo status.

[5] [11]

1.2.4.3 Evento dell'oggetto XMLHttpRequest

L'oggetto XMLHttpRequest possiede un solo evento chiamato **onreadystatechange**. Questo evento specifica una funzione di callback che viene richiamata quando cambia il valore della proprietà readyState. Tipicamente viene impostata una funzione che elabora i dati ricevuti dal server per poi effettuare delle modifiche all'interfaccia utente. [11]

1.2.5 Formati per i dati: XML e JSON

Anche se la X in Ajax sta per XML, il formato dei dati restituiti dall'oggetto XMLHttpRequest non deve essere per forza XML. I dati possono essere testo normale o altri formati come JSON (JavaScript Object Notation).

JSON è una notazione standard per lo scambio di dati basato su un sottoinsieme di regole sintattiche di JavaScript. Il vantaggio è dunque quello di poter trasferire direttamente in variabili JavaScript il suo contenuto tramite l'uso della funzione eval(). Il grosso svantaggio è invece quello di essere ancora troppo limitato nella tipologia di informazione che è in grado di rappresentare.

Il linguaggio principe per inviare dati strutturati è certamente XML. L'oggetto XHR offre un oggetto DOM con cui il client è in grado di navigare gli elementi del file XML restituito dal server.

I dati in formato XML possono essere recuperati tramite la proprietà responseXML, che permette di usare le API DOM per ottenere informazioni di interesse:

```
strData = objXHR.responseText;

objXML = objXHR.responseXML.documentElement;
```

JSON e altri formati possono essere recuperati tramite la proprietà responseText:

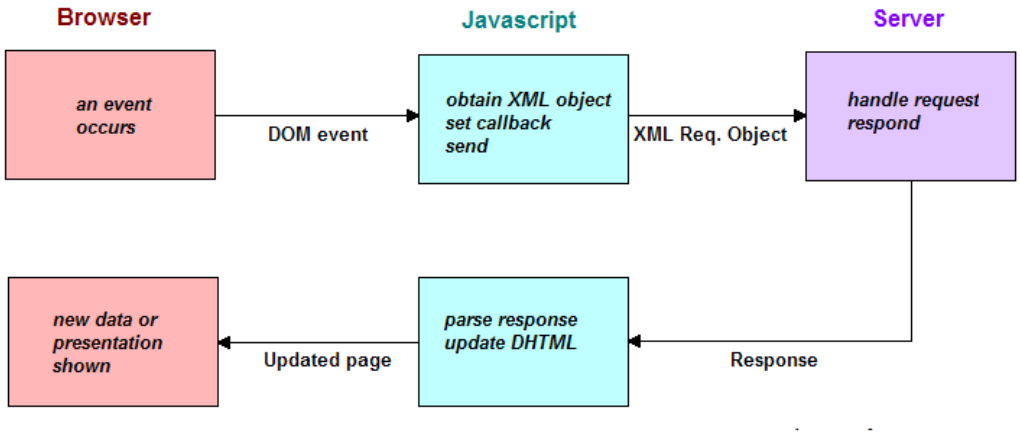
```
strData = objXHR.respondeText;
```

o

```
objJSON = eval (‘(‘ + objXHR.responseText + ‘)’ );
```

[10]

1.2.6 Come inviare e come ricevere richieste



1.2.6.1 L’oggetto request

Il punto di partenza per creare una comunicazione stile Ajax è l’oggetto request. JavaScript, infatti, può utilizzare questo oggetto per interrogare un server web per avere informazioni, memorizzarle e aggiornare la pagina quando il server le ha fornite.

1.2.6.2 Verifica del supporto per DOM e XMLHttpRequest

Il primo passo è la creazione dell’oggetto request. Come già detto, questo oggetto non è supportato da tutti i browser e in tutte le versioni allo stesso modo. Differenti browser usano differenti metodi per creare l’oggetto. Quindi per garantire una funzionalità cross-browser, bisogna verificare il supporto dell’oggetto XHR.

Internet Explorer usa un oggetto ActiveX mentre gli altri browser (Mozilla, Opera, Safari) usano l'oggetto JavaScript chiamato XMLHttpRequest.

Bisogna verificare, per prima cosa, se la versione di JavaScript utilizzata dall'utente supporta il DOM Scripting, verificando se sono supportati document.getElementById e document.getElementByTextNode:

```
if(!document.getElementById || !document.createTextNode)
```

Bisogna verificare poi se c'è il supporto per l'oggetto XMLHttpRequest e questo si può fare verificando se l'istruzione typeofXMLHttpRequest non abbia come valore "undefined". Dato che Explorer 6 e versione precedenti utilizzano il controllo ActiveXXMLHTTP, dobbiamo verificare che anche l'oggetto ActiveXObject non sia disponibile prima di poter desumere che il browser non supporta l'AJAX.

```
If (typeof XMLHttpRequest == 'undefined' && !ActiveXObject)
```

1.2.6.3 Come creare l'oggetto request

Il punto di partenza è la creazione dell'oggetto request, tenendo presente tutti i problemi di incompatibilità che ci sono. Possiamo scrivere il blocco per creare l'oggetto request così:

```
var request = null;

if (window.XMLHttpRequest) {

request = new XMLHttpRequest();}

else if (window.ActiveXObject) {

request = new ActiveXObject ("Microsoft.XMLHTTP"); }
```

1.2.6.4 Dove inviare la richiesta

A questo punto l'oggetto request chiede informazioni al server web che ha servito la pagina contenente il JavaScript responsabile della richiesta.

```
if (request) {

request.open ("GET", the_request);

…
```

L’oggetto request, per sapere dove inviare la richiesta, deve conoscere l’URL di un programma o script che risiede nel server web. Il programma lato-server elabora la richiesta e risponde con le informazioni.

1.2.6.5 La risposta

Una volta che un oggetto request ha inviato la richiesta, il browser può fare quello che vuole: anche creare altri oggetti request per fare nuove richieste. Una caratteristica fondamentale di Ajax è appunto l’asincronia. Ogni oggetto request creato deve tenere traccia del processo di effettuazione della richiesta, attendere la risposta e capire quando tutte le informazioni richieste sono state inviate. Quando l’oggetto passa da uno stato all’altro, cambia il valore della proprietà di request chiamata readystate. Con Ajax quindi possiamo scrivere una funzione che viene chiamata ogni volta che la proprietà cambia valore.

…

```
Request.onreadystatechange = function (){

If (request.readyState == 4) {
```

…

onreadystatechange è il gestore di evento dell’oggetto request e viene chiamato automaticamente ogni volta che cambia il valore di readystate. A questo gestore viene associata una funzione anonima. Questa funzione di solito non fa nulla fino a che il valore di readystate diventa 4, cioè quando la risposta è arrivata e tutte le informazioni sono state scaricate. A questo punto la funzione può fare qualcosa con i dati.

1.2.6.7 Gestire gli errori

L’oggetto XHR permette a JavaScript di comunicare direttamente con il server senza caricare una pagina nel browser e questo è senza dubbio un aspetto molto positivo in quando rende l’applicazione veloce e molto interattiva. Ma questo aspetto può diventare negativo in back end.

1.2.6.8 Inviare la richiesta

Dopo aver detto all’oggetto dove inviare la richiesta e cosa fare quando c’è una risposta, bisogna dire all’oggetto di inviare la richiesta, utilizzando il metodo send dell’oggetto request. Quando effettuiamo una richiesta di tipo GET il parametro del metodo send è impostato a null, quando la richiesta è di tipo POST contiene le informazioni da inviare.

…

```
request.send (null):
```

…

[5] [10] [11]

1.2.7 Dove e come si può usare Ajax

1.2.7.1 Utilizzabilità di Ajax

Il grande merito di Tim Berners Lee nella creazione del web è quello di aver unificato diversi concetti in una singola idea, la pagina:

- Unità di visualizzazione = unità di navigazione (link e segnalibri)
- URL, indirizzo testuale per recuperare informazioni nella rete
- Memorizzazione delle informazioni sul server e unità di modifica dell'autore.

Il web è stato progettato con l'idea di considerare la pagina come un'unità atomica di informazione e la nozione di pagina permea tutti gli aspetti del web. La facilità d'uso e al semplicità hanno fatto in modo che il web venisse adottato da un numero sempre crescente di utenti.

Ajax di fatto rompe questo modello unificato del web, introducendo un nuovo modo di manipolare i dati, che non è ancora stato ben integrato con gli altri aspetti del web. Con Ajax il modo dell'utente di visualizzare le informazioni sullo schermo è determinato da una sequenza di azioni di navigazione, invece che da una singola azione di navigazione. Quindi l'unità di navigazione è diversa dall'unità di visualizzazione. Non c'è più la rappresentazione dello stato del contenuto della pagina. A questo punto viene a mancare lo scopo principale dell'URL, non c'è più una completa specificazione delle informazioni visualizzati nella finestra del browser.

Non sempre Ajax viene utilizzato in modo corretto: esistono buoni esempi d'uso ma è facile rischiare di creare un'applicazione confusa e poco usabile. E' necessario prestare attenzione ad una serie di aspetti per garantire una corretta implementazione dell'Ajax.

- ***Pulsante indietro.***

Tipicamente in una applicazione Ajax il pulsante indietro non funziona come ci si aspetterebbe. In una pagina gestita con Ajax infatti tutto avviene in una singola pagina quindi, cliccando sul tasto

indietro, si rischia di uscire dall'applicazione stessa. Esistono dei framework Ajax che sono in grado di far funzionare il pulsante indietro in modo più utile per l'utente.

- ***URL e segnalibri***

  L'URL di una pagina web di un'applicazione Ajax non cambia quando cambia il contenuto in quanto gli aggiornamenti avvengono nella stessa pagina. Cambiare l'hash (l'ancora di un URL) è l'unico modo per cambiare l'indirizzo quando cambia lo stato e, anche in questo caso, ci si può affidare ai framework.

- ***Design di bassa qualità***

  Quando ci sono dei cambiamenti in una pagina web di un applicazione Ajax c'è il rischio che questi non vengano notificati nel modo opportuno e che quindi l'utente non si accorga che i contenuti sono stati aggiornati. E' importante quindi accertarsi che l'utente sia avvisato tramite tecniche di design dei cambiamenti più importanti. E' da evitare inoltre la creazione di uno stile di navigazione che si discosta troppo da quello standard in quanto potrebbe disorientare l'utente che naviga nell'applicazione.

  [11]

1.2.7.2 Quando non si deve usare Ajax?

- Usare Ajax solo perché è l'ultima moda e permette di inserire trucchi grafici particolari.
- Usare Ajax perché è una novità (novità non significa che è una tecnica che risolve problemi precedenti, anzi in realtà ne introduce di nuovi).
- Usare Ajax per sostituire qualcosa che è già perfettamente funzionante così com'è. Confondere gli utenti con forme di navigazione nuove e non necessarie non fa che allontanarli dall'applicazione stessa. E' sempre meglio evitare di creare nuove convenzioni per l'interfaccia utente.
- Usare Ajax senza prevedere un'alternativa per coloro che non possiedono o hanno già disabilitato JavaScript.
- Caricare una grossa quantità di testo senza ricaricare la pagina può essere un grosso problema per i motori di ricerca. Il testo dell'applicazione ha bisogno di essere in qualche modo statico per fare in modo che gli spider possano leggerlo.

- Cambiamenti asincroni della pagina possono confondere l'utente se non vengono usati con giudizio e se manca un adeguato avviso per ciò che sta accadendo.

  [11]

1.2.7.3 Quando Ajax può essere utile?

- Manipolazione di dati (modificare i dati in una tabella);
- Widget interattivi;
- Salvare lo stato;
- Miglioramento di interazione (inclusa la fase di validazione) nelle form;
- Utilizzo di menù a discesa con suggerimenti può aiutare utenti con difficoltà di lettura e/o difficoltà motorie;
- Un cursore per ingrandire il testo può essere un controllo utile per utenti con difficoltà di lettura;
- Possibilità di creare liste select dinamiche;

1.2.7.4 Applicazioni che utilizzano Ajax

Utilities

- ✓ **Meebo**, servizio che permette di connettersi con gli account di tutti i più famosi programmi di istant messaging dal browser, senza la necessità di avere il programma istallato
- ✓ **MoeMoe Design**
- ✓ **Facebook,** social network di grande successo
- ✓ **lFvR**

Agende, Organizers, TODO

- ✓ **Protopage**, servizio che permette di realizzare una home page personalizzata ricca di servizi,
- ✓ **Netvibes**, portale web personalizzabile basato sull'aggregazione on-line di feed RSS,
- ✓ **Kiko**, calendario online basato sul motto "click on anything",

- ✓ **Tadalists**, servizio web creato da 37signals che permette di realizzare e condividere liste di cose da fare,
- ✓ **Remember The Milk**, webapp per ricordare e tenere traccia delle cose da fare,
- ✓ **HipCal**, agenda appuntamenti e calendario
- ✓ **Basecamp**, strumento di project management fruibile esclusivamente via web rivolto al settore business ed educativo.

Editor di testo/HTML/fogli di calcolo

- ✓ **Ajaxwrite**, word processor sul web,
- ✓ **Writely**, word processor sul web in versione beta, acquisito da Google,
- ✓ **FCKeditor**
- ✓ **Ajaxxls**

Applicazioni Google

- ✓ **Google Maps**, servizio Google che consente la visualizzazione di mappe geografiche
- ✓ **Google Calendar**, servizio Google che consente di creare e condividere un calendario on-line,
- ✓ **Google Suggest**, servizio che analizza cosa viene scritto nel box di ricerca e offre suggerimenti di ricerca in tempo reale,
- ✓ **Google Spreadsheets**, servizio web di Google per la creazione di fogli di calcolo,
- ✓ **Gmail**, servizio gratuito di posta elettronica via web
- ✓ **MyTextFile**, editor di testo che può essere usato per memorizzare appunti, frammenti di testi o qualsiasi altra informazione.

2 Seconda parte: Accessibilità

"The power of the Web is in its universitality. Access by everyone regardless of disability is an essential aspect."

Tim Berners-Lee

2.1 Definizione

Un sistema si dice ACCESSIBILE se può essere usato da qualsiasi utente indipendentemente dal computer usato, dalla velocità di collegamento, dal browser utilizzato, dall'interfaccia utente e dalle periferiche alternative usate.

L'accessibilità non va confusa con l'usabilità. Si tratta di due aspetti diversi, seppur correlati tra loro. Mentre l'accessibilità ha lo scopo di allargare il numero di utenti l'usabilità fa in modo di rendere gli utenti più soddisfatti ed efficienti.

Non esiste un'unica definizione di accessibilità. L'aspetto che viene messo in evidenza da tutte le definizioni del termine è l'idea di accessibile come un qualcosa di raggiungibile, non con degli ostacoli, ma con una certa facilità. Inoltre l'accessibilità riguarda sia l'aspetto fisico (accessibile = raggiungibile per mezzo del corpo) sia l'aspetto mentale (accessibile = raggiungibile con la comprensione).

L'accessibilità nel web riguarda le difficoltà a percepire l'informazione (vista,udito), nella comprensione del testo e nell'interazione. Vanno considerati i limiti imposti dal dispositivo e dalla connessione utilizzata e le situazioni in cui i canali percettivi sono già occupati. Inoltre i vari brower e sistemi operativi non lavorano allo stesso modo.

L'accessibilità quindi non riguarda solo le disabilità permanenti ma anche tutte quelle situazioni che, indipendentemente da deficit fisici o psichici, sono causa di limitazioni della possibilità di fruire dei contenuti web.

Si distinguono tre categorie di disabilità nell'uso del computer:

1. Disabilità sensoriale: non vedenti, ipovedenti, sordi;
2. Disabilità motorie: paralisi, spasticità;

3. Disabilità cognitive: afasia, dislessia.

Possiamo quindi considerare come beneficiari dell'accessibilità:

- persone con disabilità e/o fisiche permanenti;
- persone che per ragioni contingenti incontrano difficoltà simili a quelle di chi è affetto da disabilità fisiche o cognitive permanenti;
- persone che usano tecnologie potenzialmente "invalidanti" (connessioni lente, browser vecchi).

I disabili possono accedere ad internet tramite particolari strumenti assistivi:

- Screen Reader;
- Barre elettroniche in rilievo Braille;

Figura 14: dispositivo Braille

- Ingranditori;

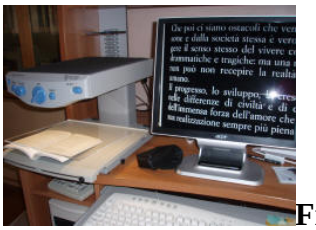


Figura 15: video ingranditore da tavolo

- Tastiere e mouse speciali.

Figura 16: mouse per disabili

[5] [6] [7]

2.3 Leggi sull'accessibilità

Quando si progetta un'applicazione web è necessario seguire delle regole per fare in modo che le tecnologie assistive funzionino al meglio e permettano così ai disabili di accedere alle informazioni nel miglior modo possibile.

I primi a richiedere per legge requisisti per l'accessibilità sono stati gli USA con la Section 508 (1998). In Italia l'accessibilità dei siti web è regolata dalla cosiddetta "**Legge Stanca**" o "Disposizione per favorire l'accesso dei soggetti disabili agli strumenti informatici" (9 gennaio 2004).

In questa normativa rientrano dunque anche le applicazioni web ed, infatti, al suo interno viene data la definizione di applicazione internet:

Una applicazione web è un programma sviluppato utilizzando il protocollo HTTP per il trasferimento dei dati e il linguaggio a marcatori XHTML per la presentazione e la struttura dell'informazione.

La normativa prevede quindi 22 requisiti tecnici ispirati alle linee guida WCAG 1.0 del W3C che devono essere soddisfatti necessariamente da parte di tutte le applicazioni web, anche non direttamente disponibili on-line.

Il punto fondamentale è che nessun contratto tra una pubblica amministrazione e un fornitore è valido se nel contratto stesso non si pone un riferimento al rispetto dei requisiti tecnici di questa legge.

I requisiti tecnici della legge Stanca seguono le linee guida WCAG del W3C e sono tutt'ora quelli che regolano l'accessibilità dei siti web in Italia. Diverse leggi esistono a livello europeo. A livello internazionale gli standard sono le WCAG, le ATAC e le UAAG, tutti realizzati dal consorzio W3C.

[5] [9]

2.4 Linee guida per l'accessibilità

Le linee guida di riferimento per l'accessibilità delle applicazioni web sono:

- **WCAG** Web Content Accessibility Guidelines
- **ATAG** Authoring Tools Accessibility Guidelines
- **UAAG** User Agent Accessibility Guidelines
- **EARL** Evaluation And Report Language
- **ARIA** Accessible Rich Internet Applications

[5]

2.4.1 WCAG

2.4.1.1 Introduzione

E'la sezione WAI (Web Accessibility Initiative) del W3C (World Wide Web Consortium), ente sovranazionale senza scopo di lucro, che si occupa di chiarire il concetto di accessibilità, fornendo gli strumenti per produrre contenuti accessibili. Gli esperti di questa sezione hanno lavorato alla stesura di una serie di linee guida sull'accessibilità dei contenuti Web e sono giunti alla versione WCAG 2.0.

2.4.1.2 Differenze WCAG 1.0 – WCAG 2.0

Indipendenza dalla tecnologia

Il concetto fondamentale che distingue le due linee guida è quello dell'indipendenza dalla tecnologia: mentre le WCAG 1.0 non sono indipendenti dalla tecnologia, le WCAG 2.0 sono progettate per esserlo. Questo si spiega tenendo presente che, quando le WCAG 1.0 sono state scritte, era il 1999 e il web era costituito fondamentalmente di documenti HTML. Ma negli anni successivi il web si è arricchito di applicazioni, contenuti, modalità di interazione e tecnologie a cui i criteri di accessibilità WCAG 1.0 possono essere estesi, e non sempre, solo per analogia. Quindi le WACG 1.0 fanno riferimento ad un web che ormai è cambiato. Le WACG 2.0 partono quindi dall'idea di avere una ricetta universale per l'accessibilità, con delle linee guida che siano valide nel tempo.

Accessibility supported

"WCAG 1.0, linea guida 10. *Usare soluzioni di accessibilità temporanee in modo che le tecnologie assistive e i browser meno recenti possano operare correttamente."*

Uno dei limiti maggiori delle WCAG 1.0 è quello di dare raccomandazioni transitorie, cioè che vanno applicate solo per ragioni di compatibilità all'indietro, non perché rappresentano un principio universale di sviluppo accessibile. Questo inevitabilmente lascia gli sviluppatori nell'incertezza e con troppa responsabilità.

Le WACG 2.0 eliminano quindi le raccomandazioni transitorie e qualsiasi raccomandazione orientata a una specifica tecnologia. Rimane comunque il problema della compatibilità con le tecnologie passate. Un primo tentativo di soluzione è il concetto di baseline, che poi è stato sostituito da accessibility supported.

Con il termine baseline si intende l'elenco completo delle tecnologie che un browser o un plug-in, eventualmente in combinazione con una tecnologia assistiva, devono supportare, affinché i contenuti delle pagine web dichiarate conformi alle WCAG 2.0 siano effettivamente accessibili. In questo modo lo sviluppatore può definire liberamente la griglia delle tecnologie, che i programmi devono supportare, affinché gli utenti possano accedere ai contenuti.

Ma questo concetto viene sostituito da quello di accessibilità supported, cioè supportato per l'accessibilità. Supportato per l'accessibilità significa supportato dalle tecnologie assistive degli utenti comuni come pure dalla caratteristiche di accessibilità di browser e altri programmi utente. Il WAI/W3C si impegna a creare delle liste pubbliche di informazioni sul supporto per l'accessibilità fornito dai programmi utente alle proprie tecnologie.

Il concetto di accessibility supported è stato preferito a quello di baseline in quanto quest'ultimo introduce un elemento di potenziale discriminazione, nel momento in cui la scelta è libera e affidata agli autori. Con accessibility supported questa discriminazione si dovrebbe eliminare, o quanto meno ridurre. La scelta della miglior infrastruttura tecnologica per il web dipenderà quindi dalla disponibilità e dalla qualità degli elenchi pubblici di tecnologie accessibility supported.

2.4.1.3 Organizzazione WCAG 2.0

La WCAG 2.0 è costituita da 4 principi, fondamenti dell'accessibilità:

1. **PERCEPIBILE**: l'utente deve essere in grado di percepire le indicazioni indipendentemente dalla propria disabilità;
2. **OPERABILE**: l'utente deve essere in grado di interagire con gli elementi dell'interfaccia, cioè l'interfaccia non può richiedere azioni per le quali l'utente non può reagire;

3. **COMPRENSIBILE**: gli utenti devono essere in grado di capire le informazioni e il funzionamento dell'interfaccia utente;
4. **ROBUSTO**: gli utenti devono essere in grado di accedere al contenuto anche con l'evoluzione delle tecnologie, ovvero il contenuto deve rimanere accessibile nel tempo.

Prevede 12 linee guida per aiutare a rispettare i principi e garantire che il contenuto sia direttamente accessibile al maggior numero di persone:

1. Alternative testuali
2. Media sincronizzati
3. Adattabilità
4. Distinguibilità
5. Accessibile da tastiera
6. Tempi di esecuzione
7. Convulsioni
8. Navigabilità
9. Leggibilità
10. Prevedibilità
11. Assistenza
12. Compatibilità

All'interno di ogni linea guida ci sono dei criteri di successo che descrivono in modo specifico cosa è necessario implementare per essere conformi a questa indicazione tecnica. Sono progettati per essere verificabili, cioè sono affermazioni che ammettono solo due risposte: sì o no.

Per ogni criterio di successo esiste una linea di tecniche, ciascuna fornita di un test di valutazione, che il Gruppo di Lavoro ha ritenuto sufficienti per soddisfare il requisito. Il meccanismo del criterio di successo è stato introdotto proprio per cercare di superare l'incertezza nella valutazione delle conformità che è ineliminabile nelle WCAG 1.0.

Le WCAG 2.0 ereditano dalle precedenti linee guida 3 livelli di conformità (A, doppia A, tripla A). Ciascun criterio di successo è essenziale per alcuni utenti e i livelli sono costruiti l'uno sull'altro, comunque anche i contenuti che sono conformi ad AAA possono non essere pienamente accessibili per alcune persone con una disabilità o con una combinazione di esse. I criteri di successo di livello A raggiungono l'accessibilità supportando le tecnologie assistive e imponendo allo stesso tempo i minori limiti possibili sulla presentazione. I criteri di successo di livello AA forniscono un supporto aggiuntivo alle tecnologie assistive, imponendo maggiori limiti sulla presentazione visuale e su altri aspetti del contenuto. I livelli di successo AAA incrementano sia l'accesso diretto sia l'accesso attraverso tecnologie assistive, ponendo limiti più forti su presentazione e contenuti.

[5] [8]

2.4.2 ATAG

Le ATAG sono delle linee guida indirizzate agli sviluppatori di sistemi software e il loro scopo è di guidarli nella realizzazione di contenuti web accessibili. Rispettare queste linee guida nella progettazione di una pagina web o di un CMS garantisce la conformità alle WCAG. [5]

2.4.3 UAAG

Le UAAG sono delle linee guida indirizzate a chi sviluppa i sistemi software che navigano i contenuti, detti User Agent. Implementando queste linee guida all'interno di un browser o di una tecnologia assistiva la fruizione di contenuti conformi alle WACG sarà migliore. [5]

2.4.5 EARL

EARL è un linguaggio per rappresentare il risultato di test di verifica dell'accessibilità in modo indipendente rispetto a strumenti in formato proprietario. [5]

2.4.6 ARIA

ARIA è una suite di documenti tecnici per generare applicazioni e contenuti web dinamici accessibili alle persone con disabilità.

2.4.6.1 Come nasce WAI-ARIA

La specifica Web Accessibility Initiative – Accessible Rich Internet Applications (WAI-ARIA) è stata sviluppata dal World Wide Web (W3C) ed è disponibile online all'indirizzo http://www.w3.org/WAI/intro/aria. Le limitazioni imposte dal linguaggio XHTML causano

problemi di accessibilità e usabilità alle applicazioni web 2.0. La prima bozza standard del W3C si propone quindi come risposta al problema, fornendo modi diversi di comunicare il significato, colmando alcune lacune dell'XHTML e migliorando l'usabilità proponendo modelli di navigazioni simili alle applicazioni desktop. L'estensione è stata proposta da Richard Schwerdtfeger dell'IBM, membro del gruppo di lavoro WAI su protocolli e formati, e da Becky Gibson, membro del gruppo di lavoro WAI WACG ed è stata poi implementata insieme ad Aaron Leventhal. Nonostante qualche critica, nel 2005 venne fornita una grande quantità di codice da parte dell'IBM per il progetto di Mozilla per l'accessibilità. Subito dopo venne pubblicata la prime bozze in lavorazione del W3C sui Ruoli e su Stati e Proprietà. Dopo un anno è stato introdotto come modulo XHTML 1.1.

2.4.6.2 I documenti di WAI-ARIA

Il First Public Working Draft di WAI-ARIA si basa su 3 documenti correlati:

1. WAI-ARIA Roadmap: tabella di marcia che elenca i problemi di accessibilità delle applicazioni dinamiche attualmente irrisolti e definisce i passi necessari per giungere a soluzioni standardizzate;
2. WAI-ARIA Roles: documento che definisce in maniera comprensibile alle tecnologie assistive il ruolo che ogni oggetto svolge all'interno dell'applicazione;
3. WAI-ARIA States and Properties: modulo che si occupa di standardizzare gli stati e le proprietà di un elemento.

[5] [10]

2.5 Strumenti per verificare l'accessibilità

La valutazione dell'accessibilità è un processo complesso che necessita delle conoscenza delle linee guida relative a questo argomento. Esistono comunque degli strumenti di validazione, applicabili in modo meccanico ad un sito, che possono aiutare nell'analisi.

Per fare una valutazione di accessibilità di un sito web occorre, per prima cosa, istallare una serie di strumenti che renderanno più facile il lavoro:

- Mozilla Firefox. E' possibile installare tutti i vari componenti aggiuntivi che si occupano di accessibilità e di validazione: Web Developer,HTML Validator, Firebug, Firefox Accessibility Extensions, Greasemonkey, Juicy Studio Accessibility Toolbar.

- Quick Accessibility Page Tester è un bookmarklet che si può cliccare in ogni momento per fare una rapida analisi della pagina web. Vengono messi in evidenza avvertimenti su possibili problemi (non identificati con la specifica WCAG) e vengono evidenziati aree della pagina che potrebbero trarre beneficio da alcuni miglioramenti ARIA.
- Browser testuale come Lynx
- Screen Reader (Jaws, Firevox,)

I punti fondamentali per la valutazione dell'accessibilità sono:

- Validazione HTML e CSS
- Non usare frames
- Strumenti automatici di verifica dell'accessibilità
- Immagini con testo alternativo
- Assicurarsi l'uso di JavaScript non invasivo
- Provare ad aumentare la dimensione del testo
- Usare il markup semantico
- Provare il sito con i CSS disabilitati

I controlli più specifici riguardano:

- Contrasto di colore
- Titoli del documento
- Link di testo
- Formati non HTML
- Discriminazione per piattaforma
- Navigazione da tastiera
- Tabelle dati

- Etichette e controlli per moduli
- Utilizzare uno screenreader
- Non trascurare i contenuti

3 Terza parte: Ajax accessibile

3.1 Introduzione

Le applicazioni web 2.0 e, in particolare, quelle realizzate con Ajax, possono risultare inaccessibili se non si progettano tenendo conto delle necessità degli utenti. Dato che i lavori per standardizzare le interfacce di applicazioni RIA sono ancora in atto, quello che deve fare chi si occupa di pagine web accessibili con AJAX è determinare la soluzione efficace per lo specifico problema che si incontra volta per volta. [5]

3.2 JavaScript

3.2.1 Cosa è JavaScript

JavaScript è un linguaggio di scripting orientato agli oggetti che viene interpretato dal browser, applicando, senza bisogno di compilazione preliminare , istruzioni ad alto livello. Uno script è una serie ordinata di istruzioni. Questo viene letto e interpretato dal browser, che esegue le istruzioni descritte. E' stato sviluppato dalla Netscape Communications ma poi è stato formalizzato con una sintassi più simile a Java di Microsystems.

3.2.2 Java Script non istallato o disabilitato

Linee guida sull'attuazione della Legge Stanca, requisito 15

"Assicurarsi che le pagine siano utilizzabili quando script, applet o altri oggetti di programmazione sono disabilitati oppure non supportati. Se questo non è possibile:

- *Fornire una spiegazione della funzionalità svolta;*
- *Garantire un'alternativa testuale equivalente."*

Quando si progetta un'applicazione web bisogna tenere presente, come prima cosa, che JavaScript potrebbe non essere istallato o disabilitato, quindi non è possibile utilizzare solo Ajax per gestire funzionalità fondamentali dell'applicazione. Infatti se, disabilitando JavaScript, l'utente non riesce ad accedere alle informazioni e ai servizi fondamentali di un'applicazione, vuol dire che questa non è accessibile. Inoltre ci sono dei browser, come Opera, che non interpretano bene l'oggetto XMLHttpRequest, seppure JavaScript sia abilitato. Quindi una pagina accessibile non può essere progettata solo utilizzando Ajax ma tutte le funzionalità ottimizzate tramite Ajax devono essere fruibili anche senza di esso.

3.2.3 JavaScript e accessiblità

Il rapporto tra JavaScript e accessibilità è complesso. Il solo fatto di utilizzare JavaScript però non significa che la pagina che si sta progettando sia inaccessibile. Anzi, in alcuni casi JavaScript può essere usato per migliorare l'accessibilità.

L'utilizzo di JavaScript richiede un browser dotato di un adeguato supporto, rendendo dipendente dal dispositivo la possibilità di accedere ai contenuti generati o modificati con JavaScript.

L'accessibilità intrinseca delle interfacce modificate con JavaScript non è così scontata, anche in browser ben supportati, ma dipende da una serie di fattori:

- presenza di semantica nel linguaggio di marcatura per descrivere funzioni e proprietà degli oggetti dell'interfaccia modificati tramite JavaScript;
- capacità dello sviluppatore di utilizzare la marcatura in modo utile per le tecnologie assistive;
- il tipo di interazione richiesta dall'utente.

[7]

3.2.4 JavaScript non invasivo

L'espressione Unobtrusive JavaScript, intesa come insieme di pratiche di buono sviluppo, ha cominciato a diffondersi nel 2005 ed ora può considerarsi uno standard di fatto.

Il JavaScript è diventato ormai un elemento fondamentale per garantire l'interattività nelle moderne applicazione e, quindi, deve garantire che queste funzionalità siano fruibili ad una larga massa di utenti.

Il JavaScript non invasivo è solido e perfettamente organizzato, in modo da potersi integrare con gli altri livelli (contenuto, struttura e presentazione).

Quando si vuole progettare un JavaScript non invasivo bisogna tenere presente due principi fondamentali:

1. gli script vanno utilizzati per potenziare l'usabilità e l'interattività di un'applicazione web ma senza che l'accesso alle funzionalità base sia dipendente dal JavaScript;
2. totale separazione tra gli strati che compongono un documento.

L’applicazione dei 2 principi richiede l’applicazione di una serie di principi di sviluppo compatibili con le richieste dell’accessibilità:

- gli script devono essere conformi alle specifiche DOM del W3C;
- dichiarare solo variabili locali;
- progettare interazioni non dipendenti dal dispositivo;
- non inserire codice di script all’interno del codice di marcatura;
- evitare di modificare gli stili tramite JavaScript;
- raggiungere, modificare e generare contenuti solo attraverso il DOM.

[7]

3.3 Problematiche RIA

3.3.1 Cosa sono le RIA

RIA (Rich Internet Application) è un termine coniato da Macromedia e definisce un’applicazione web-based con caratteristiche e funzionalità tipiche delle applicazioni desktop. Grazie ad Ajax c’è un aumento della velocità di interazione e si possono aggiornare i contenuti senza perdere i dati contenuti nei form. Si evita inoltre il termine della sessione utente senza dover richiedere di nuovo l’intera pagina.

I primi esempi di RIA sono stati creati da Google con GMail prima, poi seguito da Google Maps.

[5]

3.3.2 Problematiche RIA

Le problematiche delle RIA si possono distinguere in 2 aspetti:

1. Elementi (widget): le proprietà dell’oggetto che vengono rappresentate visivamente non vengono inoltrate alle tecnologie assistive.
2. Feedback: l’utente di tecnologie assistive deve essere informato se succede qualcosa.
3. Disponibilità: è necessario che anche tramite tecnlogie assistive si possano trovare ed utilizzare i contenuti che vengono aggiunti, rimossi e modificati.

[5]

3.4 Problematiche specifiche di AJAX

3.4.1 Introduzione

Le problematiche di Ajax non sono legate solo ai problemi che JavaScript manifesta per sé ma, ogni volta che Ajax rompe le convenzioni del comportamento di un browser, ci possono essere delle difficoltà per gli utenti disabili. Questa situazione può essere risolta con accorgimenti specifici legati alle singole modalità di utilizzo di Ajax.

Con Ajax si può modificare solo parte della pagina e quindi non sempre si riesce a notificare a sufficienza queste modifiche così che l'utente si accorga degli effetti delle proprie azioni. Le tecniche utilizzate per avvisare gli utenti variano a seconda del tipo di applicazione e del tipo di disabilità di chi la utilizza.

Le problematiche del feedback e della disponibilità, in comune con le RIA, sono quelle direttamente collegate all'Ajax stesso e vengono risolte con WAI-ARIA, che ancora non è ben supportato. Bisogna quindi pensare ad alcune modalità di risoluzione differenti del problema. [5]

3.4.2 FAT (Fade Anything Technique)

3.4.2.1 Introduzione

Quando si naviga su una pagina che utilizza una tecnologia Ajax, è importante fare in modo che gli aggiornamenti accorsi alla pagina siano evidenti. Una tecnica che può essere usata per comunicare correttamente all'utente che qualcosa è cambiato è la FAT (Fade Anything Technique), un'evoluzione della YFT (Yellow Fade Technique) inventata da 37signals che usa JavaScript non invasivo.

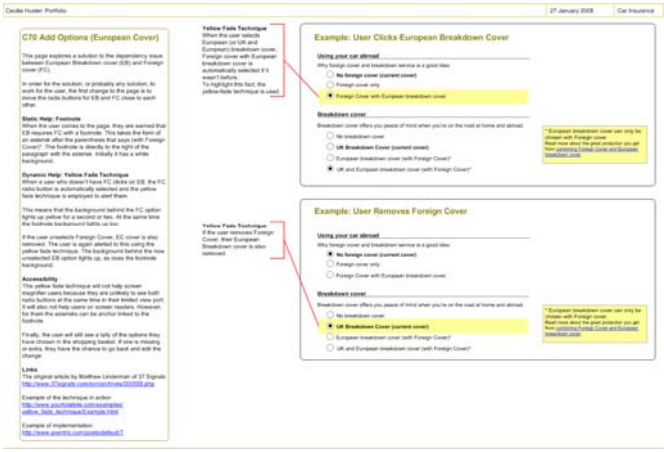


Figura 17: esempio di utilizzo di YFT

3.4.2.2 YFT

La tecnica YFT in pratica comunica al browser di prendere la parte di pagina modificata e di mostrarla in giallo, presupponendo che il giallo non sia il colore predominante dell'applicazione. L'utente in questo modo viene attirato sulla parte modificata e notare facilmente il cambiamento. La necessità di creare questa tecnica nasce dal fatto che in molti siti web e applicazioni si corre il rischio di non riuscire a visualizzare i cambiamenti che avvengono asincronicamente. La prima applicazione ad utilizzare la YFT è appunto Basecamp di 37signals. La tecnica utilizza JavaScript per creare una sottolineatura gialla che illumina per breve tempo il cambiamento quando la pagina si aggiorna. Poi dopo un paio di secondi il giallo si affievolisce e la pagina torna allo stato normale. Questo effetto rende quindi molto facile da individuare le modifiche e i cambiamenti e grazie alla sua natura non invasiva permette agli utenti di tornare rapidamente al lavoro non appena la modifica è confermata.

3.4.2.3 FAT

La FAT si costruisce quindi a partire dalla YFT consentendo di dissolvere qualsiasi elementi da qualsiasi colore al colore di sfondo nativo senza l'uso di JavaScript in linea. Chiaramente sia la YFT che la FAT non possono che essere delle soluzioni parziali per la notifica dei cambiamenti, soprattutto perché non possono aiutare in alcuna maniera gli utenti di screen reader.

3.5 Possibili approcci di progettazione

3.5.1 Interfaccia alternativa senza scripting

La soluzione più semplice, ma anche quella più limitata, è quella di fornire un'alternativa solo testo che non faccia uso di script in modo da poter essere visualizzata con qualsiasi device. La pagina alternativa è accessibile se permette di eseguire le funzioni principali, tuttavia non sarà mai possibile avere delle versioni del tutto equivalenti in quanto alcuni servizi non sono gestibili senza ricorrere all'uso degli script.

3.5.2 Funzioni alternative senza scripting

Non sempre uno script ha una funzione fondamentale per portare a termine un determinato compito ma spesso è solo un modo di arricchire l'esperienza dell'utente e di rendere più piacevole la grafica dell'interfaccia. In questo caso l'uso di JavaScript non rende inaccessibile un'applicazione. Quindi l'applicazione è accessibile se tutte le funzioni presenti sono utilizzabili senza bisogno di ricorrere allo scripting. Lo script serve quindi solamente per ottimizzare lo svolgimento di alcuni compiti.

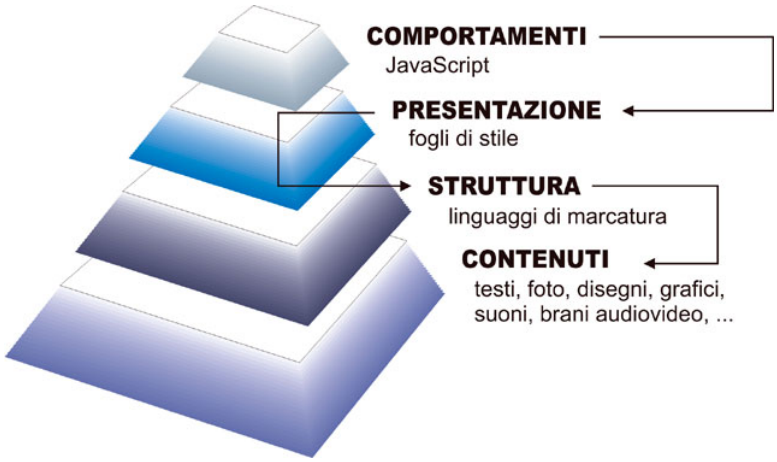
3.5.3 Progressive Enhancement (Miglioramento progressivo)

La tecnica del potenziamento progressivo invece prevede di realizzare un'unica applicazione, strutturando a piramide i dati in modo che i contenuti strutturati permettano all'utente, tipicamente grazie a collegamenti ipertestuali e form, di utilizzare tutte le funzionalità dell'applicazione anche senza JavaScript. Quando si realizza il livello dei comportamenti, l'aggiunta delle interazioni dinamiche basate su Ajax viene fatta "dirottando" le azioni eseguite da link e moduli per mezzo di script realizzati seguendo il modello di Javascript non invasivo. In questo modo è possibile ricaricare solo gli elementi modificati e non l'intero documento. L'oggetto XMLHttpRequest deve essere utilizzato nel modo più semplice possibile e quindi i suoi compiti saranno spedire dati dal client al server e spedire frammenti di pagina dal server al client. I processi client-side sono ridotti al minimo, lasciando il grosso del lavoro al server. Ajax è dunque posto ad un livello aggiuntivo ed opzionale della marcatura HTML. Il teorizzatore di questo uso non invasivo di Ajax è Jeremy Keith e lo ha definito Hijax. "*Progettare per Ajax dall'inizio, implementare Ajax alla fine.*" è la filosofia di questo approccio, che comporta una serie di vantaggi rispetto al vecchio metodo di progettazione della versione alternativa non-Ajax:

Figura 18: Jeremy Keith

- Non si perde tempo per creare due applicazioni distinte;
- I file JavaScript (file esterni) sono di dimensioni ridotte;
- I link sono visibili agli spider e ai bookmark.

E' una filosofia che prevede una progettazione modulare, separando i livelli:



- **XHTML** per il livello di comprensione;
- **CSS** per il livello di presentazione;
- **DOM SCRIPTING** per il livello dei comportamenti.

3.5.4 Rendere gli script accessibili

La soluzione migliore sarebbe rendere gli script che fanno parte di un'applicazione RIA completamente accessibili. Per rendere possibile questo approccio bisogna sfruttare le API per le accessibilità. E' necessario definire per ogni elemento la funzione e lo stato, in modo da poter essere presentato all'utente nel modo migliore a seconda del dispositivo utilizzato. E' in questa direzione che sta lavorando il progetto WAI-ARIA del W3C.

[19] [5] [22]

3.6 Come usare un Ajax non invasivo

3.6.1 Cosa è l'Ajax non invasivo?

Progettare un Ajax non invasivo significa progettare mantenendo separazione tra contenuto (HTML), presentazione (CSS) e comportamento (javascript).

La separazione fisica riguarda:

- Mettere i fogli di stile in un file .css e gli script javascript in un file .js;
- Evitare di usare `onload` e `onclick` quando possibile;
- Non usare `<style>`
- Non usare tag deprecati come `<font>` o `<align>`.

La separazione concettuale riguarda:

- Contenuto e form devoono essere disponibili in formato html;
- Form e link devono funzionare anche senza javascript;
- La presentazione è nel CSS, non nell'HTML o nel javascript;
- Ognuno deve poter accedere al contenuto con qualsiasi client, con CSS e javascript disabilitati o senza mouse.

[30]

3.6.2 HTML non invasivo

Il punto di partenza della progettazione è il codice HTML, che deve essere non invasivo. Per realizzare un'applicazione web si deve sfruttare tutta la gamma dei tag, facendo attenzione ad usare i tag `<h1>` per gli headers, i `<label>` per le form. e le tabelle solamente per i dati modulari.

[30]

3.6.3 CSS non invasivo

Tutti i fogli di stile sono non invasivi. Quando applicano dei css ad una applicazione è buona norma usare classi riutilizzabili quando è possibile e metterli in un file esterno .css o in un tag `<style>`.

[30]

3.6.4 JAVASCRIPT non invasivo

L'idea è di partire dall'HTML per poi migliorarlo con il javascript. La pagina deve funzionare comunque anche se javascript è disabilitato. E' preferibile non utilizzare `onclick` o `javascript:void(0)`. Anche il javascript va messo in un file esterno (.js) o in `<script>`.

[30]

3.6.5 AJAX non invasivo

L'uso dell'Ajax non invasivo migliora l'accessibilità di un applicazione web. Il codice è più chiaro e più facile da mantenere. La progettazione a strati separati rende possibile fare il restyling del sito solamente cambiando i CSS. Inoltre il javascript può essere cambiato facilmente e anche l'HTML si può cambiare in sicurezza.

[30]

3.7 WAI-ARIA come soluzione per un Ajax accessibile

3.7.1 Premessa

Risolvere i problemi di accessibilità non è così semplice. Il problema fondamentale, come abbiamo già detto, è JavaScript. W3C si sta sforzando per creare uno standard di accessibilità per le applicazioni web basate su JavaScript. Infatti il Web Content Guidelines 2.0 sarà il primo standard di accessibilità del W3C che non scoraggerà l'uso del JavaScript. Già questo è un enorme passo avanti. Lo sforzo per rendere accessibile Ajax potrebbe richiedere anni ma gli sviluppatori hanno

bisogno di avere soluzioni ora. Gli sviluppatori dovranno superare molte sfide nel cercare di attenersi agli standard di accessibilità odierni in quanto le linee guida WAI attuali scoraggiano l'uso di JavaScript. Tradizionalmente nell'accessibilità del web si chiede di fornire degli equivalenti, strutturando i dati semanticamente e facendo in modo che l'utente sia consapevole di ciò che sta succedendo nell'applicazione.

[5]

3.7.2 Problemi su cui lavora WAI-ARIA

Per capire meglio le soluzioni che propone il W3C per superare i problemi delle applicazioni con Ajax è bene capire quali sono i problemi su cui i documenti di WAI-ARIA lavorano:

1. bisogna avvisare la tecnologia assistiva dei cambiamenti dinamici che avvengono nell'interfaccia;
2. è importante il modo in cui la tecnologia assistiva viene avvisata dei cambiamenti dell'interfaccia.

[5]

3.7.3 Potenziamento del DOM

Il DOM può essere potenziato con un set di proprietà chiamate W3C ARIA che consentono ad un browser conforme di generare eventi specifici in grado di attivare le tecnologie assistive. Incrementare il DOM con le proprietà ARIA è una condizione necessaria per l'accessibilità delle applicazioni Ajax.

[5]

3.7.4 Aspetti Tecnici di WAI-ARIA

3.7.4.1 Introduzione

WAI-ARIA fornisce un framework per aggiungere attributi che identificano le caratteristiche dell'interazione con l'utente, il modo in cui si relazionano e il loro stato corrente. WAI-ARIA descrive nuove tecniche di navigazione per marcare regioni e comuni strutture web con menù, banner di informazione, contenuto principale e altri tipi di struttura web. WAI-ARIA include inoltre tecnologie per mappare i controlli, regioni attive Ajax, APIs, compresi i controlli personalizzati usati per le RIA. Le tecniche WAI-ARIA si applicano ai widget come ai bottoni, elenchi a cascara, funzioni di calendario, menù espandibili e altro.

WAI-ARIA fornisce ai progettatori web:

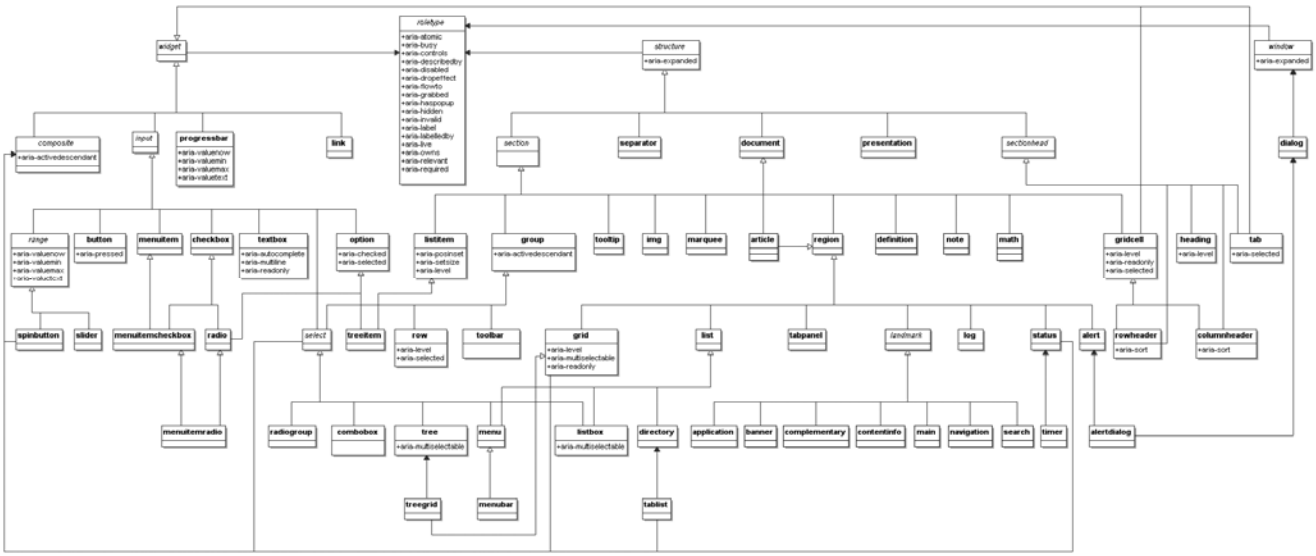
- **Ruoli** per descrivere il tipo di widget;
- **Ruoli** per descrivere la struttura della pagina web;
- **Proprietà** per descrivere lo stato in cui sono i widget;
- **Proprietà** per definire le regioni attive di una pagina che sono in grado di ricevere l'update;
- un modo per fornire la navigazione da tastiera per oggetti ed eventi.

[5]

3.7.4.2 Ruoli

3.7.4.2.1 Cosa è un ruolo

Un ruolo è un indicatore del tipo, è un modo per standardizzare la funzione svolta dall'oggetto all'interno dell'applicazione. Il ruolo di un oggetto è la classe dell'oggetto di cui è membro. Questa associazione semantica permette ai vari tools di presentare e supportare l'interazione con l'oggetto in modo che sia coerente con le aspettative degli utenti su oggetti di quel tipo. I ruoli sono tipi di elementi che non dovrebbero cambiare con il tempo o con l'azione dell'utente. Cambiare il ruolo di un elemento rispetto al suo valore iniziale confonderebbe una tecnologia assistiva ed anche l'utente stesso. I ruoli definiti dallo sviluppatore per ogni elemento dell'interfaccia vengono inviati alle API per l'accessibilità.



I ruoli sono definiti e descritti dalle loro caratteristiche. Le caratteristiche definiscono la funzione strutturale di un ruolo, cioè ci dicono quello che è un ruolo, i concetti che stanno dietro e quale istanza del ruolo può o deve contenere. Nel caso di widgets il ruolo include anche il modo in cui esso interagisce con lo user agent basandosi sulla mappatura dei form HTML e degli XForms. Vengono indicati anche gli stati e le proprietà di WAI-ARIA che sono supportate dal ruolo.

[19] [29]

3.7.4.2.2 Ruoli astratti

I ruoli astratti rappresentano la base su cui sono stati costruiti tutti i ruoli ARIA. Non devono essere utilizzati dagli sviluppatori perché non sono implementati nelle API. I ruoli astratti sono stati realizzati per:

- organizzare la tassonomia dei ruoli e fornire ruoli con un significato nel contesto di concetti noti;
- razionalizzare l'aggiunta di ruoli che includono caratteristiche necessarie.

[38]

3.7.4.2.3 Categorizzazione dei ruoli

Per supportare lo scenario attuale, la specifica ARIA categorizza i ruoli che definiscono l'interfaccia utente dei widget e quelli che definiscono la struttura della pagina. [38]

3.7.4.2.4 Tipi di base

Questi ruoli sono usati come tipi di base per ruoli applicati. Le classi di base sono utilizzate per descrivere il livello più alto della classe gerarchica della tassonomia dei ruoli. Tutti i ruoli in questa sezione sono astratti ma non tutti i ruoli astratti sono definiti qui. I ruoli astratti sono usati per costruire l'ontologia quindi gli sviluppatori non devono usarli nelle loro applicazioni.

- Composite
- Landmark
- Roletype
- Structure
- Widget
- Window

[38]

3.7.4.2.5 Ruoli per User Input Widgets

I seguenti ruoli si riferiscono ad elementi di una form o altri elementi di interaccia utente comuni nei widget, utilizzati per raccogliere e mantenere l'input dell'utente.

- Checkbox
- Combobox
- Input (astratto)
- Listbox
- Option
- Radio
- Radiogroup
- Range (astratto)
- Select (astratto)
- Slider
- Spinbutton
- Textbox

[29]

3.7.4.2.6 Ruoli per elementi di User Interface

I seguenti ruoli sono usati come parte dell'intefaccia grafica per l'utente.

- Button
- Link
- Menu
- Menubar
- Menuitem

- Menuitemcheclbox
- Menuitemradio
- Tablist
- Tarpane
- Tab
- Toolbar
- Tooltip
- Tree
- Treegrid
- Treeitem

[38]

3.7.4.2.7 Ruoli per la struttura del documento

I seguenti ruoli descrivono la struttura che organizza il contenuto della pagina. Solitamente la struttura del documento non è interattiva.

- Article
- Columnheader
- Definition
- Directory
- Document
- Grid
- Gridcell
- Group

- Heading
- Img
- List
- Listitem
- Math
- Note
- Presentation
- Region
- Row
- Rowheader
- Section (astratto)
- Sectionhead (astratto)
- Separator

[38]

3.7.4.2.8 Ruoli per la struttura dell'applicazione

I seguenti ruoli sono speciali in quanto contengono campi dell'interfaccia utente di un'applicazione.

- Alert
- Alertdialog
- Dialog
- Log
- Marquee
- Progress bar
- Status

- Timer

[38]

3.7.4.2.9 Ruoli strutturali (Landmark)

I seguenti ruoli sono regioni della pagina intese come punti di riferimento navigazionali. Tutti questi ruoli ereditano dalla base Landmark il tipo e, con l'eccezione di application, sono tutti importati dalla specifica XHTML Role Attribute Module. I ruoli sono inclusi in questa categoria per renderli chiaramente parta della tassonomia dei ruoli ARIA.

- Application
- Banner
- Complementary
- Contentinfo
- Main
- Navigation
- Search

Figura 19: ruoli landmark

[36] [38]

Vediamo poi quali sono i ruoli che sono stati definiti nella bozza WAI-ARIA Roles per i quali HTML e XHTML non possiedono specifici marcatori strutturali:

- slider - elemento che funge da cursore e che deve essere governabile anche da tastiera;
- tree – una serie di elementi che formano una struttura ad albero;
- treeitem – elemento che fa parte di un albero;
- alert – contenitore di un messaggio di avviso o di errore;
- region – gruppo di elementi, dotati di un titolo, che insieme formano una regione percepibile;
- menu – menù di navigazione;
- toolbar – collezione di funzioni usate comunemente, rappresentate in una forma visuale compatta;
- menuitem – link in un menù di navigazione;
- breadcrumbs – gruppo di link che forma le “briciole di pane”.

[38]

3.7.4.3 Stati e Proprietà

3.7.4.3.1 Cosa sono gli stati e le proprietà

L’informazione data dal ruolo non è sufficiente: ogni elemento, a seconda del suo ruolo, può trovarsi in una serie di stati diversi e avere differenti proprietà. Dato che stati e proprietà possono essere modificati dinamicamente tramite script, è necessario comunicare anche alle tecnologie assistive lo stato in cui si trova l’elemento dell’interfaccia.

Il meccanismo preferito dalle tecnologie assistive per accedere alle proprietà di un elemento è quello di mappare gli stati e le proprietà presso le API per l’accessibilità del sistema operativo.

Quando è combinato con i ruoli, lo user agent può supplire alla tecnologia assistiva con l’informazione in modo da renderla in ogni caso utile per l’utente in tempo utile. I cambiamenti negli stati o nelle proprietà si tradurranno in una notifica alla tecnologia assistiva, che può avvertire l’utente che si è verificato un cambiamento.

Alcuni stati di accessibilità, chiamati managed states, sono controllati dagli user agent, inclusi il focus da tastiera e la selezione. Questi stati spesso corrispondono a pseudo-classi CSS per definire lo stile che cambia. Tutti gli altri stati invece sono controllati dallo sviluppatore e vengono chiamati unmanaged states.

Lo stato è un proprietà dinamica che esprime le caratteristiche di un oggetto che può cambiare in risposta all'azione dell'utente o di processi automatizzati. Lo stato non riguarda la natura essenziale dell'oggetto ma rappresenta i dati associati con l'oggetto e la possibilità di interazione dell'utente.

Le proprietà sono essenziali per la natura di un determinato oggetto. Sono quindi meno soggette a cambiamenti, rispetto agli stati, in quanto un cambiamento di una proprietà può avere un impatto significativo sul significato o la rappresentazione di un oggetto.

Quindi è chiaro che i termini stato e proprietà fanno riferimento a caratteristiche simili: entrambi forniscono informazioni specifiche su un oggetto ed entrambi fanno parte delle definizione della natura dei ruoli. Sono comunque dei concetti che devono essere mantenuti distinti, per mettere in evidenza sottili differenze sul loro significato, ma possono essere trattati entrambi come attributi di marcatura prefissati con ARIA.

[38]

3.7.4.3.2 Tassonomia degli stati e delle Proprietà ARIA

Attributi Widget

In questa sezione sono contenuti gli attributi specifici per gli elementi dell'interfaccia utente che si trovano nei sistemi GUI o nelle applicazione internet ricche che ricevono l'input dell'utente e processano le sue azioni. Questi attributi sono usati per supportare i ruoli user input e user interface. Possono essere mappati da uno user agent ad una piattaforma di Api o possono essere letti direttamente dal DOM. I cambiamenti negli stati devono dar luogo ad una notifica alla tecnologia assistiva o tramite eventi di cambiamento negli attributi del DOM o tramite eventi nella piattaforma di accessibilità API.

- aria-checked
- aria-disable
- aria-expanded

- Aria-hidden
- Aria-invalid
- aria-pressed
- aria-selected

- aria-autocomplete
- aria-hashpopup
- aria-level
- aria-multiline
- aria-multiselectable
- aria-readonly
- aria-required
- aria-sort
- aria-valuemax
- aria-valuemin
- aria-valuenow
- aria-valuetext

[38]

Attributi delle regioni attive

In questa sezione sono contenuti gli attributi specifici delle regioni attive nelle applicazioni internet ricche. Possono essere applicati a qualsiasi elemento. Lo scopo è indicare che il contenuto può subire modifiche anche se l'elemento non ha il focus e fornire alle tecnologie assistive informazioni su come processare questi contenuti aggiornati. Alcuni ruoli specificano un valore di default per gli attributi aria-live specifici per quel ruolo.

- aria-busy

- aria-atomic
- aria-live
- aria-relevant

[38]

Attributi drag-and-drop

In questa sezione sono elencati gli attributi che indicano informazioni sul drag-and-drop degli elementi di interfaccia. Queste informazioni devono essere rese visibili e disponibili alla tecnologie assistive attraverso una modalità alternativa.

- aria-dropeffect
- aria-grabbed

[38]

Attributi Relazionali

In questa sezione sono elencati gli attributi che indicano relazioni o associazioni tra elementi che non possono essere facilmente determinati dalla struttura del documento.

- aria-activedescendant
- aria-controls
- aria-describedby
- aria-flowto
- aria-label
- aria-labelledby
- aria-owns
- aria-posinset

- aria-setsize

[38]

3.7.4.3.3 Uso di ruoli,stati e proprietà ARIA in XHTML

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:wairole="http://www.w3.org/2005/01/wai-rdf/GUIRoleTaxonomy#"
xmlns:aaa="http://www.w3.org/2005/07/aaa">
<head>...</head>
<body>
...
<span class="checkboxtristate"
id="chbox1"
role="wairole:checkboxtristate"
aaa:checked="mixed"
onkeydown="return checkBoxEvent(event);"
onclick="return checkBoxEvent(event);"
tabindex="0">
Un'etichetta per una casella di spunta
</span>
...
</body>
</html>
```

I ruoli vengono definiti prefissandoli col nome qualificato **wairole**, stati e proprietà con **aaa**. L'aspetto più importante è che i valori di stati e proprietà possono essere modificati dinamicamente per mezzo di script che accedono al DOM. [7]

3.7.4.4 Modo in cui le tecnologie assistive dialogano

L'intuizione su cui si basa WAI-ARIA è che le tecnologie assistive funzionano con i controlli delle tradizionali applicazioni desktop. Ma i controlli personalizzati delle applicazioni web non sono altro che delle varianti dei controlli tradizionali, quindi possono essere gestiti allo stesso modo.

3.7.4.5 Modalità con cui si processa il focus

3.7.4.5.1 Il problema del focus

Una applicazione deve sempre avere un elemento con il focus quando è in uso, così come un'applicazione richiede che gli utenti abbiano un posto dove gli venga fornito l'input. Gli elementi con il focus non devono essere nascosti o messi fuori dallo schermo ma tutti gli elementi interattivi devono poter ricevere il focus n maniera evidente e rilevabile, tramite tabulazione o altre tecniche di navigazione standard, per permettere agli utenti che accedono da tastiera di portare il focus ad ogni elemento interattivo.

Quando si progetta un'applicazione utilizzando (X)HTML e widget WAI-ARIA gli sviluppatori possono semplicemente manipolare l'ordine di tabulazione o usare uno script per creare scorciatoie da tastiera per gli elementi della pagina. Un uso di widget più complessi richiede allo sviluppatore di gestire il focus al loro interno.

WAI-ARIA include una serie di widget "managing container", chiamati anche widget compositi. Tipicamente il contenitore è responsabile del monitoraggio dell'ultimo discendente che è stato attivato (di default il primo elemento del contenitore). Quando un contenitore, che precedentemente ha ricevuto il focus, lo riceve di nuovo, il nuovo discendente attivo dovrebbe essere lo stesso elemento discendente che è stato attivato quando il contenitore ha ricevuto l'ultimo focus.

Quando qualcosa nel contenitore riceve il focus, l'utente può navigare attraverso il contenitore premendo i tasti aggiuntivi, come i tasti freccia, per muoversi sull'elemento selezionato. Ogni ulteriore pressione del tasto di navigazione principale (solitamento il tab) farà uscire dal contenitore per andare al widget successivo.

Quindi questo vuol dire che il focus degli elementi di ogni contenitore può essere gestito dal contenitore stesso utilizzando la prorietà aria-activedescendant o con la gestione del tabindex degli elementi figli del contenitore.

Contenitori che gestiscono il focus in questo modo sono:

- combobox
- grid
- listbox

- menu
- menu bar
- tree
- treegrid
- tablist
- toolbar

[32]

3.7.4.5.2 Perché è importante garantire il focus da tastiera?

E' un aspetto fondamentale per l'accessibilità il dovere di mantenere persistente l'indicazione del focus. Lo sviluppatore è quindi responsabile di mantenere, negli script, un focus visuale e programmatico, osservando regole di comportamento accessibili. Screen reader e utenti che utilizzano solamente la tastiera si affidano al focus per poter utilizzare un'applicazione internet ricca. [39]

Usare Tabindex per gestire il focus nei widget

L'attributo tabindex può essere usato per includere elementi addizionali nell'ordine di tabulazione e di impostare il focus su di loro. Tabindex è implementato per la prima volta su Internet Explorer 5, ora la funzionalità è estesa anche a Firefox e Mozilla. Se l'attributo tabindex non è presente solo i controlli dei form e le ancore possono ricevere il focus, l'ordine di navigazione è quello di default. Se il valore di tabindex è 0 l'elemento può ricevere il focus con il mouse o con il metodo `element.focus()` di JavaScript e l'ordine di navigazione dipende dalla posizione relativa dell'elemento all'interno del documento. Se il valore di tabindex è -1 l'elemento può ricevere il focus ma non viene inserito nella navigazione.

Quindi inserendo tabindex=-1 sugli elementi modificati dinamicamente, facendo in modo che uno script passi il focus a questi elementi solo dopo che sono stati modificati, è possibile fare leggere agli screenreader i contenuti modificati con Ajax. Tuttavia l'uso di tabindex con valori negativi non è previsto dalle specifiche HTML e XHTML e quindi produce errori di validazione nel codice di marcatura. [39]

Attributo Tabindex	Focusabe con mouse o con element.focus()	Navigazione con tab
Non presente	Segue il comportamento di default dell'elemento (solo i campi delle form e le ancore possono ricevere il focus)	Segue il comportamento di default dell'elemento
tabindex="0"	si	la sequenza di tabulazione segue l'ordine del documento
tabindex positivo	si	la sequenza di tabulazione segue il valore di tabindex
Tabindex negativo	si	l'elemento non viene inserito nell'ordine di tabulazione ma è abilitato a ricevere il focus da tastiera tramite element.focus() come risultato della pressione di un tasto (tasto freccia o altro)

3.7.4.5.3 Usare activedescendant per gestire il focus

Oltre a tabindex, ARIA prevede la proprietà aria-activedescendant per gestire il focus di un elemento figlio all'interno di un widget. Il genitore può utilizzare questa proprietà per indicare quale è il figlio attivato. L'elemento contenitore deve mantenere il punto di relazione e gestire quale dei singoli elementi figli deve essere percepito come attivo. Il gestore di eventi sul padre cattura le azioni della tastiera e determina quale è il successivo elemento che diventa attivo e aggiorna la prorietà aria-activedescendat con l'ID dell'appropriato elemento figli attivo. Il browser prende l'informazione dall'ID e genera l'evento di focus per la tecnologia assistiva. Ogni singolo elemento non deve essere reso focusable tramite un valore negativo del tabindex ma tramite lo stile CSS si deve indicare lo status attivo. [39]

3.7.4.5.4 La navigazione da tastiera tra e all'interno dei widget

Navigazione da tastiera tra i widget

La proprietà tabindex permette di portare al focus agli elementi HTML e per il WAI-ARIA si può applicare a tutti gli elementi (non c'è più la limitazione a form e elementi ancora) in modo da aiutare gli sviluppatori a produrre widget 2.0 accessibili da tastiera.

Nella tabella precedente abbiamo visto come cambia l'ordine di tabulazione a seconda del valore tabindex impostato.

Una volta che il widget ha il focus della tastiera, i tasti freccia, la barra spaziatrice, il tasto invio e altri comandi da tastiera possono essere usati per navigare tra le opzioni del widget, cambiare il suo stato o attivare una funzione ad esso associata.

Navigazione da tastiera all'interno dei widget

Ogni elemento deve ricevere il focus da tastiera. Questo è essenziale per comunicare le informazioni sul widget alle varie tecnologie assistive, operando attraverso le specifiche API di accessibilità. Ogni sviluppatore può, teoricamente, scegliere liberamentei tasti che vuole utilizzare per muoversi all'interno del widget ma, tuttavia, è consigliabile utilizzare le pratiche raccomandate per cui si consiglia di usare gli stessi comandi da tastiera di un widget di controllo dei comuni stistemi GUI come Microsoft Windows, Apple e Unix.

Javascript può usare sia il metodo focus() sia la proprietà aria-activedescendant per creare un tabindex itinerante. Se si utilizza la prorietà di WAI-ARIA le cose sono più semplici. E' comunque il browser che si occupa della gestione dei cambiamenti di focus.

[39]

3.7.4.6 Quali sono i passi per definire una struttura navigazionale logica?

1. Identificare la struttura logica della pagina.
2. Implementare la struttura logica nel markup.
3. Etichettare ogni regione.
4. Assegnare ruoli Landmark ad ogni regione.
5. Se c'è un Landmark di tipo "application" e il testo descrittivo è disponibile, allora includere in questo Landmark l'attributo aria-describedby in modo da associare l'applicazione al testo statico.

6. E’ possibile creare un Landmark personalizzato utilizzando una regione generica. Anche se non è indispensabile dare un’etichetta personalizzata a regioni con un header, bisogna prevedere comunque un titolo che garantisca che il nome della regione sia accessibile a tutti gli utenti. Il ruolo “region” è generico e deve essere usato solamente se un ruolo più specifico non è applicabile. La tecnologia assistiva tratterà questo ruolo come una regione “complementary”.
7. Indicare alla tecnologia assistiva chi controlla la navigazione da tastiera.

[39]

3.7.4.7 Come si implementano le regioni attive?

Le regioni attive sono parti della pagina web in cui avvengono dei cambiamenti e consentono alle tecnologie assistive di essere informati degli aggiornamenti senza far perdere l’utente. Per fare questo, le regioni attive suggeriscono alle tecnologie assistive il modo in cui processare i cambiamenti. Sono le tecnologie assistive che sono responsabili della gestione di questi aggiornamenti e di consentire agli utenti di ignorare questi suggerimenti. Il progetto WAI-ARIA ha quindi creato una sezione dove spiega come utilizzare queste regioni ed elenca tutti i dettagli su come applicare le varie proprietà, in modo da aiutare gli sviluppatori a progettare regioni attive che prevedano una buona esperienza di navigazione anche per gli utenti che utilizzano le tecnologie assistive.

1. Identificare le regioni attive;
2. Verificare se nell’applicazione si possono utilizzare alcune delle speciali regioni attive, dotate di proprietà predefinite, previste da WAI-ARIA;
3. Decidere la priorità di ogni regione attiva utilizzando la proprietà aria-live:

off	**cambiamenti banali da non annunciare all’utente**
polite	l’utente è avvisato dei cambiamenti solo quando è inattivo
assertive	modifiche da notificare il prima possibile ma senza interrompere immediatamente l’utente
rude	modifiche da notificare immediatamente

4. Decidere quale parte del contesto è necessaria per ogni aggiornamento, utilizzando la proprietà aria-atomic (true o false);
5. Decidere quali tipi di cambiamenti sono rilevanti per ogni regione attiva, utilizzando aria-relevant.

[29] [32]

3.7.4.8 Quali sono i passi da seguire per costruire un widget WAI-ARIA?

1. Scegliere il role dalla taxonomia ARIA;
2. A partire dal ruolo, ottenere l'elenco di stati e proprietà consultando la specifica WAI-ARIA;
3. Stabilire la struttura del widget nel markup (parent/child);
4. Ripetere i passi 1-3 per tutti i i nodi;
5. Stabilire la navigazione da tastiera;
6. Applicare e gestire gli stati ARIA necessari in risposta agli eventi di input dell'utente;
7. Sincronizzare la UI (user interface) visiva con stati e proprietà accessibili per supportare gli user agents;
8. Mostrare e nascondere sezioni in un widget;
9. Supportare l'accessibilità di base;
10. Stabilire le relazioni ARIA tra tutti i widget dell'applicazione.

[29]

3.7.4.9 Come costruire un'applicazione accessibile con WAI-ARIA

1. Ogni elemento o widget deve possedere una semantica completa e corretta che descriva completamente il suo comportamento, in modo tale da poter essere interpretato dalla tecnologia assistiva;
2. Tutti i componenti interattivi devono essere utilizzabili tramite tastiera: supportare una navigazione da tastiera completa ed usabile;
3. Usare markup nativo quando possibile;

4. Applicare i ruoli in modo appropriato;
5. Costruire relazioni;
6. Impostare stati e proprietà in risposta agli eventi;
7. Sincronizzare l'interfaccia visuale con l'interfaccia accessibile.

[29]

3.8 Interazione Screen Readers - Ajax

3.8.1 Introduzione

L'aumento dell'uso di Ajax per aggiornare dinamicamente il contenuto senza ricaricare completamente la pagina ha portato a problemi di accessibilità soprattutto per gli utenti che utilizzano lo screen reader.

3.8.2 Come funziona lo screen reader

Uno screen reader (lettore di schermo) è un'applicazione software che identifica ed interpreta il testo mostrato sullo schermo di un computer, presentandolo ad un utente disabile tramite sintesi vocale o attraverso un display braille.

Sul mercato esistono diversi software, differenti per prestazione e per prezzo, e quindi la scelta non è facile. Tuttavia possiamo dire che il più utilizzato è Jaws.

3.8.3 Jaws

Jaws (acronimo di Job Access With Speech) è lo screen reader prodotto dalla Freedom Scientific.

Fu originariamente creato per il sistema operative MS-DOS ma acquistò popolarità solo quando nel 1992 venne acquistato da Microsoft. Nel tempo è diventato sempre più adatto per tale sistemato operative e sempre più performante. L'ultima versione disponibile di Jaws (quella che supporta le WAI-ARIA) è la 10.0 sviluppata nel novembre 2008 (rilasciata in versione beta in agosto).

[18]

3.8.4 I cursori di Jaws

I cursori servono per dire allo screen reader “cosa deve leggere”. Sono 3 i cursori disponibili con Jaws:

1. Cursore PC. E’ il cursore del sistema operativo.
2. Cursore Jaws. Rappresenta l’emulatore del mouse. Permette di muoversi liberamente nello schermo ma solo all’interno dell’applicazione attiva.
3. Cursore PC Virtuale. E’ il cursore che si utilizza per il web. La pagina internet viene vista come un componente editazionale e si possono usare i tasti di scorrimento per navigare all’interno della pagina internet.

[14][15] [18]

3.8.5 Il buffer virtuale nelle varie versioni di Jaws

Gli screen reader utilizzano un buffer virtuale per permettere agli utenti di interagire con i contenuti di una pagina web. Nelle varie versioni di Jaws che si sono susseguite sono state introdotte funzionalità che rendono più semplice utilizzare lo screen reader in applicazioni Ajax.

Lo screen reader cattura un’istantanea della pagina web (una copia del DOM della pagina) e la mette nel buffer virtuale. Senza la memoria virtuale lo screen reader può accedere alle parti della pagina che possono ricevere il focus da programmi utente non assistiti (ancora ed elementi dell’interfaccia) ma non può interagire con gli altri elementi presenti nel contenuto. Questo vale per le versione di Jaws fino alla 7.1. E’ chiaro che comprendere cosa sia e come funzioni il buffer virtuale è un requisito necessario per tutti gli utenti di screen reader dato che, attivandolo e disattivandolo in modo opportuno, è possibile aggiornare il contenuto dell’applicazione. Lo screen reader viene quindi forzato ad aggiornare il buffer virtuale deve rileggere l’intero documento per accedere ai contenuti aggiornati: la risposta in tempo reale tipica di Ajax viene a mancare completamente.

In Jaws 7.1 e successivi la Freedom Scientific ha introdotto una miglioria fondamentale: tutte le modifiche dinamiche risultanti dall’attivazione di un link con la tastiera da parte dell’utente aggiornano il buffer virtuale, che risulta così sincronizzato col contenuto sullo schermo. Si tratta di un notevole miglioramento sia per quanto riguarda le modifiche di contenuto dipendenti da Javascript sia quelle indipendenti. Il problema di latenza (in particolare in relazione ad Ajax) non è

più un problema e l'utente non deve più fare niente per il contenuto modificato per essere aggiornato nella vista dello screen reader.

[14] [15] [16]

3.8.6 Cosa c'è di nuovo in Jaws 10.0?

In generale c'è da dire che la visualizzazione delle pagine statiche è migliorata in modo significativo rispetto al precedenti versioni di Jaws. La lettura diventa un'esperienza più piacevole per tutti in quanto non c'è più il problema della lettura del testo insieme a certe strutture HTML. [18]

3.8.7 Supporto ARIA Live Region

Jaws 10 può annunciare aggiornamenti nelle regioni attive di una pagina web quando si utilizza Firefox 3 o superiore e in Internet Explorer 8, quando verrà rilasciato ufficialmente. Questa funzionalità è attivata di default ed è possibile stoppare Jaws da questo annuncio di aggiornamenti (premendo INSERT+V). [18]

3.8.8 Supporto ARIA Landmark

La combinazione di tasti INS+CTRL+tasto virgola ora supporta la funzione di selezione di un Landmark. I Landmark sono tag ARIA che definiscono la struttura di una pagina web. Questa funzionalità fornisce quindi informazioni utili ad un utente jaws quando naviga su una pagina che supporta i tag Landmark ARIA. Inoltre le combinazioni tasto virgola e SHIFT+tasto virgola ora stanno a indicare Passa al prossimo Landmark e Torna al Landmark precedente. Questa funzionalità si può utilizzare con Mozilla 3 e successivi e con Internet Explorer 8 quando sarà rilasciato ufficialmente. [18]

3.8.9 Modalità Auto Forms

Tradizionalmente la modalità Forms in Jaws deve essere attivata manualmente quando bisogna inserire testo in campi editabili o selezionare valori di caselle in una pagina web. Per default Jaws è in modalità virtuale che consente di utilizzare la navigazione rapida dei tasti come F per il campo form successivo e E per la casella successiva. Quando c'è bisogno di interagire con una form per scrive del testo bisogna disattivare manualmente la modalità Forms usando il tasto ENTER. E' possibile attivare o disattivare la modalità Forme usando ESC o il tasto NUM PAD PLUS. La modalità Auto Forms si accenderà automaticamente su modalità Forms quando ci si sposta in una casella di controllo utilizzando TAB o i tasti freccia. Questa funzionalità rende molto più facile ed

intuitivo interagire con i campi della form di una pagina web, senza dover utilizzare tasti extra. Quando si premono i tasti freccia o TAB per muoversi in un campo editabile, si sentirà un suono che indicherà che in quel momento si può scrivere senza dover premere prima il tasto ENTER. Quando si lascia il controllo utilizzando un tasto freccia, la modalità Forms si spegne e ciò permetterà di continuare a navigare con facilità di nuovo nella modalità virtuale. Di default la modalità Auto Forms è attivata. Ci sono due modi per disattivare questa funzionalità. Premere INSERT+V e cambiare la relativa opzione o usare il Configuration Manager. [18]

3.8.10 Modalità Application

Quando il ruolo ARIA nel tag <body> è impostato come "application" Jaws tratterà la pagina web come se si trattasse di una normale applicazione stand-alone. Questo significa che il buffer virtuale che Jwas tipicamente usa per mostrare le pagine web sarà spento, e l'utente potrà navigare nella pagina utilizzando la tabulazione o altri tasti che vengono normalmente utilizzati nelle applicazioni stand-alone per facilitare l'accesso da tastiera. La pagina web non appare all'utente come un documento web ma piuttosto come un'applicazione con form di controllo con campi editabili, pulsanti e liste. [18]

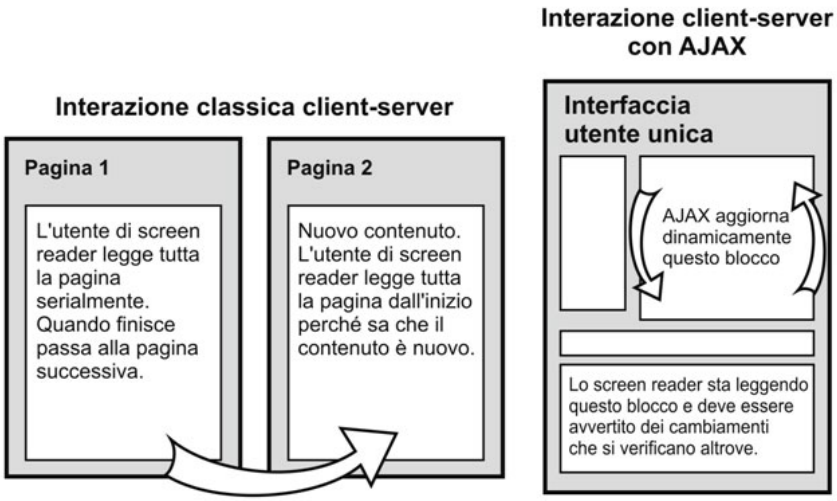
3.8.11 Problemi con le applicazioni Ajax

Gli script negli screen reader funzionano in modo incostante. Ci sono approcci frammentari che lavorano per uno o più dispositivi ma non un approccio globale che riguarda tutti. Non è possibile far lavorare Ajax con tutte le tecnologie assistive e non è possibile far lavorare Ajax con i vecchi browser. Ajax è inconsistente perché non tutti gli screen reader sono in grado di rispondere a onreadystatechange dell'oggetto XMLHttpRequest e perché non c'è focus sull'elemento che è cambiato.

I problemi degli screen readers con le applicazioni Ajax sono dunque:

- ✓ capire SE qualcosa è cambiato nella pagina;
- ✓ capire DOVE è cambiato qualcosa nella pagina.

[13] [14]



3.8.12 Soluzioni per l'accessibilità

Abbiamo già visto come il primo punto viene risolto nelle ultime versioni di Jaws dove il contenuto del buffer virtuale non deve essere più aggiornato manualmente ma anche gli utenti di screen reader vengono informati automaticamente dei cambiamenti che avvengono nella pagina web.

Per far capire allo screen reader dove è cambiato qualcosa nella pagina bisogna fare in modo che lo possa leggere direttamente i contenuti dinamici anche quando riguardano elementi che non possono ricevere il focus secondo le specifiche HTML 4 e XHTML 1. Per trovare una soluzione standardizzata a questi ed altri problemi di accessibilità delle applicazione dinamiche il W3C ha realizzato una "Tabella di marcia per i contenuti web dinamici accessibili" introducendo come soluzione ai problemi di accessibilità proprio l'utilizzo di ruoli, stati e proprietà di WAI-ARIA. Infatti la presenza di una struttura semantica della pagina web permette a ciascun sito di supportare lo stesso standard e permette alla tecnologia assistiva di fornire una consistente esperienza di navigazione.

[14] [21]

3.9 AsxJAX

3.9.1 Come nasce AxsJAX

AsxJAX (Access JAX) è un frame work open source sviluppato da Charles L Chen e T. V. Raman che inserisce le proprietà ARIA nelle applicazioni web in modo da renderle accessibili alle tecnologie assistive. L'idea di sviluppare questo frame work è nata dopo che Chen ha inserito il supporto ARIA alle regioni attive su Google Reader, che già era dotato di un buon supporto da tastiera. [5] [35]

3.9.2 Perchè si inseriscono le proprietà ARIA?

Abbiamo già detto che i problemi con le applicazioni web 2.0 sono fondamentalmente 2: trasmettere ruolo, stato e proprietà alle tecnologie assistive e fornirgli un feedback in modo da informare l'utente dei cambiamenti avvenuti. WAI-ARIA risolve questi problemi fornendo un feedback attraverso le regioni attive di WAI-ARIA, supportate da Jaws, Window-Eyes e Firevox. [35]

3.9.3 Come si inseriscono le proprietà ARIA

AxsJAX lavora inserendo automaticamente le proprietà ARIA nel DOM, basandosi sul design patterns. Le applicazioni che al momento supportano AxsJAX sono:

- Google Web Search
- Google Gmail
- Google Calendar
- Google Reader
- Google Scholar
- Google Books
- Google Finance
- Google Health
- Jawbreaker Game
- The xkcd Web Comic
- www.Amazon.com
- www.Weather.com
- Orkut

Esistono diversi modi per inserire le proprietà ARIA in una applicazione web 2.0:

- approccio cross-browser usando l'AxsJAX bookmarklet, cioè attivare il bookmarklet ogni volta che si visita una pagina che supporta il frame work;
- usare il componente aggiuntivo di Firefox GreaseMonkey che inserisce automaticamente le proprietà ARIA quando la pagina viene caricata;
- Firevox inserisce automaticamente le proprietà ARIA utilizzando il framework AxsJAX se l'opzione "Use site specific enhancements" è abilitata. [35]

3.9.4 Come usare AxsJAX

Quello che occorre per utilizzare il frame work è un browser che supporta WAI-ARIA (Firefox) e una tecnologia assistiva che supporta WAI-ARIA (Jaws). Se la tecnologia assistiva ha un buffer virtuale, bisogna assicurarci che esso sia disattivato. [35]

3.9.5 Obiettivi di AxsJAX

L'obiettivo primario è quello di sfruttare gli stessi vantaggi dell'uso di JavaScript nei browser in modo da creare soluzioni di accessibilità potenti e flessibili.

All'inizio AxsJAX riguardava solamente le applicazione Google ma, se si guarda il design pattern, possiamo vedere come l'utilizzo di modelli comuni permette di renderli riutilizzabili in qualsiasi applicazione sviluppata sul web, sfruttandoli per inserire miglioramenti di accessibilità.

L'obiettivo più a lungo termine è quello di creare una grande comunità costruita su una piattaforma aperta per migliorare l'accessibilità delle applicazioni web 2.0. [5] [35]

3.10 ACTF Accessibility Tools Framework

L'interesse sempre maggiore per i nuovi standard di accessibilità ha causato un acceleramento nell'attività di ricerca e di sviluppo per la prossima generazione di strumenti per l'accessibilità.

Il progetto ACTF è stato realizzato da Chieko Asakawa (IBM) e Mike Paciello (The Paciello Group). ACTF è una collezione di strumenti sviluppati da IBM per mezzo dei quali gli sviluppatori possono creare strumenti di accessibilità in modo semplice ed efficace, grazie ai componenti di accessibilità riutilizzabile, al design standardizzato e all'interfaccia di programmazione dell'applicazione che il sistema offre. ACTF offre agli sviluppatori la possibilità di utilizzare vari strumenti per l'accessibilità.

IBM contribuirà con una certa quantità di codice iniziale che sarà incentrato sugli aspetti fondamentali del progetto ACTF. Esso include:

- componenti del frame work
- motore di validazione estensibile
- motore di visualizzazione per utenti non vedenti
- implementazioni esemplari
- implementazione di interfaccia alternativa per i contenuti multimediali

[33]

3.11 Linee guida per progettare un Ajax accessibile

- Assicurarsi che il sito funzioni anche con JavaScript disabilitato, tenendo presente che l'Ajax va usato solo per potenziare le funzionalità;
- Fornire sempre un alternativa quando Ajax non può essere reso accessibile
- Progettare l'applicazione a strati (contenuti, struttura, presentazione e comportamenti). Pianificare Ajax all'inizio ma implementarlo solo alla fine.
- Informare che la pagina verrà aggiornate dinamicamente, dando la possibilità di aggiornare manualmente la pagina (tramite un bottone, un link o salvando la scelta nelle preferenze)
- Evidenziare le aree che hanno subito le modifiche (utilizzo della tecnica FAT)
- Non spostare il focus nella zona dell'interfaccia che ha subito le modifiche senza avvertite nella maniera adeguata la tecnologia assistiva
- Uso di valori negativi del tabindex per gestire il problema del focus
- Utilizzare il markup WAI-ARIA
- Usare interazioni indipendenti dal dispositivo
- Rispettare integralmente gli standard

3.12 Caso applicativo

Lo studio dell'accessibilità delle applicazioni Ajax si è concluso con la riprogettazione di un sito museale. Scopo del lavoro è quello di inserire script Ajax per migliorare le funzionalità dell'applicazione, mantenendo l'accessibilità utilizzando gli attributi di WAI-ARIA.

3.12.1 Descrizione generale dell'applicazione di partenza

L'applicazione utilizzata è un sito museale realizzato dal laboratorio HIIS ISTI-CNR per il museo del marmo e dei beni culturali della città di Carrara. Il sito presenta tutte le informazioni tradizionali del museo (orari e tariffe, servizi, news, storia del museo e statistiche visitatori) e permette di visitare tutte le opere presenti nel museo adattando l'interfaccia di ricerca e di visualizzazione a seconda del tipo di utente che sta accedendo al sito (studente, turista ed esperto).

L'applicazione è raggiungibile all'indirizzo http://urano.isti.cnr.it:8880/museo/home.php. Si tratta di un sito realizzato in php. Le informazioni su opere del museo e della città di Carrara e sugli autori sono ricavate da un database sql.

Il sito quindi è stato riprogettato nell'ottica di inserire script Ajax per fornire le informazioni in maniera più dinamica, soprattutto nella ricerca delle opere.

Gli script Ajax sono stati inseriti cercando di mantenere accessibile l'applicazione e l'utilizzo di WAI-ARIA ha rappresentato la migliore soluzione da utilizzare.

3.12.2 Struttura dell'applicazione

3.12.2.1 Home Page

L'home-page del sito riprogettato non si presenta molto invariata. La struttura prevede sempre un menù di navigazione principale sulla sinistra e una colonna più grande per il contenuto principale.

Il modulo per contattare il museo non si trova più in una pagina a parte ma è stato inserito in fondo al menù ed è quindi presente in ogni pagina. I dati inseriti vengono inviati al server tramite uno script Ajax ed è quindi possibile proseguire nella navigazione della pagina una volta che i dati saranno inviati, dal momento che la pagina non sarà ricaricata.

In questa pagina è stata inserita un'area in cui vengono visualizzate informazioni sulle opere presenti nel museo. Tramite script Ajax viene aggiornata la regione attiva ARIA, impostando la priorità del cambiamento a rude. Il valore live rude indica una priorità maggiore rispetto a polite, valore standard e che è stato assegnato a tutte le altre regioni attive inserire nell'applicazioni.

Questo significa che ogni volta che la regione attiva con valore rude verrà aggiornata lo Screen Reader passa immediatamente a leggerne il contenuto, interrompendo qualsiasi altra azione dell'utente.

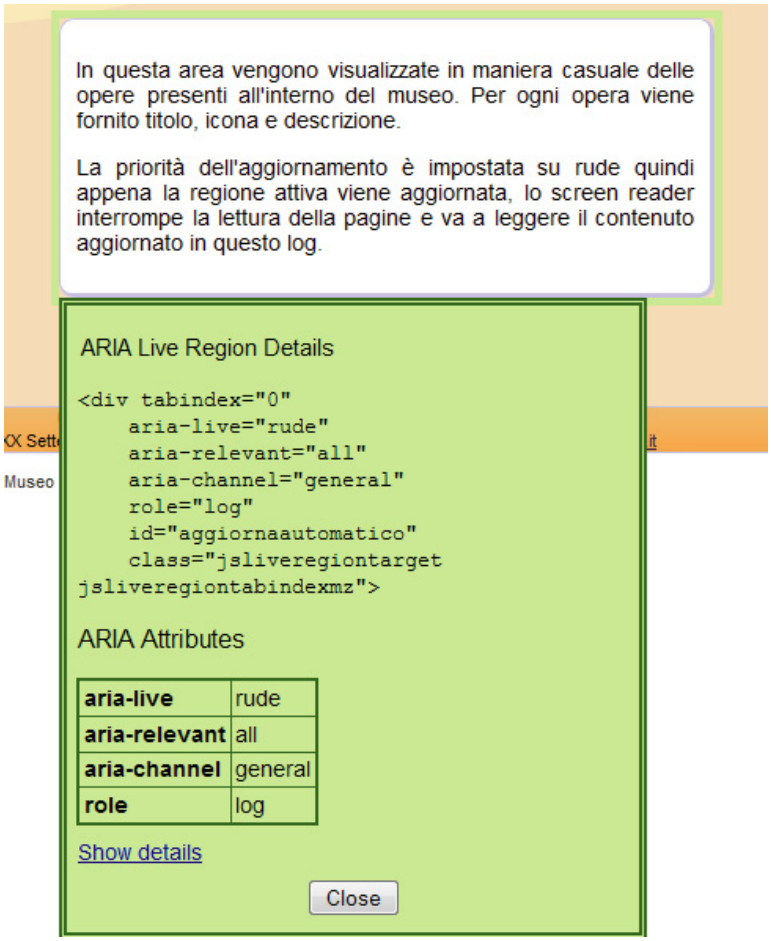


Figura 20: caratteristiche della regione attiva in cui vengono visualizzata l'anteprima sulle opere del museo

3.12.2.2 Informazioni

Nella sezione contenente le informazioni principali del museo il contenuto presentato è rimasto invariato tranne per l'aggiunta di un nuovo link (Libro degli Ospiti) in cui è possibile inserire commenti sul sito o sul museo e visualizzare tutti quelli già presenti. Inserimento e visualizzazione di commenti sono gestiti tramite tecnologia Ajax. Script che utilizzano Ajax sono stati inseriti anche nella pagina delle Statistiche dei visitatori per rendere più dinamica la visualizzazione dei dati.

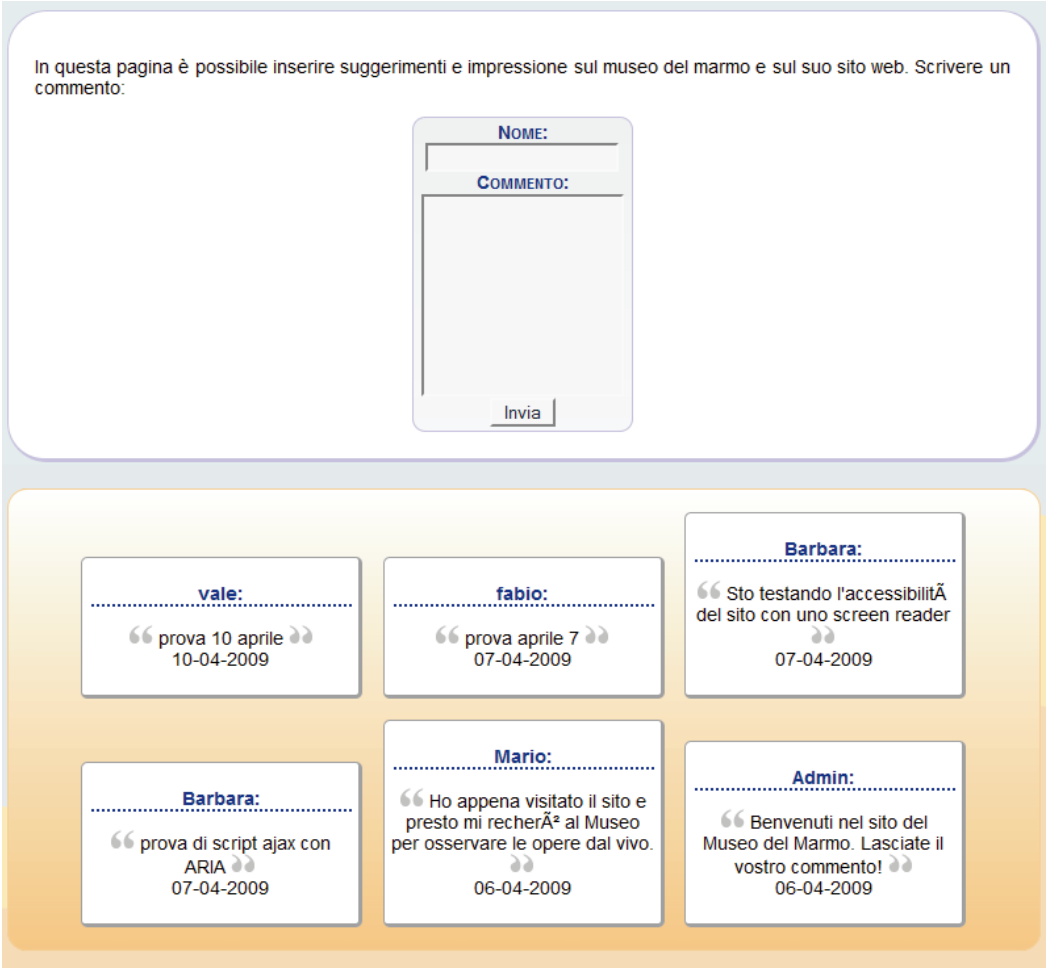


Figura 21: inserimento commento nel libro degli ospiti

3.12.2.3 Visita

Per quanto riguarda la ricerca delle opere, i profili sono stati riprogettati in modo da permettere una ricerca e una visualizzazione delle opere più dinamica.

3.12.2.4.1 Profilo Turista

Nel profilo turista è presente una lista di link che rappresentato tutte le sale del museo. Selezionando la sala interessata la pagina si aggiorna tramite uno script Ajax e visualizza:

- Localizzazione della sala nella mappa del museo;
- Informazioni generali sulla sala con possibilità di visualizzare le opere presenti.

Figura 22: profilo turista

3.12.2.4.1 Profilo Studente

Nel profilo studente è possibile ricercare le opere scegliendo di partire da autore, anno di realizzazione o definizione utilizzata.

Ogni ricerca prevede la selezione dei criteri di interesse utilizzando delle liste di selezione dinamiche.

Se consideriamo la ricerca che parte dall'autore, una volta selezionato quello di interesse, viene creata dinamicamente una nuova lista di selezione contenente le definizione utilizzate dall'autore selezionato.

Una volta scelta anche la definizione di interesse verrà visualizzata la lista di tutte le opere che soddisfano i due criteri e sarà possibile cliccare sull'icona dell'opera per avere informazioni più dettagliate.

Figura 23: profilo studente

3.12.2.4.1 Profilo Esperto

Nel profilo esperto è possibile ricercare le opere scegliendo di partire da autore, anno di realizzazione, materiale o definizione.

Se vogliamo partire ad esempio da autore, una volta selezionato l'autore di interesse, viene utilizzato uno script Ajax per fornire una serie di informazioni collegate:

- Anni in cui l'autore selezionato realizza opere;
- Definizioni utilizzate dall'autore;
- Materiali utilizzati dall'autore;
- 5 immagini casuali delle opere dell'autore.

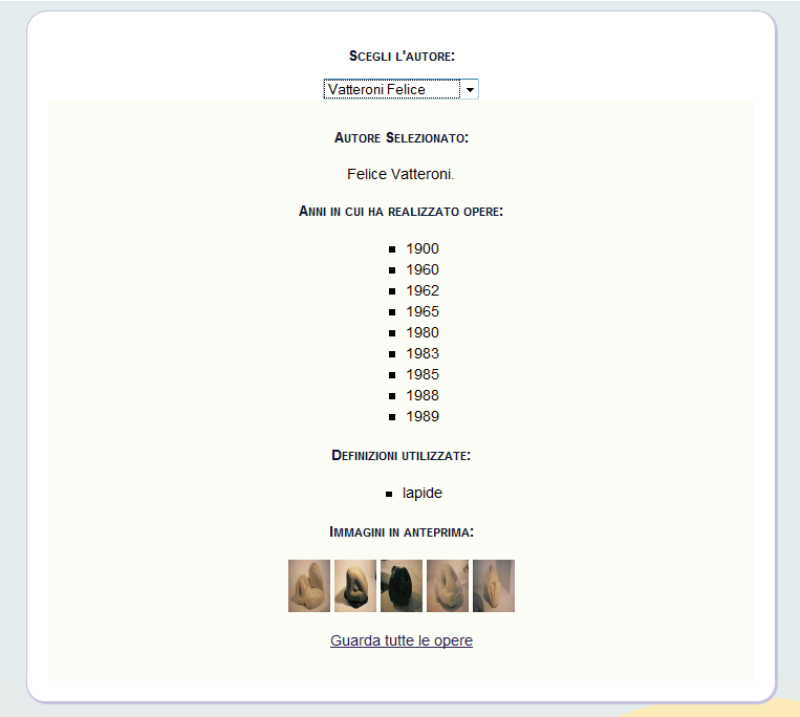


Figura 24: profilo esperto, selezione dell'autore

A questo punto si possono visualizzare titolo e icone di tutte le opere realizzate dall'autore e cliccare (o premere invio) sull'immagine per avere informazioni più dettagliate sull'opera selezionata.

Figura 25: profilo esperto, visualizzazione delle opere

3.12.2.4.1 Ricerca Rapida

Al posto del profilo personalizzato è stata inserita una sezione di ricerca rapida per effettuare delle selezioni che non sono incluse nei vari profili (turista, studente, esperto).

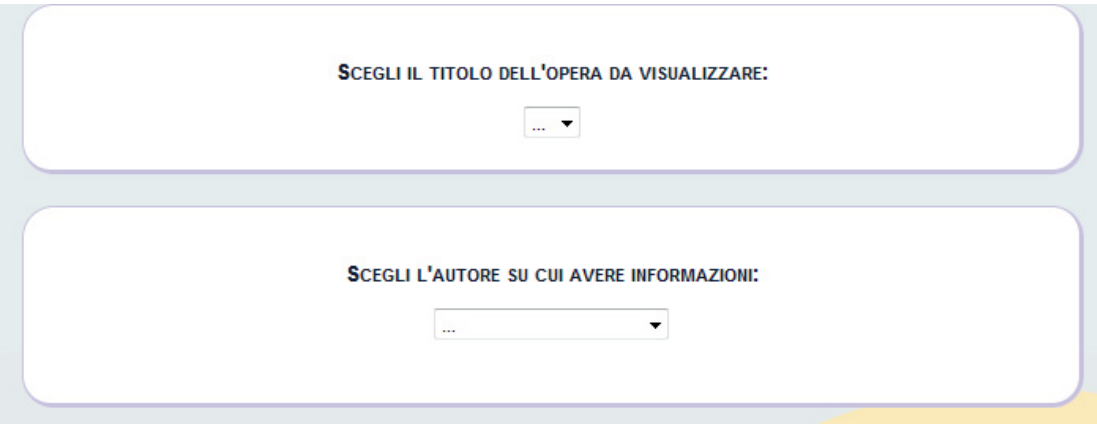


Figura 26: selezioni della ricerca rapida

La prima selezione presente permette di visualizzare tutte le informazioni di un'opera semplicemente selezionando il titolo. Le informazioni vengono fornite dinamicamente tramite script Ajax.

La seconda selezione permette di visualizzare informazioni biografiche sugli autori selezionando cognome e nome dell'autore di interesse. Anche queste informazioni vengono fornite dinamicamente tramite script Ajax.

3.12.2.4 Percorsi

Nella sezione dedicata ai percorsi del museo le informazioni sono state rese più dinamiche tramite l'utilizzo degli script Ajax.

Nel percorso "alla scoperta" è possibile visualizzare le informazioni sulle varie tappe del percorso utilizzando i bottoni mentre nel percorso didattico è possibile selezionare i punti sensibili dell'image map. La funzionalità dello script è garantita anche utilizzando il tasto di tabulazione.

Figura 27: percorso "alla scoperta"

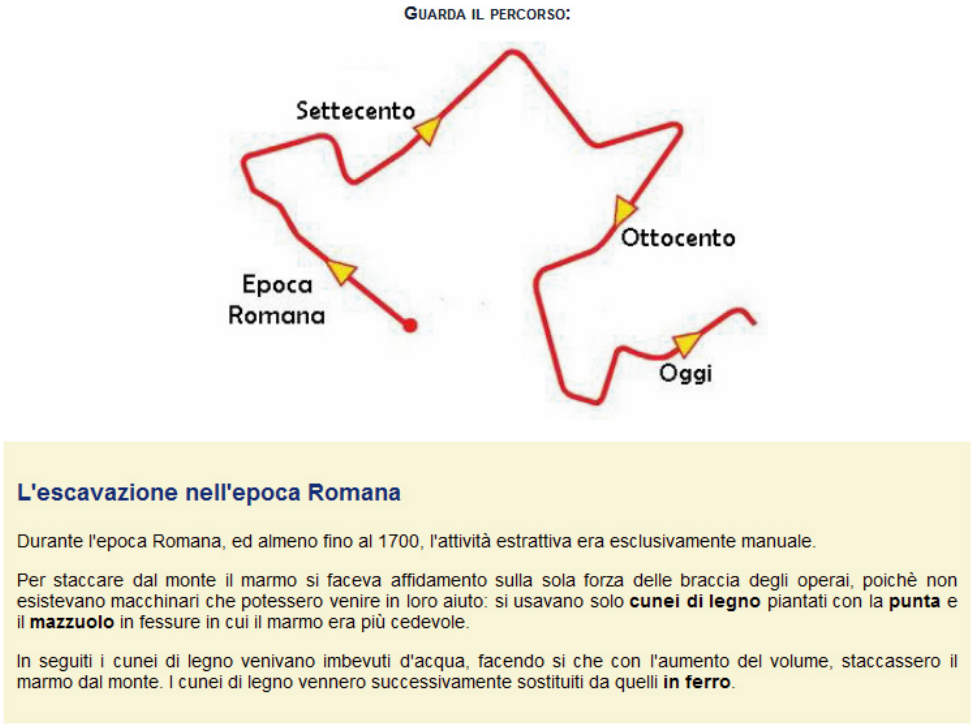


Figura 28: percorso didattico

3.12.2.5 Carrara

Anche in questa sezione l'utilizzo della tecnica Ajax va a garantire più dinamicità all'applicazione ed evita il refresh della pagina ogni volta che avvengono degli aggiornamenti.

E’ possibile visualizzare le informazioni su opere e monumenti presenti nella città di Carrara andando a selezionare l’elemento di interesse nella image map della città. Lo script Ajax andrà a visualizzare le informazioni presenti nella regione attiva. Le stesse informazioni si possono ottenere andando a scegliere dalle liste di selezione che si trovano sotto la mappa.

3.12.2.6 Visualizzazione Opere

La visualizzazione delle informazioni sull’opera varia a seconda del profilo che scegliamo per effettuare la ricerca.

Le informazioni visualizzate nel profilo turista sono:

- Titolo
- Immagine
- Descrizione
- Autore
- Materiale

Figura 29: informazioni visualizzate per il profilo turista

Le informazioni visualizzate nel profilo studente sono:

- Titolo
- Immagine
- Descrizione
- Autore
- Materiale
- Definizione

Le informazioni visualizzate nel profilo esperto sono:

- Titolo
- Immagine
- Descrizione
- Autore
- Dimensioni (altezza, larghezza, profondità, diametro)
- Materiale
- Definizione
- Collezione di appartenenza

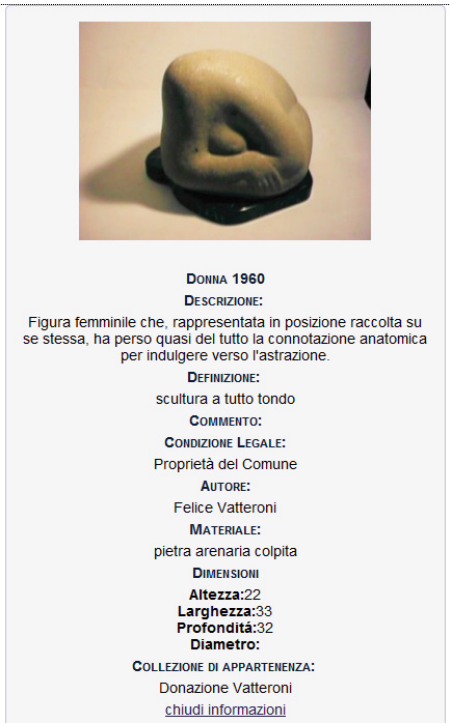


Figura 30: informazioni visualizzate per profilo esperto

Le informazioni visualizzate nel profilo di ricerca rapida (ricerca per titolo) sono:

- Autore
- Descrizione
- Cronologia
- Definizione
- Materiale
- Dimensioni
- Collezione di appartenenza
- Località
- Soggetto
- Ubicazione
- Informazioni storiche

Figura 31: visualizzazione informazioni per ricerca rapida (titolo)

Le informazioni visualizzate nel profilo di ricerca rapida (ricerca per autore sono:

- Biografia
- Riferimento bibliografico
- Luogo di nascita
- Luogo di morte

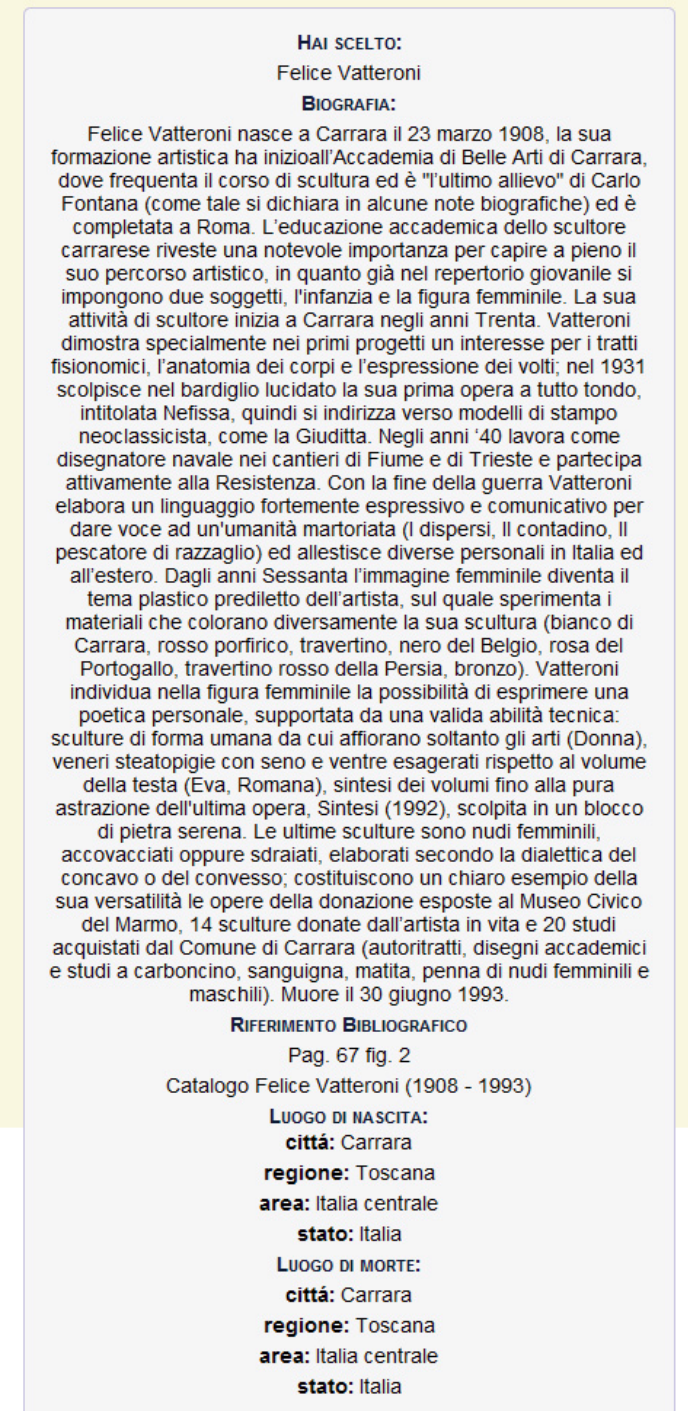


Figura 32: visualizzazione informazioni per ricerca rapida (autore)

Le informazioni visualizzate nella mappa virtuale di Carrara sono:

- Indirizzo
- Immagine
- descrizione
- autore

Figura 33: informazioni visualizzate per mappa virtuale

Le informazioni visualizzate nella anteprima delle opere del museo presente nell'home-page sono:

- icona
- titolo
- descrizione

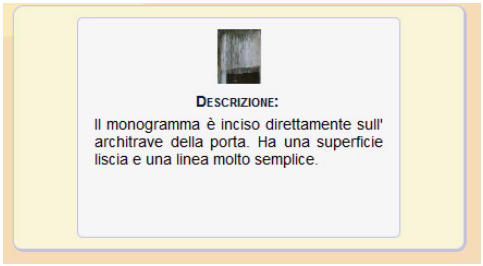


Figura 34: informazioni visualizzare per anteprima opere

3.12.3 Interventi sull'applicazione di partenza

3.12.3.1 Inserimento di script Ajax

L'applicazione già esistente è stata implementata inserendo script Ajax per:

- gestire il modulo per contattare il Museo;
- visualizzare informazioni di anteprima sulle opere nella home-page;
- Visualizzare dinamicamente le statistiche dei visitatori;
- Visualizzare una lista degli ospiti del museo;
- Visualizzare dinamicamente i video;
- Visualizzare dinamicamente le informazioni sul Percorso "alla Scoperta"
- Visualizzare dinamicamente le informazioni dei vari profili visitatori e della sezione Ricerca Rapida.

Tutti gli script sono stati inseriti cercando di mantenere l'accessibilità dell'applicazione. Il modo migliore per garantire l'accessibilità è l'inserimento dei ruoli, degli stati e delle proprietà WAI-ARIA, inserirti in tutta l'applicazione, sia per gli elementi dinamici che in quelli statici. L'uso di WAI-ARIA permette infatti di far utilizzare l'applicazione agli utenti che utilizzano lettori di schermo e navigazione da tastiera.

Quindi tutti gli script realizzati con Ajax utilizzano le regione attive di ARIA per far in modo che la parte aggiornata della pagina venga letta anche dallo Screen Reader.

Quando l'utente invia una richiesta al server, il server web cerca di recuperare i dati richiesti. Nel frattempo l'utente può continuare ad interagire con la pagina ed altre richieste Ajax possono essere inviate al server web, in quanto è in grado di gestirle contemporaneamente. Una volta ottenuti, questi dati vengono aggiornati all'interno di una regione attiva ARIA. Lo Screen Reader interpreta le proprietà delle regioni attive e può quindi annunciare i cambiamenti, senza richiedere l'aggiornamento forzato della pagina o la rilettura dell'intera pagina per accedere al contenuto aggiornato.

Per creare lo script Ajax che visualizza le informazioni nel profilo esperto relative al materiale, si deve realizzare per prima cosa il modulo con la lista di selezione e il div in cui verranno aggiornati i dati richiesti al server web.

```
<form name="form_sceglimat" method="get">
```

```
<select role="combobox" name="mat" onChange="mostramat(this.value); return
false;">

 <option>...</option>

<?php include ("connessione.php");

$query="SELECT * FROM materialdetails WHERE language='it'";

$ris = mysql_query($query);

while ($array_ris = mysql_fetch_array($ris))

{

$ris_C3 = trim($array_ris['nome_mat']);

$ris_C4 = $array_ris['material'] ;

?>

<option value="<?php echo "$ris_C4" ;?>"> <?php echo "$ris_C3"?> </option>

<?php

}

?>

</select>

</form>

<div id="resultMat" aria-channel="notify" aria-atomic="false" aria-live="polite"
role="log"></div>
```

Utilizzando gli attributi ARIA, questo div diventa una regione attiva (log) della pagina e tutte le proprietà ARIA associate a questa regione sono interpretate dal lettore di schermo che potrà così notificare ogni cambiamento che avverrà in questa zona. A seconda delle proprietà ARIA che verranno applicate alla regione, la notifica varierà. Nel nostro caso i cambiamenti della pagina avvengono sempre in seguito ad un'azione dell'utente (quasi sempre sull'evento onChange in seguito alla selezione su lista) e quindi è stato scelto di utilizzare la proprietà ARIA- live impostata su assertive, ad indicare dei cambiamenti che sono importanti all'interno della pagina ma che non devono interrompere l'attività dell'utente.

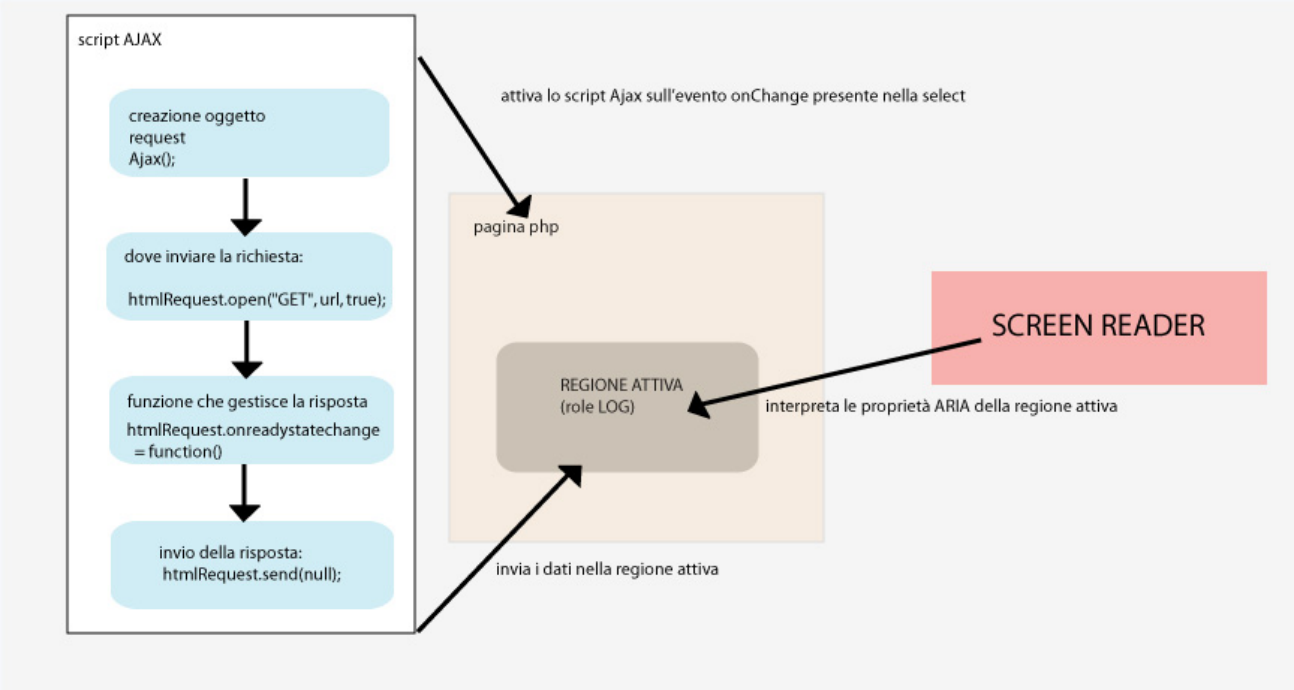


Figura 35-: schema dell'interazione tra Screen Reader, Ajax e ARIA

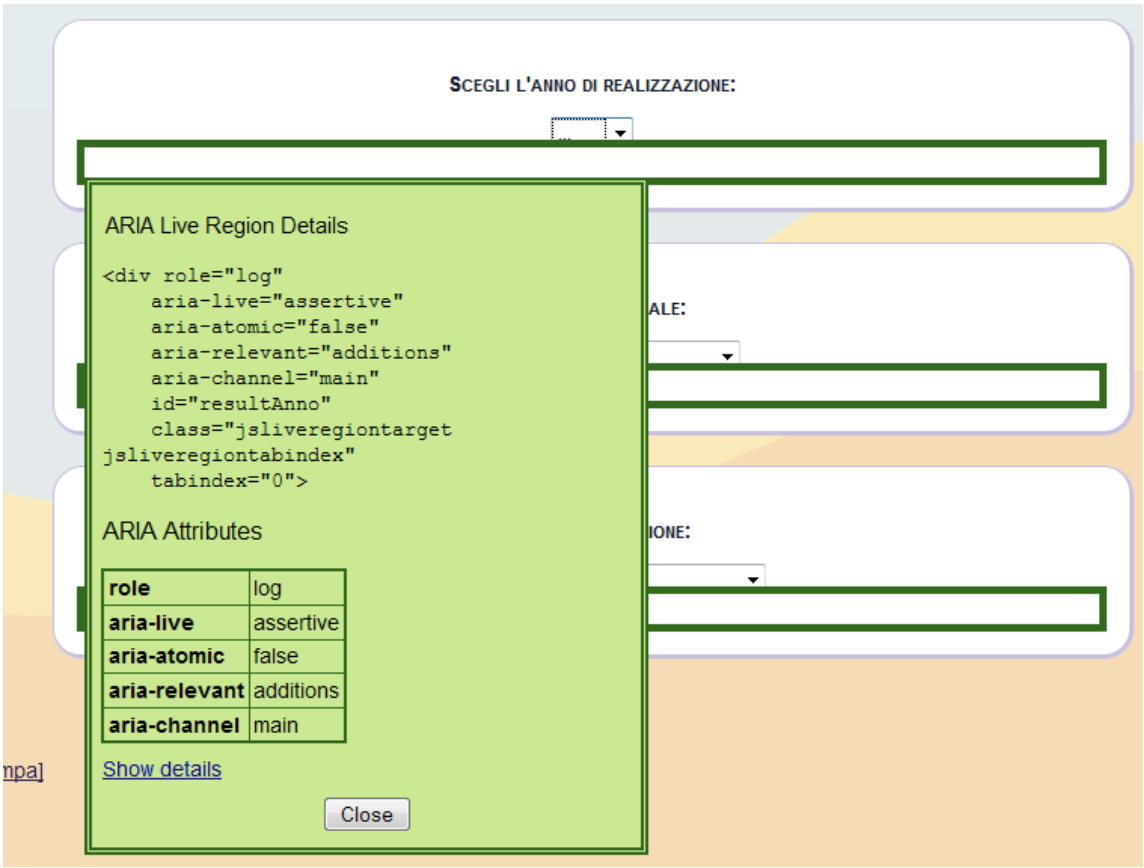


Figura 36: individuazione delle regione attive con juicystudio toolbar

La funzione JavaScript che viene richiamata sull’evento onChange() della select è contentuta in un file JavaScript esterno:

```
// funzione per mostrare i dati
function mostramat(str) {
  var url="mostraMat.php";
  url=url+"?q="+str;
  url=url+"&sid="+Math.random();
  htmlRequest = ajax();
  // controllo nel caso in cui non possa richiamato l'oggetto Xmlhttp
  if (htmlRequest==null){
    alert ("Il browser non supporta richieste HTTP");
    return;
  }
  htmlRequest.onreadystatechange = function(){
    // Restituisce lo stato della richiesta
    if(htmlRequest.readyState == 4){
      // Restituice il corpo della risposta come stringa
                  var mostraMat = document.getElementById("resultMat");
      mostraMat.innerHTML = htmlRequest.responseText;
                  fadeUp(mostraMat,255,179,0);
    }
  }
  // chiamata della pagina PHP che estrae i records
  htmlRequest.open("GET", url, true);
  htmlRequest.send(null);
}
```

All’interno di questo file JavaScript vengono richiamate altre 2 funzioni che sono contenute in relativi file JavaScript:

- Ajax() : funzione che crea l’oggetto XMLHttpRequest, fondamentale per inviare richieste asincrone al server;
- fadeUp(): funzione che crea l’effetto di fade sulla zona aggiornata della pagina.

Vediamo in particolare il file che crea l’oggetto XHR, che verrà riutilizzato per tutto gli script del sito che fanno uso di Ajax:

```
// funzione per la chiamata dell'oggetto XMLHttpRequest
function ajax(){
  var ajaxRequest;
  try{
    // controllo per i browser diversi da IE
    ajaxRequest = new XMLHttpRequest();
  }catch (e){
    // controllo per IE
    try{
      ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");
    }catch (e){
      try{
        ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");
      }catch (e){
        // controllo per i browser che non supportano l'XMLHttpRequest
        alert("Il browser non supporta questo guestbook");
        return false;
      }
    }
```

```
  }

  return ajaxRequest;

}
```

Rimane da vedere la pagina php contenente i dati da visualizzare:

```
<?php

include ("connessione.php");

$risultato=$_GET["q"];

$sql2 = mysql_query("SELECT * FROM materialdetails WHERE language='it' AND
material='$risultato' group by nome_mat") or die (mysql_error());

if(mysql_num_rows($sql2)>0){

while ($new=mysql_fetch_array($sql2))

{

echo "<p><h4 class='titolo'>Materiale selezionato:</h4> $new[nome_mat] </p>";

}}

else { //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}?>

<h4 class="titolo">Autori che la utilizzano:</h4>

<ul class="ulpuntato">

<?php

$sql4 = mysql_query("SELECT * FROM materialdetails JOIN exhibitsmaterials on
materialdetails.material = exhibitsmaterials.material JOIN exhibits on
exhibitsmaterials.exhibit = exhibits.id JOIN exhibitdetails ON exhibits.id =
exhibitdetails.exhibit JOIN  artworkauthor ON exhibitdetails.exhibit =
artworkauthor.artwork JOIN authors on artworkauthor.author = authors.id

where materialdetails.material='$risultato' group by authors.id") or die
(mysql_error());

if(mysql_num_rows($sql4)>0){
```

```
while ($new=mysql_fetch_array($sql4))

{

echo "<li> $new[name] $new[surname]  </li>";

}}

else { //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}?>

</ul>

<h4 class='titolo'>Anni in cui viene usato:</h4>

<ul class="ulpuntato">

<?php

$sql5 = mysql_query("SELECT * FROM materialdetails JOIN exhibitsmaterials on
materialdetails.material = exhibitsmaterials.material JOIN exhibits on
exhibitsmaterials.exhibit = exhibits.id JOIN timespans ON
exhibits.chronology=timespans.id

where  materialdetails.material='$risultato' group by startYear") or die
(mysql_error());

if(mysql_num_rows($sql5)>0){

while ($new=mysql_fetch_array($sql5))

{

echo "<li>  $new[startYear]  </li>";

}

}

else { //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}?>

</ul>
```

```
<h4 class='titolo'>Definizione utilizzata:</h4>

<ul class='ulpuntato'>

<?php

$sql3 = mysql_query("SELECT * FROM materialdetails JOIN exhibitsmaterials

on materialdetails.material = exhibitsmaterials.material JOIN exhibits

on exhibitsmaterials.exhibit = exhibits.id JOIN artworkartworktype

on exhibits.id = artworkartworktype.artwork JOIN artworktypedetails

ON artworkartworktype.artworkType = artworktypedetails.artworkType

where materialdetails.language = 'it' AND artworktypedetails.language = 'it' AND
materialdetails.material='$risultato' group by artworktypedetails.name") or die
(mysql_error());

if(mysql_num_rows($sql3)>0){

while ($new=mysql_fetch_array($sql3))

{

echo "<li> $new[21] </li>";

}

}

else { //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}

?>

</ul>

<p><h4 class='titolo'>Immagini in anteprima:</h4></p>

<?php

include ("connessione.php");

$risultato=$_REQUEST['q'];
```

```
$sql = mysql_query("SELECT * FROM materialdetails JOIN exhibitsmaterials on
materialdetails.material = exhibitsmaterials.material JOIN exhibits on
exhibitsmaterials.exhibit = exhibits.id JOIN exhibitdetails ON exhibits.id =
exhibitdetails.exhibit JOIN

artworkauthor ON exhibitdetails.exhibit = artworkauthor.artwork WHERE
materialdetails.material='$risultato' AND

exhibitdetails.language='it' AND materialdetails.language = 'it' order by RAND()
LIMIT 0,5") or die (mysql_error());

if(mysql_num_rows($sql)>0){

while ($new=mysql_fetch_array($sql))

{

?>

<img src="images/icon<?php echo "$new[id].jpg"?>"/>

<?php

}

}

else{ //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}?>

<?php

$risultato=$_REQUEST['q'];

$sql7 = mysql_query("SELECT * FROM materialdetails

WHERE language='it' AND material='$risultato' group by nome_mat") or die
(mysql_error());

if(mysql_num_rows($sql7)>0){

while ($new=mysql_fetch_array($sql7))

{
```

```
 echo"<a href='?q=$risultato' id='autop4' onClick='vedioperemat($risultato);
return false;' role='link'>Guarda tutte le opere</a>";

}

}

else { //se non ha trovato record

echo "<p>Nessuna informazione disponibile</p>";

}

 ?>

<div id="contentopmat" aria-channel="main" aria-relevant="additions" aria-
atomic="false" aria-live="assertive" role="log">

</div>
```

3.11.3.2 Inserimento dei ruoli strutturali (Landmark) di ARIA

I ruoli strutturali vanno a definire le aree principali dell'applicazione. L'informazione semantica viene interpretata dalla tecnologia assistiva. Il lettore di schermo annuncia così all'utente tutti gli elementi strutturali presenti nella pagina.

Nel sito sono stati inseriti i seguenti ruoli strutturali:

- **banner:** ruolo che indica una regione che contiene l'header del sito, contenente il logo del museo e il titolo principale.
- **navigation:** ruolo per indicare la collezione di link che vanno a formare il menù principale di navigazione del sito museale.
- **main:** ruolo che indica il contenuto principale del sito; è consigliabile inserire un solo Landmark di questo tipo in ogni pagina dell'applicazione.
- **complementary:** ruolo che indica una regione del sito che rimane significati anche quando è separata dal contenuto principale. In generale esistono vari tipi di contenuto che possono utilizzare in modo appropriato questo Landmark. Nel caso del nostro sito è stato utilizzando per evidenziare la zona contenente le news sul museo, la zona del modulo per contattare il museo e la zona in cui viene visualizzata l'anteprima sulle opere dell'intero museo.

- **contentinfo:** ruolo per indicare una regione che contiene una serie di metadati relativi al sito. Nel conteninfo del sito museale sono contenute informazioni sull'ultimo aggiornamento del sito, sull'indirizzo civico ed elettronico del museo.

Figura 37: visualizzazione degli elementi landmark con juicystudio toolbar

3.11.3.3 Gestione delle regione attive con le proprietà ARIA

WAI-ARIA fornisce il ruolo log per identificare delle regione attive della pagina in modo da avvisare le tecnologie assistive dei cambiamenti che avvengono in queste regioni. In tutta

l’applicazione museale ogni script che utilizza Ajax inserisce i dati in regioni marcate con il ruolo log e caratterizzate da una serie di proprietà ARIA:

- **aria-atomic**. Proprietà che indica alla tecnologia assistiva se deve aggiornare all’utente tutta o solo una parte della regione attiva quando questa viene aggiornata. Il valore di default è false e indica che lo screen reader deve leggere solamente il nodo che è cambiato. Se il valore è impostato a true la tecnologia assistiva dovrebbe presentare tutti i contenuti dell’elemento.
- **aria-live.** Come già detto, questa proprietà indica che l’elemento sarà aggiornato e descrive il tipo di cambiamento che l’utente e la tecnologia assistiva si devono aspettare dalla regione attiva. Tipicamente il valore utilizzato è polite: gli utenti riceveranno la notifica degli aggiornamenti ma generalmente non verrà interrotto il task corrente, assegnando quindi a tali aggiornamenti una priorità più bassa. Le regioni che vengono individuate con il ruolo log hanno come valore di default proprio polite.

3.11.3.4 Inserimento di ruoli e proprietà ARIA

Oltre ai ruoli Landmark per definire meglio la struttura dell’applicazione e alle proprietà applicate alle regioni attive, nel sito sono stati inseriti una serie di ruoli WAI-ARIA sempre allo scopo di garantire una maggiore accessibilità dell’intera applicazione.

Ruoli inseriti nel sito sono:

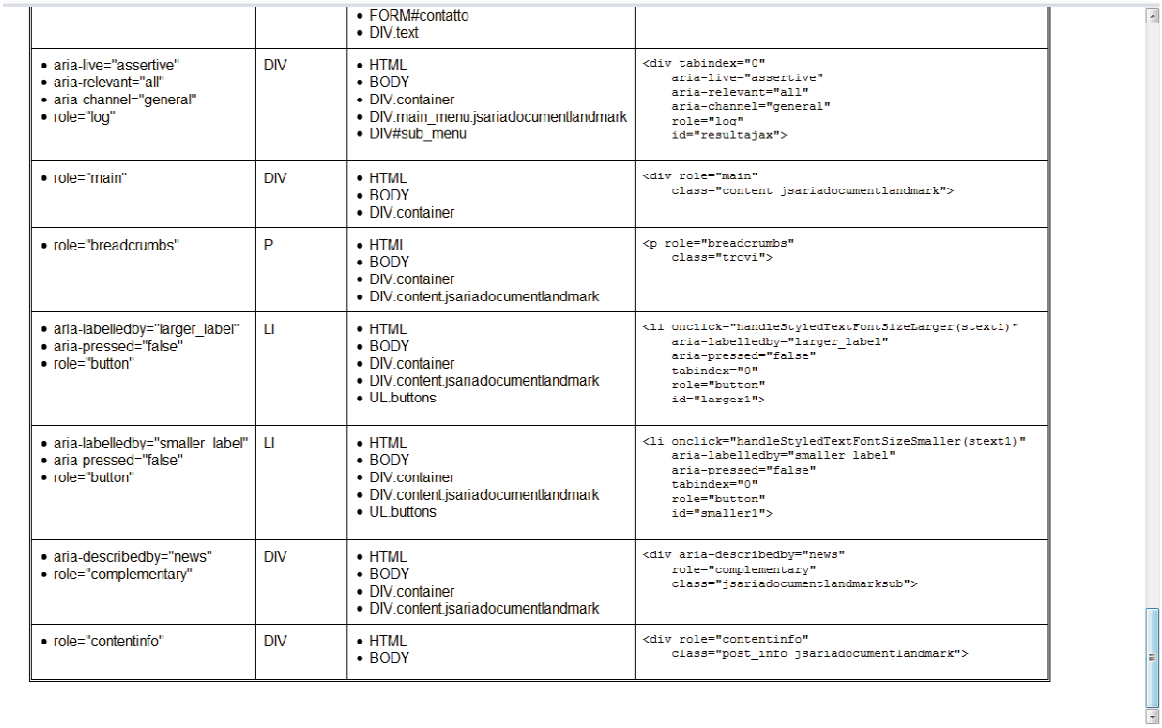
- **link**. Ruolo che indica un riferimento interattivo ad una risorsa interna o esterna all’applicazione. Se si tratta di un ancora html con un valore href, l’attivazione del link causerà la navigazione della risorsa da parte della tecnologia assistiva.
- **menu**. Ruolo che indica un tipo di widget che offre una lista di voci all’utente, spesso una lista di link ad importanti sezioni di un sito. Nel sito museale è stato utilizzato questo ruolo per indicare il menù laterale a sinistra, che permette di accedere alle sezioni del sito.
- **menubar**. Ruolo utilizzato per indicare tipicamente il menù orizzontali, utilizzato quindi nel sito museale per il menù orizzontale in alto.
- **menuitem**. Ruolo che indica una scelta in un gruppo contenuto da un menù o una menu bar.
- **button**. Con questo ruolo viene indicato un input che permette di attivare azioni user-triggered quando viene cliccato o premuto. Standardizzare l’aspetto dei bottoni migliora il

riconoscimento da parte degli utenti e permette una visualizzazione più compatta delle toolbar.

- **tooltip**. Ruolo che indica un pop-up contestuale che visualizza una descrizione dell'elemento con un mouse-over o con il focus da tastier. Gli elementi con questo ruolo dovrebbero essere descritti attraverso la proprietà aria-describedby nel momento in cui il tooltip è visualizzato.
- **textbox**. Ruolo che indica un input in cui si può inserire come valore del testo.
- **breadcrumbs**
- **option.** Ruolo che indica un elemento selezionabile in una lista di selezione. Le opzioni devono essere associate con un ruolo non astratto che deve essere combo box, listbox, menu, radiogroup o tree in modo da poter essere mappate correttamente dalle API di accessibilità.
- **list**. Ruolo che indica un gruppo non interattivo di elementi listitem o di gruppi di elementi che contengono figli di tipo listitem.
- **listitem**
- **combobox**. Con questo ruolo viene indicata una presentazione di selezioni. Una volta che viene effettuata una selezione è possibile attivare l'evento onChange da tastiera premendo il tasto invio: il focus rimane nella select e tramite i tasti freccia è possibile cambiare la selezione e riattivare il gestore di eventi.

  Proprietà inserite:

- **aria-required**. Proprietà che indica alla tecnologia assistiva che l'elemento di input va riempito prima che il modulo riceva il submit.
- **aria-describedby**. Proprietà che identifica l'elemento che descrive l'oggetto.



		• FORM#contatto • DIV.text	
• aria-live="assertive" • aria-relevant="all" • aria-channel="general" • role="log"	DIV	• HTML • BODY • DIV.container • DIV.main_menu.jsariadocumentlandmark • DIV#sub_menu	<div tabindex="0" aria-live="assertive" aria-relevant="all" aria-channel="general" role="log" id="resultajax">
• role="main"	DIV	• HTML • BODY • DIV.container	<div role="main" class="content jsariadocumentlandmark">
• role="breadcrumbs"	P	• HTML • BODY • DIV.container • DIV.content.jsariadocumentlandmark	<p role="breadcrumbs" class="trovi">
• aria-labelledby="larger_label" • aria-pressed="false" • role="button"	LI	• HTML • BODY • DIV.container • DIV.content.jsariadocumentlandmark • UL.buttons	<li onclick="handleStyledTextFontSizeLarger(stext1)" aria-labelledby="larger_label" aria-pressed="false" tabindex="0" role="button" id="larger1">
• aria-labelledby="smaller_label" • aria-pressed="false" • role="button"	LI	• HTML • BODY • DIV.container • DIV.content.jsariadocumentlandmark • UL.buttons	<li onclick="handleStyledTextFontSizeSmaller(stext1)" aria-labelledby="smaller_label" aria-pressed="false" tabindex="0" role="button" id="smaller1">
• aria-describedby="news" • role="complementary"	DIV	• HTML • BODY • DIV.container • DIV.content.jsariadocumentlandmark	<div aria-describedby="news" role="complementary" class="jsariadocumentlandmarksub">
• role="contentinfo"	DIV	• HTML • BODY	<div role="contentinfo" class="post_info jsariadocumentlandmark">

Figura 38: visualizzazione di ruoli e proprietà ARIA con juicystudio toolbar

3.11.4 Tecniche di implementazione usate

- XHTML
- CSS
- WAI-ARIA
- JAVA SCRIPT
- PHP
- SQL

3.11.5 Verifica dell'accessibilità

L'accessibilità del sito è stata valutata utilizzando il browser Mozilla (versione 3+) e il lettore di schermo Jaws 10.0, in quanto sono gli unici strumenti in grado di supportare pienamente gli attributi di WAI-ARIA.

Da segnalare comunque il tentativo di supportare ARIA anche da parte di Internet Explorer con la versione 8.0, anche se ancora rimangono delle difficoltà. E' comunque evidente il crescente

interesse verso il linguaggio di WAI-ARIA a testimonianza delle grandi opportunità che è in grado di fornire.

Per testare l’accessibilità sono stati utilizzati i vari plugin disponibili su Mozilla, con attenzione particolare alla toolbar di JuicyStudio che fornisce utili strumenti per l’individuazione delle regioni attive, dei ruoli strutturali e di tutte le proprietà e gli stati di ARIA.

Conclusioni

Era il 1991 quando Tim Berners Lee inventava il World Wide Web e da allora il web ha subito un lento processo di trasformazione passando da un'idea iniziale di web informativo ad un web partecipativo, collaborativo, sociale e, in questa fase attuale, applicativo.

L'utilizzo della tecnologia Ajax migliora l'interattività, la velocità e l'usabilità di un'applicazione web. Resta da chiarire come si può risolvere il problema dell'accessibilità.

Abbiamo visto il lavoro del W3C con WAI-ARIA e abbiamo visto come è realmente possibile utilizzare questo linguaggio per migliorare l'accessibilità di un'applicazione web, seppure limitatamente all'utilizzo di Mozilla e di Jaws 10.0. E' chiaro che ancora non rappresenta una soluzione definitiva per l'accessibilità di un'applicazione con Ajax ma ARIA va considerato un piccolo ma importante passo in avanti ed è sicuramente la strada giusta da seguire.

La necessità di avere applicazioni accessibili non impedisce di avere caratteristiche avanzate. Quello che serve è una progettazione che tenga conto dell'accessibilità fin dall'inizio e che applichi soluzioni caso per caso quando necessario, tenendo comunque presente che l'utilizzo di WAI-ARIA al momento risulta il modo migliore per garantire degli Ajax script accessibili.

Lo studio realizzato per questo lavoro di tesi dimostra che la suite WAI-ARIA presenta delle soluzioni per l'accessibilità molto interessanti ma che devono essere ulteriormente approfondite per capire la reale applicabilità nel web.

Nel sito del museo l'utilizzo di WAI-ARIA si è rivelato una ottima soluzione per garantire l'accessibilità di tutti gli script che utilizzano una tecnologia Ajax grazie all'utilizzo delle regioni attive e delle relative proprietà. Resta da capire fino a che punto l'utilizzo di tutti i ruoli, gli stati e le proprietà che sono stati definiti in WAI-ARIA può portare ad un miglioramento dell'accessibilità ma anche dell'usabilità di una pagina internet. Il fatto che il supporto di WAI-ARIA sta aumentando (anche Internet Explorer 8.0 comincia a supportarlo) è un segnale positivo e sta ad indicare che la strada tracciata dal WAI-ARIA è quella giusta da seguire.

Bibliografia e Sitografia

Web 2.0

[1] Mary Zaijcek. *Web 2.0: hype or happiness?* Department of Computing Oxford Brookes University, Wheatley Campus Oxford

[2] Il web 2.0 - Daniele Simonin - 2007
http://projects.melodycode.com/Web20/

[3] Cos'è il Web 2.0, Design patterns e modelli di business per la prossima generazione software - Tim O'Reilly
http://www.xyz.reply.it/web20/

[4] Wikipedia, Web 3.0
http://it.wikipedia.org/wiki/Web_3.0

Accessibilità

[5] Roberto Scano. *Accessibilità delle applicazioni web.* Pearson Education. 2008

[6] Usabilità ed Accessibilità: slide Progettazione di interfacce 2007-2008

[7] Guida completa accessibilità – Michele Diodati
http://accessibile.diodati.org/agc/

[8] Linee guida per l'accessibilità dei contenuti WCAG 2.0
http://webaccessibile.org/articoli/linee-guida-per-l-accessibilita-dei-contenuti-wcag-20-struttura-e-novita/

[9] Legge Stanca – Guida ai 22 requisiti tecnici
http://webaccessibile.org/articoli/legge-stanca-guida-ai-22-requisiti-tecnici/
http://www.pubbliaccesso.it/normative/DM080705-A.htm

[10] OpenWeb, Internet accessibile, *Applicazioni web 2.0 accessibili con WAI-ARIA*
http://blog.openweb.in/articoli/applicazioni-web-20-accessibili-con-wai-aria/

Ajax

[10] Bruce W Perry. *Ajax trucchi e segreti.* Milano, Tecniche nuove. 2006

[11] Dave Thau. *La grande guida Java Script: ora con Ajax!* Mondadori Informatica. 2008

[12] Adaptive Path, *A new approach to web application – Jasse James Garret*
http://www.adaptivepath.com/ideas/essays/archives/000385.php

Ajax e Screen Readers

[13] Sitepoint, *Ajax and Screenreaders: when can it work? – James Edward*

http://www.sitepoint.com/article/ajax-screenreaders-work

[14] Juicystudio, *Making Ajax work with screen readers*

http://juicystudio.com/article/making-ajax-work-with-screen-readers.php

[15] Juicystudio, *Improving Ajax application for JAWS users*

http://juicystudio.com/article/improving-ajax-applications-for-jaws-users.php

http://webaccessibile.org/articoli/ajax-e-jaws-i-rapporti-migliorano/)

[16] The Paciello Group Blog, *Ajax and Screen Readers – Content Access Issues*

http://www.paciellogroup.com/blog/?p=15

[17] Code Talks, *AT/JAWS*

http://wiki.codetalks.org/wiki/index.php/AT/JAWS

[18] Freedom Scientific, *What's New in Jaws 10*

http://www.freedomscientific.com/downloads/jaws/JAWS-whats-new.asp

Ajax accessibile

[4] Roberto Scano. *Accessibilità delle applicazioni web.* Pearson Education. 2008

[19] *Dal web 2.0 ai media sociali, Tracce e percorsi della partecipazione in rete.* CSP

http://estrablog.net/Dal%20web%202.0%20ai%20media%20sociali.pdf

[20] Guida completa accessibilità – Michele Diodati

http://accessibile.diodati.org/agc/

[21] Lau, *Realizzare pagine accessibili con Ajax – Filippo Pusset*

http://lau.csi.it/realizzare/accessibilita/linguaggi_programmazione/Ajax/pagine-accessibili.shtml

[22] Lau, *Progettare con Ajax: il Progressive Enhancement – Filippo Pusset*

http://lau.csi.it/progettare/accessibilita/strutturare_il_codice/progressive_enhancement.shtml

[23] W3C, *Roadmap for Accessible Rich Internet Applications (WAI-ARIA Roadmap)*

http://www.w3.org/TR/2006/WD-aria-roadmap-20060926/

[24] Joe Clark, *Build half a product: Is ajax accessible? At all? – Iceweb 2006*

http://joeclark.org/access/research/ice//iceweb2006-notes.html

[25] Juicystudio, *The AxsJAX framework for ARIA*

http://juicystudio.com/article/axsjax-framework-aria.php

[26] Juicystudio, *WAI-ARIA in HTML*

http://juicystudio.com/article/wai-aria-in-html.php

[27] Standards schmandards, *A pragmatic approach to web accessibility*

http://www.standards-schmandards.com/2007/rdfa-and-accessibility/

[28] Hijax: Progressive Enhancement with Ajax – Jeremy Keith

http://xtech06.usefulinc.com/schedule/paper/29

[29] W3C, *WAI-ARIA overview*

http://www.w3.org/WAI/intro/aria

[30] Unobtrusive Ajax – Jesse Skinner

http://www.thefutureoftheweb.com/talks/2006-10-ajax-experience/slides/

[31] W3C, *WAI ARIA Best Pratices: Providing Keyboard Focus*

http://www.w3.org/TR/2009/WD-wai-aria-practices-20090224/#kbd_focus

[32] W3C, *WAI ARIA Best Pratices: Managing Focus*

http://www.w3.org/TR/wai-aria/#managingfocus

[33] The Paciello Group, *WAI-ARIA it's easy – Steve Faulkner*

http://www.paciellogroup.com/blog/misc/ARIA/atmedia2008/

[34] iCITA, *ARIA Examples*

http://test.cita.uiuc.edu/aria/

[35] AxsJAX FAQ

http://google-axsjax.googlecode.com/svn/trunk/docs/faq.html#sec-2

[36] The Paciello Group Blog, *Using WAI ARIA Landmark Roles*

http://www.paciellogroup.com/blog/?p=106

[37] Dev.Opera, *Introduction to WAI ARIA – Gez Lemon*

http://dev.opera.com/articles/view/introduction-to-wai-aria/

[38] WAI-ARIA, *Role, State and Properties*

http://www.w3.org/TR/wai-aria/#Using_intro

[39] WAI-ARIA, *Best Pratices*

http://www.w3.org/TR/2009/WD-wai-aria-practices-20090224/