Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Linguaggi "XML-like" per il Web Semantico

Ing. Giovanni Canfora Sig.ra Daniela Di Fatta
Ing. Giovanni Pilato

RT-ICAR-PA-04-05 **aprile 2004**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Linguaggi "XML-like" per il Web Semantico

Ing. Giovanni Canfora[2] **Sig.ra Daniela Di Fatta**[1]
Ing. Giovanni Pilato[1]

Rapporto Tecnico N.6: ***Data:***
RT-ICAR-PA-04-05 **aprile 2004**

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sezione di Palermo Viale delle Scienze edificio 11 90128 Palermo
[2] Università degli Studi di Palermo Dipartimento di Ingegneria Informatica Viale delle Scienze 90128 Palermo

Sommario

Attualmente, Internet si presenta come un enorme contenitore di dati di vario genere, in cui le informazioni sono comunemente espresse senza fornire indicazioni sul loro significato. Ciò ne impedisce un'interpretazione automatica e rende imprescindibile l'apporto umano.

Il Web Semantico, mediante una serie di linguaggi formali, si prefigge di associare un significato alle informazioni presenti sul Web, in modo da consentire la realizzazione di applicazioni, che possano interpretare automaticamente i dati.

In tale contesto, un ruolo fondamentale è svolto dal linguaggio ***XML*** (*eXstensible Markup Language*), che può essere visto come un *meta-linguaggio*, composto da un insieme di regole sintattiche, per la creazione di nuovi linguaggi semantici per la marcatura delle informazioni contenute nei documenti.

L'*annotazione semantica* dei documenti, realizzata dagli autori mediante opportuni programmi o in modo automatico, attraverso la dichiarazione esplicita di *meta-informazioni* espresse in un linguaggio formale universale, consentirà la gestione automatica dei dati e delle informazioni presenti su Internet.

Il presente rapporto tecnico si propone di analizzare i diversi aspetti del Web Semantico e dei linguaggi necessari per la sua realizzazione. Particolare attenzione è stata posta allo studio dei linguaggi basati su XML fra cui si annovera anche il nuovo linguaggio per le applicazioni "audio" XML Voice.

INDICE

Capitolo 1

Il Web Semantico

> "Il Web Semantico è un estensione del Web attuale in cui le informazioni hanno un significato ben definito ed abilitano la co-operazione fra uomo e macchina."
>
> *Tim Berners-Lee, James Hendler, OraLassila,*
> *The Semantic Web, Scientific American, May 2001*

Il **Web Semantico** rappresenta un nuovo modo di concepire il Web ed i suoi contenuti, presentato per la prima volta da *Tim Berners-Lee*.
In tale visione del Web, si renderà accessibile alle macchine il contenuto *concettuale* delle informazioni presenti sul Web: il Web Semantico rappresenterà "un'estensione del Web attuale in cui le informazioni saranno strutturate con un senso compiuto, migliorando il lavoro tra le persone e gli elaboratori" [3].
Il risultato sarà un nuovo modo di intendere il Web e più in generale il rapporto fra l'uomo e l'elaboratore, poiché gli elaboratori interagiranno fra di loro, con gli esseri umani e con altre risorse Web, tramite l'utilizzo di programmi per elaboratori distribuiti.
Documenti comprensibili alle macchine permetteranno lo sviluppo degli "agenti software", programmi che si muovono sul Web, interagendo con altri "agenti", in modo da eseguire compiti complessi per i loro utenti, in completa autonomia senza richiedere l'intervento umano [4][5].
L'autonomia degli "agenti software", nell'eseguire compiti complessi, sarà da ascrivere sia alle capacità di ragionamento degli "agenti" stessi, ma anche alla capacità di comprendere il reale contenuto dei documenti Web.
Nello svolgere i compiti assegnati, gli "agenti" opereranno tenendo conto delle preferenze dell'utente, quindi, ad esempio un agente software potrà essere utilizzato per prenotare una visita medica e per fare ciò interagirà con gli "agenti" personali dei medici presenti sul Web.

La ricerca del medico sarà effettuata tenendo conto delle preferenze dell'utente, del costo della visita, del calendario degli impegni dell'utente e del medico, delle percentuali di successo del medico e considerando eventuali suggerimenti di altri "agenti" personali presenti sul Web, come quelli di persone fidate (come i familiari o di "agenti" legati a siti che svolgono la funzione di esperti di settore).

Con l'avvento del Web Semantico, quindi, si verrà a creare un mondo virtuale in cui gli "agenti software" saranno capaci di collegare automaticamente fra loro le informazioni di differenti sorgenti, elaborare e scambiare dati con altri "agenti", usare i servizi messi a disposizione sul Web, in modo da velocizzare l'esecuzione dei compiti, limitando sempre più la necessità di interazione degli utenti.

Per realizzare ciò, gli "*agenti software*" necessitano che i contenuti presenti sul Web siano leggibili dalle macchine e devono essere in grado di apprendere conoscenza dagli utenti.[1]

Ma attualmente, la maggiore difficoltà è costituita dal fatto che i contenuti presenti sul Web sono progettati per l'uomo e per la sua capacità interpretativa, con una maggiore attenzione per la rappresentazione dei contenuti piuttosto che sul loro significato. Non è possibile, quindi, interpretare automaticamente.

È opportuno invece avere un'"*annotazione semantica*" dei documenti (realizzata dagli autori mediante opportuni programmi o in modo automatico), attraverso la dichiarazione esplicita di "*meta-informazioni*" espresse in un linguaggio formale che dovrà essere:

- universale (in grado di rappresentare la conoscenza);
- conciso (per essere ampliamente adottato deve essere semplice);
- non ambiguo (per consentire il ragionamento automatico);
- espandibile (cioè flessibile ed aperto alle personalizzazioni ed alle evoluzioni future);
- globale (adottato da tutti coloro che pubblicano sul Web);

[1] L'apprendimento di conoscenza è fondamentale per permettere la condivisione della conoscenza fra utente e agente in modo che l'esecuzione di specifici compiti possa essere effettuato dall'agente avendo a disposizione la stessa conoscenza. In questo modo si può sperare di sostituire gli utenti con gli "agenti software" nell'esecuzione di compiti specifici.
Il fatto che l'agente abbia appreso dall'utente come comportarsi costituisce una base su cui si può fondare la fiducia che deve avere l'utente nell'agente, in modo da potere delegare l'agente a svolgere determinati compiti.

- comprensibile alle macchine e trattabile con un basso costo computazionale.

Le "*Meta-informazioni*", dette anche " *meta-dati*", sono "informazioni che descrivono altre informazioni" e essendo comprensibili alle macchine, permettono di descrivere le risorse.

La distinzione fra "dati" e "metadati" non è di tipo assoluto, infatti, la stessa risorsa potrà essere interpretata simultaneamente nei due modi.

Le "*meta-informazioni*" permetteranno agli autori di specificare informazioni sui loro documenti così da renderli non soltanto leggibili, ma anche interpretabili in maniera intelligente da diverse applicazioni.

Occorre sottolineare che spesso una parte rilevante dell'informazione è espressa in forma non testuale, grafica e multimediale, rendendo così ancora più complessa l'operazione di estrazione automatica del significato. Ma il Web Semantico dovrà permettere anche l'acquisizione di questa forma di conoscenza.

I principali obiettivi del Web Semantico:

- migliorare l'efficienza e la precisione dei motori di ricerca;
- realizzare sistemi di catalogazione dei contenuti;
- favorire la condivisione e lo scambio di informazioni fra *"agenti software" intelligenti;*
- aumentare l'accessibilità dell'informazione e l'integrazione di informazioni provenienti da sorgenti diverse;
- riunire in un unico documento logico collezioni di pagine Web correlate in modo semantico, ma distribuite su più siti;
- semplificare l'automazione di transazioni di tipo commerciale, aumentandone la sicurezza;
- permettere a ciascun utente di esprimere le preferenze sul trattamento elettronico dei propri dati personali e ai siti Web di comunicare le politiche di riservatezza che essi adottano[2];

[2] il progetto **P3P** (*Platform for Privacy Preferences Project*), promosso dal ***W3C***, ha lo scopo di consentire ai siti Web di descrivere la propria politica di privacy in un formato standard che possa essere automaticamente recuperato ed interpretato dagli "agenti software" degli utenti ; gli utenti potranno istruire gli agenti sulle proprie decisioni in materia di privacy

- aumentare il livello di fiducia degli utenti sulla qualità dei servizi e delle risposte del Web Semantico, grazie all'uso estensivo della firma digitale.

1.1 Principi Progettuali

I principi progettuali generali di riferimento da considerare nella definizione dell'architettura del Web Semantico sono [6]:

- ***Semplicità*:** una maggiore semplicità facilita l'utilizzo delle applicazioni rendendolo più agevole l'utilizzo.
- ***Progettazione Modulare*:** nella progettazione di un sistema si deve cercare di individuare le caratteristiche delle diverse parti da realizzare, in modo da raggrupparle in "*moduli*" omogenei internamente, ma debolmente correlati con altri "*moduli*". La progettazione modulare consente, quindi, di realizzare la suddivisione il sistema in moduli separati, con conseguenti vantaggi sia in fase di costruzione sia di modifica[3].
- ***Decentralizzazione*:** tale principio indica che un sistema centralizzato, con singole parti che intervengono in tutte le operazioni, ha una scalabilità limitata, con conseguente maggiore rischio di fallimento.
- ***Test d'Invenzione Indipendente*:** rappresenta il principio di "*modularità totale*", in quanto consente, nella fase di progettazione, di pensare il sistema come una parte di un sistema più ampio, non ancora specificato.
- **Principio del "Minimo Potere":** bisogna cercare di realizzare soluzioni meno potenti, perché minore è la potenza del linguaggio, maggiore sono la trattabilità e l'usabilità dei dati.

Ma, oltre questi principi generali di buona progettazione, vi sono altri principi più specifici del contesto del Web Semantico[7]:

[3] In fase di costruzione o durante la vita del sistema, ogni modifica che riguarda un dato modulo avrà impatti limitati sugli altri moduli, essendo indipendenti fra di loro e collegati tramite l'interfaccia. Sarà, quindi, sufficiente modificare e testare il singolo modulo, con vantaggi per il fatto che sui singoli moduli possono lavorare contemporaneamente e indipendentemente gruppi di lavoro differenti.

- **Informazione parziale:** nel Web Semantico ciascuno potrà dire qualunque cosa a proposito di qualunque argomento, essendovi ampia liberta di espressione di contenuti.
- **Web della Fiducia:** contrariamente a quanto accade nel Web attuale in cui non c'è alcun modo di verificare l'attendibilità e la correttezza delle informazioni, nel Web Semantico le affermazioni presenti sono ritenute non "vere" e le applicazioni necessitano di ricevere informazioni esplicite sull'attendibilità.
- **Evoluzione**: il Web Semantico deve permettere che il lavoro svolto in maniera autonoma da comunità diverse possa essere combinato efficacemente, fornendo loro la capacità di risolvere le ambiguità e di chiarire le inconsistenze.
- **Progettazione Minimalista:** si deve cercare di rendere semplici e possibili i progetti complessi.

1.2 Architettura

Nella visione di Berners-Lee, il Web Semantico potrà essere realizzato attraverso una struttura a più livelli, che sarà sviluppata completamente nel giro di pochi anni. In tale struttura, ogni livello avrà il compito di estendere e completare i servizi offerti dal livello subito inferiore, presentando al livello superiore nuove funzionalità attraverso appositi linguaggi [8].

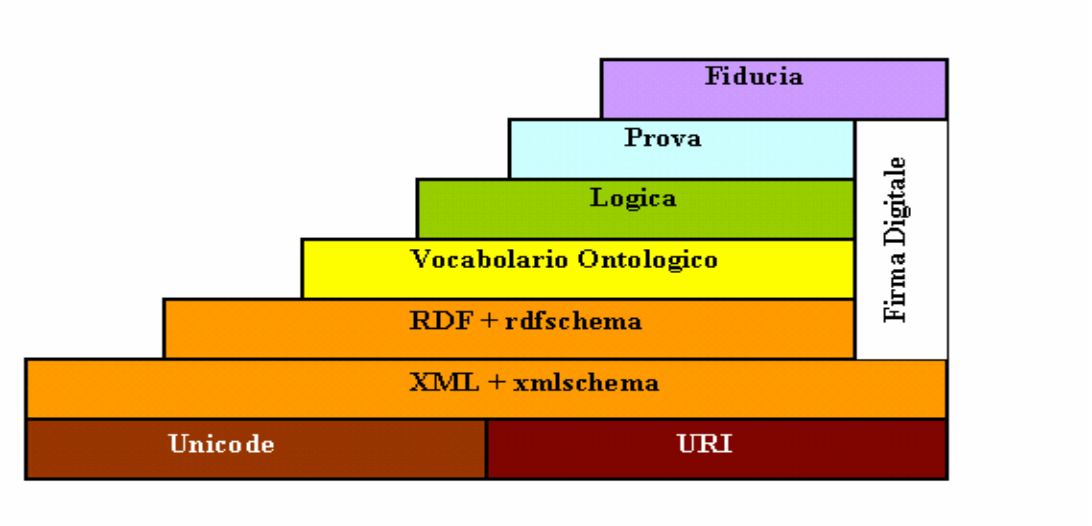

Figura 1.1: Architettura del Web Semantico:struttura a livelli

Per comprendere la funzione di ogni linguaggio, nel progetto di definizione del Web Semantico, bisogna analizzare ogni singolo livello iniziando dagli strati inferiori che realizzano l'infrastruttura su cui si fonda tutto il Web Semantico.

1.2.1 Unicode & URI

Figura 1.2: Livelli dell'architettura del Web Semantico fino a Unicode & URI

Come detto precedentemente, al primo livello dell'architettura vi sono gli elementi che permettono la definizione di una sintassi comune: ***Unicode*** e ***URI***.

- **Unicode** è un "*codice universale*" simile al codice ***ASCII***, ma più completo, che permette di codificare i caratteri dei maggiori linguaggi oggi in uso e di molti dei linguaggi precedentemente esistenti.
  La necessità di codificare tutti i caratteri esistenti è dovuta alla "*globalizzazione*" di Internet che ha portato ad un Web in cui sono presenti documenti scritti in varie lingue.
- ***URI*** nasce dall'esigenza di realizzare un meccanismo per identificare e individuare qualunque cosa esiste sul Web. Se si identificano con le "*risorse*" qualunque cosa che può essere identificata non necessariamente sul Web, scopo degli ***URI*** è di realizzare un meccanismo per identificare e individuare le "*risorse*".
  L'individuazione delle risorse avviene tramite un sistema uniforme di identificatori chiamato **URI** ("*Uniform Resourse Identifiers*"), che rappresentano il nome assegnato ad una risorsa, mediante cui è possibile identificare e localizzare le risorse.
  Una forma particolare di ***URI*** è costituita dagli ***URL*** ("*Uniform Resourse Locator*"), che rappresentano "*l'indirizzo univoco*" utilizzabile per localizzare un documento sul Web.
  Esistono diversi schemi di ***URI***, alcuni completamente decentralizzati (come quello dei "*freenet*"), in cui si possono avere oggetti diversi con lo stesso nome, altri invece controllati in modo centralizzato (come lo schema ***URI*** http:), in cui il controllo è

effettuato per mezzo del **DNS** (**D**omain **N**ame **S**erver) che impedisce che lo stesso nome si riferisca ad oggetti diversi.

Nulla, però, impedisce ch'e ***URI*** diversi possano riferirsi allo stesso oggetto, non consentendo la cosiddetta "*Assunzione di Nome Unico*" o **UNA** ("**U**nique **N**ame **A**ssumption") in cui si presume che ogni oggetto del dominio sia nominato attraverso un solo termine [9].

La sintassi degli ***URI*** é progettata per essere:

a) **Estensibile**: si possono aggiungere nuovi schemi, per mantenere l'accessibilità delle risorse anche in presenza di nuovi protocolli e identificatori;
b) **Completa:** tutti i nomi esistenti sono codificabili e nuovi protocolli sono comunque esprimibili tramite ***URI***;
c) **Stampabile:** é possibile esprimere gli ***URI*** con caratteri ***ASCII*** a "*7-bit*", permettendo gli scambi lungo un qualunque canale fisico.

Gli ***URI*** sono rappresentati mediante una stringa arbitraria registrata, usata come prefisso e seguita da una parte specifica la cui decodifica é funzione del prefisso stesso:

uri = schema:parte-specifica#frammento

Il carattere "#" delimita l'identificatore di un frammento interno alla risorsa considerata, permettendo riferimenti a parti interne alla risorsa.

1.2.2 XML & XML Schema

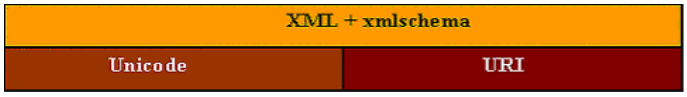

Figura 1.3: Livelli dell'architettura del Web Semantico fino a XML e XMLSchema

Il livello successivo è costituito da ***XML*** (*eXstensible Markup Language*), un "*meta-linguaggio*" composto da un insieme di regole di sintassi per la creazione di nuovi linguaggi semantici di marcatura delle informazioni contenute nei documenti.

La necessità di creare tale linguaggio è da ricercare nell'esigenza di dare ad ogni documento Web un modo per identificare gli elementi di informazione che lo

compongono. L'identificazione può essere fatta associando ad ogni elemento di informazione delle opportune sequenze di caratteri (stringhe), dette "*meta-etichette*" o anche "*markup*". Il compito di tali "*meta-etichette*" è di indicare il ruolo svolto dall'elemento di informazione nel documento stesso, secondo le scelte dell'autore.

La presenza delle "*meta-etichette*" rende gli elementi di informazione "*auto-descriventi*" e qualunque applicazione o utente, che legga un documento di questo tipo, riesce facilmente ad individuare il ruolo di ogni elemento.

In particolare, ***XML*** rappresenta un meccanismo che permette di definire dei formati dati per documenti strutturati, in modo da realizzare dei linguaggi di "*markup*" standard. Quindi, tramite ***XML*** , chiunque può progettare un proprio formato di documento e scrivere i documenti in tale formato.

La caratteristica fondamentale di ***XML*** è che è un lignaggio "*estensibile*". Il fatto che non vi sia un numero prefissato di "*meta-etichette*", permette ad ogni utente di realizzare le "*meta-etichette*" che desidera.

Importante risulta, inoltre, il ruolo dei ***namespace***, che permettono di identificare degli elementi in modo univoco all'interno del Web, tramite l'utilizzo degli ***URI***.

I ***namespace*** permettono, quindi, di miscelare differenti linguaggi in un oggetto ***XML***, in quanto ogni linguaggio sarà caratterizzato da uno specifico ***namespace***. In questo modo, si possono utilizzare diversi linguaggi in un documento, senza rischio di errata interpretazione dei singoli elementi.

Allo stesso livello di ***XML***, è presente ***XML Schema***, un linguaggio che realizza un meccanismo per definire la grammatica di documenti legali ***XML***, permettendo la descrizione del linguaggio sviluppato: vengono indicati gli elementi e i vincoli che devono essere applicati.

1.2.3 RDF & RDF Schema

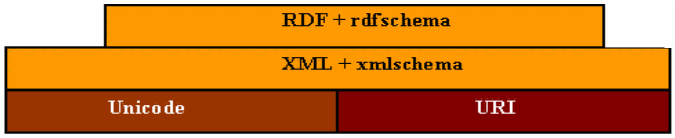

Figura 1.4: Livelli dell'architettura del Web Semantico fino a RDF e RDFS

Lo strato successivo a ***XML*** e ***XML Schema*** è presente lo strato relativo a ***RDF*** (*Resourse Description Framework*) [10] che permette di esprimere informazioni circa i dati.

Tale strato definisce un modello dati per descrivere dati trattabili dagli elaboratori, utilizzando come primitiva base per la modellazione la tripla "*oggetto-proprietà-valore*". Mediante tale "*tripla*", viene definito un modello semplice per descrivere relazioni fra le risorse, in termini di proprietà identificate da un nome e da relativi valori. Di conseguenza un documento risulta costituito da un insieme di "*triple*", dette anche dichiarazioni ***RDF***, dove gli oggetti sono le "*risorse*" precedentemente definite. Tuttavia, ***RDF*** non definisce la semantica di uno specifico dominio, ma risulta un meccanismo indipendente dal dominio. Quindi, è necessario un meccanismo che permette di definire delle proprietà specifiche del dominio e le relazioni con altre risorse.

Per tali motivi è stato realizzato il linguaggio ***RDF Schema***, che definisce dei "*vocabolari*" contenenti l'insieme delle proprietà semantiche individuate e condivise da una data comunità, in modo da permettere di definire il significato, le caratteristiche, le relazioni e i vincoli di un insieme di proprietà in un dato dominio.

RDF Schema costituisce un semplice linguaggio di modellazione su ***RDF*** che include come primitive di modellazione: classi, relazioni fra classi e fra proprietà, restrizioni di dominio e intervalli per proprietà. Mediante ***RDF Schema***, quindi, è possibile la definizione di gerarchie di classi, utili per rappresentare la conoscenza di un dato dominio.

Ma, le definizioni di ***RDF Schema*** non sono sufficienti per esprimere la conoscenza di domini complessi, il che rende necessario un vocabolario ontologico più esteso.

1.2.4 Ontologie

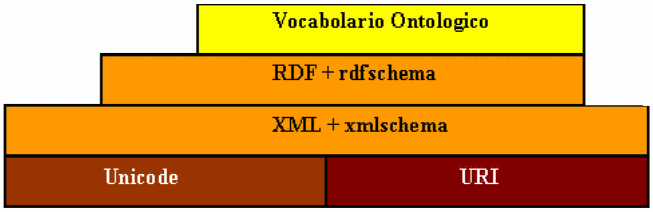

Figura 1.5: Livelli dell'architettura del Web Semantico fino al Vocabolario Ontologico

Il termine ontologia indica un documento condiviso contenente la descrizione formale dei concetti di un dato dominio; identifica le classi più importanti, le organizza in una gerarchia, specifica le loro proprietà (che caratterizzano anche gli oggetti appartenenti alla classe) e descrive anche le relazioni più significative che legano le classi.
Lo strato ontologico però non si occupa di come utilizzare le relazioni dal punto di vista del calcolo automatico, compito riservato, piuttosto, allo strato immediatamente successivo.

1.2.5 Logica

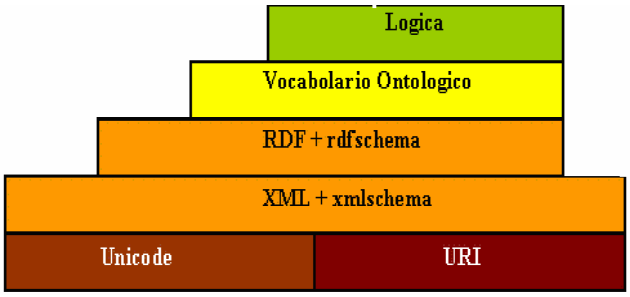

Figura 1.6: Livelli dell'architettura del Web Semantico fino alla Logica

Il livello immediatamente superiore a quello ontologico è il ***livello logico***. Spesso i due livelli, logico e ontologico, sono realizzati insieme. Tale livello e tutti quelli successivi sono ancora in una fase preliminare, e , al momento, non è risulta definito come dovranno essere realizzati.

Con i livelli fino all' ontologico è possibile riconoscere i concetti basilari, ma non è possibile effettuare ragionamenti perché non vi è inferenza[4], ma solo rappresentazione della conoscenza.

Nel livello logico le asserzioni presenti sul Web sono utilizzate per derivare (inferire) nuova conoscenza mediante la deduzione logica[5].

Affinché il Web Semantico possa divenire sufficientemente espressivo, deve essere possibile per le applicazioni estrarre autonomamente informazioni utili dalla vasta mole di documenti Web "*annotati*", utilizzando un apposito linguaggio logico per realizzare le inferenze.

Dato che per effettuare delle deduzioni potrebbe essere necessario reperire un numero elevato di regole logiche sul Web, i motori di inferenza dovranno necessariamente essere fondati su euristiche.

1.2.6 Prova

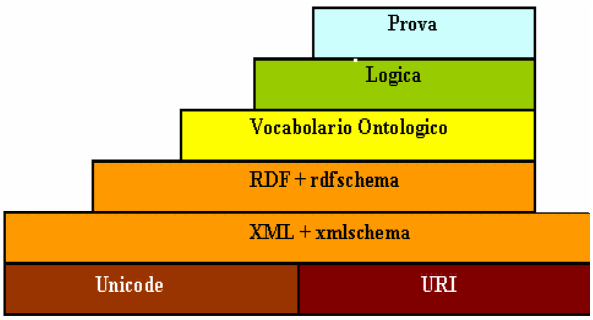

Figura 1.7: Livelli dell'architettura del Web Semantico fino alla Prova

La capacità del web semantico di verificare espressioni è molto importante, perché permette di individuare le informazioni corrette dal punto di vista dell'applicazione. Per realizzare questo livello, gli autori delle dichiarazioni dovrebbero essere capaci di fornire una prova verificabile agli elaboratori, senza bisogno che la macchina trovi autonomamente la prova.

I sistemi deduttivi non sono normalmente interoperabili, per cui, invece di progettare un sistema per supportare il ragionamento unico e onnicomprensivo, si potrebbe definire un "*linguaggio universale per rappresentare le dimostrazioni*".

[4] L'inferenza è un procedimento deduttivo mediante il quale, da una o più premesse, si ricava per via logica, una conclusione.

[5] Il tipo di inferenza dipende dalla logica scelta

Una dimostrazione rappresenta una sequenza di formule, derivate attraverso regole di inferenza, da assiomi, definizioni o formule precedenti nella sequenza.
Un "*linguaggio di prova*" serve, quindi, per comunicare le dimostrazioni e consentire ad "agenti software" di scambiarsi delle asserzioni, unitamente alla catena di inferenza attraverso cui sono state ottenute per deduzione da altre asserzioni ritenute accettabili da parte dell'agente ricevente.
Inoltre, le applicazioni potrebbero autenticare con la ***firma digitale*** le dimostrazioni ed esportarle ad altre applicazioni.

1.2.7 Firma digitale

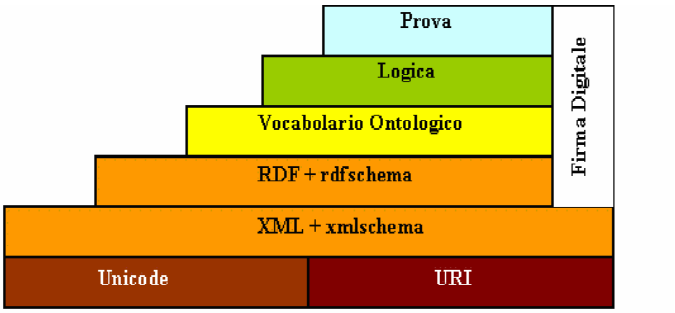

Figura 1.8: Livelli dell'architettura del Web Semantico compresa la Firma digitale

La ***"Firma Digitale"*** è importante a diversi livelli del modello astratto del Web Semantico, poiché può essere utilizzata per stabilire la provenienza dei dati nelle ontologie e nelle deduzioni e, permette di avere "agenti software" che non si "*fidano*" delle affermazioni che reperiscono sul Web finché non verificano l'attendibilità della fonte.
La firma digitale, sfruttando tecniche crittografiche, garantisce l'autenticità delle varie asserzioni e permette di scoprire la loro provenienza.
La "*crittografia a chiave pubblica*" è una tecnica nota da qualche anno, ma non ancora diffusa su larga scala, forse perché impone una scelta netta tra fiducia e non fiducia, mentre sarebbe necessaria una infrastruttura in cui le parti possano essere riconosciute e accettate come credibili solo in specifici domini.

1.2.8 Fiducia

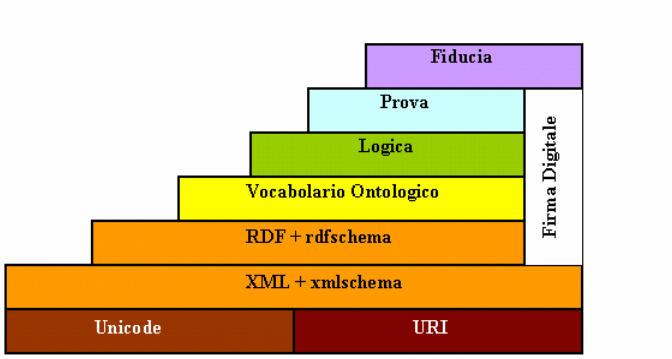

Figura 1.9: Livelli dell'architettura del Web Semantico fino alla Fiducia

L'introduzione della ***Firma Digitale*** permetterà di realizzare il "***Web della Fiducia***", in cui ogni "*agente automatico*" che trova un documento Web, potrà riconoscere se il documento viene da una fonte fidata oppure una fonte di cui sicuramente non si può fidare.

Attualmente, la libertà di espressione totale del Web Semantico comporta problemi legati al fatto che non c'è nessun modo per stabilire quando fidarsi dei documenti presenti sul Web.

Con la ***firma digitale***, le asserzioni saranno contrassegnate con la firma relativa alla persona o all'organizzazione che le ha prodotte: la firma digitale, sfruttando tecniche crittografiche, garantisce l'autenticità delle varie asserzioni e permette di scoprire la loro provenienza. Compito dell'utente sarà istruire l'applicazione a riconoscere quali sono le firme fidate e quali quelle di cui non fidarsi.

Ma l'introduzione della *"fiducia"* può avere effetti indesiderati, perché si rischia di non considerare un' ampia parte della conoscenza presente sul Web, laddove asserita da enti non direttamente conosciuti dall'utente.

Per superare queste difficoltà nasce l'idea del "***Web of Trust***", che considera la fiducia come una proprietà transitiva, sicché se l'utente A dichiara di fidarsi dell'utente B, il quale a sua volta dichiara di fidarsi dell'utente C, si ottiene che A si deve fidare anche di C, creando una "r*ete di fiducia*" che strutturandosi a grafo, lega assieme i vari utenti.

Per modulare meglio la *"fiducia"*, si potrebbe pensare di legare il "*livello di fiducia*" ad un parametro, come la distanza topologica nel grafo della *"fiducia"*, in modo che i nodi

vicini, dal punto di vista topologico, avranno fra loro un grado di *"fiducia"* maggiore rispetto a quelli posti più distanti.

Parallelamente, si potrebbe creare una "*rete delle sfiducia*" che consenta di avere indicazioni esplicite sull'inaffidabilità di una fonte informativa, consentendo ad un "*agente software*" di gestire i documenti per cui non ha prova esplicita né di *"fiducia"* né di "*sfiducia*".

L'"*agente*" potrebbe considerare il documento in merito al quale non ha informazioni, ragionevolmente più affidabile rispetto ad un documento per il quale esista invece una prova esplicita di "*sfiducia*".

Capitolo 2

Linguaggi basati su XML

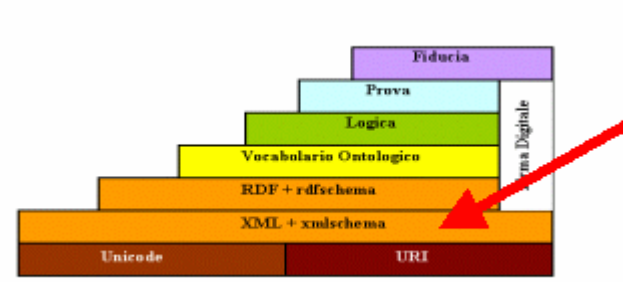

Figura 2.1: Posizione di XML nell'architettura del Web Semantico

2.1 XML

Il linguaggio ***HTML*** (*Hypertext Markup Language*) grazie alla sua semplicità ha contribuito notevolmente all'evoluzione del Web. Tuttavia , oggi le applicazioni richiedono un supporto tecnologico sempre più potente e flessibile rispetto a quello fornito da tale linguaggio.

Nato per visualizzare dati sul Web, ***HTML*** fornisce delle regole per stabilire come deve essere scritto un documento per essere compreso ed elaborato da tutti i browser presenti in rete, tralasciando la struttura ed il significato del contenuto.

Le etichette utilizzabili sono solo quelle prefissate, e non sono estendibili rendendo , quindi il linguaggio inidoneo a rappresentare nuovi tipi di documenti.

Per dare al Web una maggiore strutturazione, possibilmente semantica, è stato introdotto l'***XML*** (*eXtensible Markup Language*)[6], un linguaggio di "*marcatura*" sviluppato dal ***W3C***, che mette insieme la semplicità di ***HTML,*** la flessibilità e la possibilità di definire nuovi tipi di documenti di ***SGML***. Il linguaggio ottenuto risulta semplice e dotato della flessibilità richiesta dagli sviluppatori per risolvere problematiche complesse.

XML è il primo linguaggio che rende i documenti comprensibili all'uomo e agli elaboratori, attraverso etichette più potenti e flessibili di quelli del linguaggio ***HTML***.

[6] La prima versione di ***XML*** è stata approvata il 10 Febbraio del 1998

Al contrario di ***HTML*** che è un linguaggio per un tipo specifico di documenti ipertestuali[11], ***XML*** nasce per strutturare documenti arbitrari, permettendo di definire nuovi tipi e di realizzare veri e propri linguaggi per la formattazione di documenti.

2.1.1 Struttura

Un documento ***XML*** è composto di due parti:

1. **Prologo**: è una parte opzionale che contiene le dichiarazioni fra cui quella di tipo;
2. **istanza del documento**: rappresenta il documento vero e proprio.

La parte relativa alle dichiarazioni, posta all'inizio del documento, serve a:

- dichiarare che è un documento ***XML*** di una specifica versione;
- dichiarare l'insieme di caratteri usati nel documento;
- indicare se le dichiarazioni sono contenute solo internamente o se vi è qualche parte esterna al documento stesso.

La "*dichiarazione di tipo*" specifica quali strutture sono ammesse per il documento, generalmente mediante una definizione del tipo di dati, detta ***DTD*** (*Data Type Definition*).
Le ***DTD*** contengono informazioni sulla struttura delle "etichette" da utilizzare nel documento ***XML***.
Il "*documento di specificazione dei tipi di dati*" è importante perché permette di "*verificare*" l'istanza del documento, per individuare violazioni rispetto la struttura definita delle etichette
Dopo la dichiarazione di tipo, c'è "l'i*stanza del documento*" che costituisce il documento vero e proprio.
Un documento ***XML*** consiste di un insieme, correttamente annidato, di etichette dette "*elementi"* che possono avere un numero arbitrario di coppie "*attributo-valore*".
Le etichette, utilizzate nel documento, apparterranno alla struttura specificata nella ***DTD,*** ma il "*vocabolario delle etichette*" e la loro combinazione non sono uguali per tutti i documenti ***XML*** essendo possibile definirli localmente per l'applicazione.

XML ha regole esplicite di semantica e sintassi che stabiliscono il modo di scrivere un documento e che permettono di distinguere i documenti ***XML*** in documenti "*ben formati*" e documenti "*validi*".

Un documento si dice *"**ben formato**"* se rispetta le regole di carattere generale, semantiche e sintattiche, specificate da ***XML***, quindi, un documento per essere scritto in ***XML*** deve essere *"**ben formato***:

- avere una prima riga del tipo:
  <?XML version="1.0"?>
- essere un elemento a singola radice;
- rispettare la sintassi delle etichette;
- avere argomenti posti fra etichette di tipo virgolette;
- dare un nome alle etichette e agli attributi;
- rispettare l'annidamento delle etichette.

Invece, un documento ***XML*** si dice "***valido***" se è conforme al tipo specificato da una data ***DTD***: un documento ***XML*** può essere "*valido*" per una ***DTD*** e non "valido" per un'altra, comunque deve essere "*ben formato*" per essere un documento ***XML***.

Un applicativo per leggere documenti ***XML*** deve avere due parti distinte:

1. il **Parser** che esegue il controllo semantico e gestisce gli errori;
2. il **Processore** che visualizza il documento, utilizzando un altro documento in cui è definita la formattazione delle varie etichette.

La separazione fra struttura dei dati e rappresentazione è uno degli aspetti chiave per la buona costruzione di un ipertesto.

In ***XML***, la separazione è garantita dalla divisione fisica dei dati che governano i due aspetti e dei linguaggi (la rappresentazione è gestita da ***XSL***).

La separazione fra struttura e rappresentazione facilita l'elaborazione del documento ***XML*** da parte delle applicazioni, perché il documento descrive solo i dati e sono assenti le istruzioni di formattazione.

XML sta diventando il "*linguaggio universale*" per rappresentare i dati, permettendo a tutte le applicazioni di comunicare, purché ogni applicazione sia in grado di capire il *marcatura* o *vocabolario* di un'altra applicazione.

XML può essere utilizzato per scopi piuttosto differenti:

- come sintassi di serializzazione per altri linguaggi specificati solo a livello semantico (ad esempio ***RDF***, descritto di seguito);
- come marcatura semantico per le pagine Web;
- come formato uniforme per lo scambio dei dati.

2.1.2 Vantaggi

Il linguaggio ***HTML*** descrive il modo di visualizzazione dei dati, invece ***XML*** rappresenta un modo standard per descrivere e scambiare dati strutturati attraverso il Web, permettendo dichiarazioni del contenuto più precise.
Grazie alla sua flessibilità, ***XML*** permette di specificare il significato del contenuto di un documento attraverso l'utilizzo delle etichette ritenute più opportune dall'utente, rendendo più chiaro e ricco di informazioni il documento.
I benefici derivanti dall'utilizzo di ***XML*** nelle applicazioni per il Web sono molteplici:

- ricerche più efficaci;
- sviluppo di applicazioni più flessibili;
- integrazione di dati provenienti da differenti sorgenti;
- elaborazione e manipolazione locale dei dati;
- diverse possibilità di visualizzazione della stessa struttura dati.

2.1.3 Spazio dei nomi

Un problema riscontrabile nei documenti ***XML*** è la sovrapposizione dei nomi degli elementi e degli attributi appartenenti a diversi "*vocabolari di marcatura*". Tale problema sorge quando un processore ***XML*** elabora elementi e attributi definiti e usati da differenti applicativi.
La soluzione potrebbe essere la creazione di etichette e attributi con nomi preceduti da un "***prefisso***" la cui visibilità supera i limiti del documento che li contiene, ottenendo così dei "*nomi universali*".
La creazione di nomi "*universalmente riconosciuti*" eliminerebbe le ambiguità, permettendo di trattare le etichette in maniera diversa anche se hanno lo stesso nome.

La struttura utilizzata per ottenete ciò è quella dei ***namespace XML***[12], costituiti da un "*prefisso*" e da una parte locale del tipo:

prefisso:nome

Il "*prefisso*" è un' abbreviazione per il ***namespace***, costituito da un ***URI*** [7], mentre la parte locale è l'identificatore per i "metadati".
Utilizzando gli ***URI***, che sono oggetti univoci, si evitano sovrapposizioni fra i prefissi.
Un esempio di dichiarazione di ***namespace*** è:

<xsd:schema XMLns:xsd="http://www.w3.org/2001/XMLSchema">

Tale esempio costituisce la dichiarazione di un namespace per tutti gli elementi ***XML Schema***, per essere usato in un documento schema; esso definisce il prefisso '*xsd* ' per identificare il ***namespace*** "http://www.w3.org/2001/XMLSchema"
Esistono due modi per applicare i ***namespace*** ad un documento:
a) mettere il prefisso in ogni elemento e attributo del documento,;
dichiarare un ***namespace*** di riferimento per il documento, eliminando il prefisso dalla dichiarazione.

2.2 Componenti di XML

2.2.1 Schemi sintattici

Le etichette ***XML*** realizzano i "*metadati*" che sono informazioni circa i dati che un documento contiene.
I destinatari dei documenti, per trattare opportunamente il documento, devono accedere al significato delle etichette. Di conseguenza, oltre che allo stile, bisogna prestare attenzione alla sintassi e alla struttura del file ***XML***. A tale scopo, è possibile imporre delle restrizioni sul modo di usare le etichette.

[7] URI (Universal Resourse Identifier) è una sintassi standard per stringhe che identifica una risorsa e costituisce un termine per indirizzare e nominare oggetti (risorse) del WWW

Una volta definite le etichette, le relative informazioni devono essere trasmesse insieme al documento ***XML***, di modo che il destinatario del documento possa conoscere ciò che le etichette rappresentano per il documento [13] .

La definizione delle etichette permette di controllare la conformità dei documenti pubblicati, facilitandone la riusabilità [14] .

Gli strumenti adoperati per definire le etichette adoperate sono [15]:

- ***DTD***;
- ***XML Schema***.

2.2.2 DTD: Documento di definizione dei tipi

Le ***DTD*** [12] (*Document Type Definition*) definiscono la struttura di un documento ***XML*** e possono essere intese come una grammatica svincolata dal contesto.

Una ***DTD,*** mediante una serie di dichiarazioni e regole, stabilisce, in maniera rigorosa, la struttura del documento, specifica l'insieme e l'ordine delle etichette usate e gli attributi associati.

Il concetto di ***DTD***, ereditato da ***SGML***, è stato adottato in maniera diversa da ***XML***.

In ***SGML,*** le ***DTD*** devono essere utilizzate perché tutti i documenti appartengono ad un tipo e ogni documento deve rispettare le regole strutturali descritte nella ***DTD***. In ***XML,*** invece***,*** si possono realizzare documenti appartenenti ad un tipo e conformi ad una ***DTD***, oppure documenti non associati ad alcuna ***DTD***.

In ***XML***, quindi, l'uso delle ***DTD*** non è obbligato, ma è consigliato in presenza di un elevato numero di dati che devono essere ben strutturati. Le ***DTD*** costituiscono, infatti, un potente strumento di standardizzazione che permette di interfacciare applicazioni diverse.

La ***DTD*** contiene una rappresentazione formale e rigorosa di un tipo di documento. Qualsiasi applicazione che conosce la ***DTD*** può elaborare i documenti di quel tipo, anche se il documento è stato generato da una applicazione diversa.

Una ***DTD*** è costituita da:

1. una **dichiarazione del tipo di elementi**;
2. **dichiarazioni di tipi di elemento**: l'insieme di tipi di elemento definisce un linguaggio di marcatura ed è detto "*vocabolario*";

3. **dichiarazioni di attributi**: ogni attributo è associato ad un determinato tipo di elemento; i valori degli attributi, contenuti nell'istanza del documento, permettono di specificare le proprietà degli elementi che non corrispondono direttamente al testo che costituisce il documento;
4. **dichiarazioni di entità**: ogni entità rappresenta un'abbreviazione di una sequenza di caratteri. Il riferimento ad un'entità può, quindi, essere usato nell'istanza di documento al posto della sequenza di caratteri che l'entità rappresenta.

Tramite un opportuno "*algoritmo di validazione*" è possibile stabilire se il documento è conforme allo standard specificato dalla ***DTD***. In tal caso il documento si dice "*valido*".
Il compito di "*validare*" un documento è svolto dal "*Parser XML*", un programma che analizza il file ***XML*** e verifica se è scritto in maniera corretta, estraendo eventualmente dati da inviare ad un'altra applicazione.
Il "*Parser*" analizzando il documento ***XML*** visualizza i dati secondo la struttura illustrata nella ***DTD***: se i dati non sono "*coerenti*" con la ***DTD***, segnalerà l'errore.
Esistono due tipi di "*Parser XML*":

- "***Validanti***": verificano se un documento ***XML*** è "*ben formato*" ed è conforme ad una data ***DTD***;
- **"*Non validanti***": verificano solo se il documento è "*ben formato*".

La scelta di rendere il documento "*valido*" dipende dalle dimensioni: se il documento è di piccole dimensioni è sufficiente che sia "*ben formato*". Invece , se il file è di grandi dimensioni oppure deve far parte di un sistema informativo, è preferibile che sia "*valido*".
XML ha permesso di creare nuove etichette e strutture che hanno portato alla creazione di nuovi linguaggi:

- ***MathM*** *(Mathematical Markup Language*): sviluppato dal ***W3C*** per descrivere le espressioni matematiche;
- ***CML*** *(Chemical Markup Language*): permette di rappresentare strutture chimiche e molecolari;

- ***WML*** *(Wireless Markup Language*): consente di visualizzare le pagine Web nelle unità "*senza fili"*[8], come i telefoni cellulari e i "*PDA"* (*Personal Digital Assistant*); rappresenta il linguaggio di marcatura previsto per la definizione delle applicazioni lato cliente nello standard "*WAP"* (*Wireless Application Protocol*);
- ***CDF*** (*Channel Definition Format*) è una proposta di ***DTD*** della ***Microsoft*** per la descrizione di canali in linea, utilizzato in "*Microsoft Internet Explorer*";
- ***SMIL*** *(Synchronized Multimedia Integration Languag*e) consente di coordinare la rappresentazione di una vasta gamma d'elementi multimediali..
- ***cXML*** è un protocollo semplificato per la comunicazione di documenti affari fra applicazioni.

Altra caratteristica delle ***DTD*** è che non devono obbligatoriamente essere contenute nel documento, ma possono essere specificate in un file esterno.
Molti sono i vantaggi nell'utilizzo delle ***DTD***, tuttavia si hanno anche molti limiti:

- potere espressivo limitato;
- sintassi non ***XML***;
- capacità di lavorare solo con dati di tipo testo.

Quest'ultima limitazione comporta che le ***DTD*** non sono in grado di indicare efficientemente intervalli numerici nei documenti, essenziali nelle applicazioni B2B (Business To Business)[9] e in applicazioni e-commerce.[10]

2.2.3 XML Schema

Il limitato potere espressivo del linguaggio di definizione delle ***DTD*** ha portato il ***W3C*** a introdurre il linguaggio ***XML Schema*** [16], per esprimere vincoli più complessi nella rappresentazione della struttura di un documento ***XML*** [13] .
La motivazione principale alla base di ***XML Schema*** è consentire agli elaboratori e agli umani di conoscere il contenuto del documento, utilizzando strutture comuni, documentate in un formato comprensibile ad entrambi.

---

[8] Tali unità sono dette "*wireless*"
[9] B2B sono le applicazioni del commercio elettronico fra fornitori
[10] E-commerce applicazioni di commercio elettronico

Viene così definito un "*vocabolario*" per descrivere documenti ***XML***, facendo uso della sua stessa sintassi.

XML Schema ha diversi vantaggi rispetto alle ***DTD***, perché si avvale di una grammatica più ricca per descrivere la struttura degli elementi (ad es. può essere specificato l'esatto numero di occorrenze di un dato elemento, definire dei valori di riferimento), provvede alla tipizzazione dei dati (sono supportati tipi atomici di dati come stringhe, decimali), prevede l'utilizzo dei ***namespace*** e fornisce meccanismi d'inclusione e derivazione [12].

Quest'ultima caratteristica permette il riutilizzo di definizioni di elementi comuni e l'adattabilità di definizioni esistenti a nuove pratiche [15].

Le specifiche di ***XML Schema*** indicano la necessità di utilizzare due diversi documenti: almeno un "*documento schema*" che descrive la struttura ed il tipo di documento istanza ed un "*documento istanza*" in cui ci sono le informazioni che interessano.

La definizione di un "*documento schema*" contiene:

- **Definizioni di tipo**: in particolare ***XML Schema*** supporta due categorie distinte di tipi: i tipi "*semplici"* e tipi *"complessi"*. I "*tipi semplici*" sono ottenuti da un insieme di dichiarazioni di elementi e attributi; invece, nuovi "*tipi complessi*" possono essere definiti in termini di predefiniti "*tipi semplici*", usando due "*forme di derivazione*": la "*generazione di liste*" e la "*restrizione*".

  Le dichiarazioni sono associazioni fra un nome di un elemento o di un attributo e un vincolo specifico che ne governa la comparsa nei documenti ***XML***.
- **Dichiarazioni di elementi e attributi**: una dichiarazione di elemento associa un tipo a un nome di elemento in un certo contesto[11].

Riepilogando, le caratteristiche principali di ***XML Schema*** sono :

- gestione di tipi primitivi di dati (stringhe,interi,date,etc.) e tipizzazione degli attributi;
- gestione per tipi di dati definiti dall'utente ed ereditarietà;
- flessibilità nella definizione dei vincoli di cardinalità delle relazioni;

[11] Per una dichiarazione locale di elementi, il contesto è il tipo complesso che la contiene, mentre per le dichiarazioni globali, il contesto è l'elemento "radice" di un documento.

- modularità delle definizioni mediante meccanismi di importazione ed inclusione degli schemi.

2.2.4 Fogli di stile

Una delle caratteristiche principali di ***XML*** è la separazione tra la struttura dei dati e la loro rappresentazione.
Teoricamente, quindi, è possibile utilizzare una sintassi di uno stile qualsiasi per estrarre i dati dal documento e visualizzarli, poiché ***XML*** si occupa esclusivamente di come il documento o i dati devono essere strutturati, senza occuparsi della rappresentazione.
Per visualizzare i dati di un documento è necessario un altro documento denominato "*foglio di stile*" che contiene le regole necessarie per rappresentare e visualizzare il documento ***XML*** (ad esempio i margini, i tipi di carattere, particolari visualizzazioni dell'informazione contenuta tra certe etichette o particolari eventi da associare).
È possibile realizzare diverse rappresentazioni di un documento ***XML,*** creando diversi "*fogli di stile*", ciascuno con le caratteristiche richieste.
I linguaggi candidati per rappresentare le informazioni in ***XML*** sono tre:

1) **CSS** (*Cascading Style Sheet*) una tecnologia, rilasciata dal ***W3C*** per ***HTML***, che, se per un verso ha il vantaggio di essere diffusa, sperimentata e conosciuta, d' altro canto presenta l'inconveniente di non riuscire a visualizzare documenti ***XML*** in modo diverso da quella che è la struttura e quindi lontano dal concetto di estensibilità.
2) **DSSL** (*Document Style Semantics and Specification Language*) è un linguaggio di stile nato per ***SGML***, dotato della flessibilità necessaria, ma anche molto complesso e poco diffuso.
3) **XSL** (*eXstensible Style Language*) [17] è un linguaggio proprietario, realizzato appositamente per ***XML*** per raccogliere i pregi di ***CSS*** e di ***DSSL***, unendo la semplicità alla potenza. L'idea alla base di ***XSL*** è trasformare un documento ***XML*** in ***HTML*** e renderlo visualizzabile in un qualsiasi browser ***XLS***.
   L'***XSL*** ha due funzionalità distinte che permettono di ottenere ciò, mediante appositi linguaggi:
   - **Trasformazione**: il documento ***XML*** è trattato aggiungendo delle parti e trasformando il file sorgente in un altro file, ad esempio in un file ***HTML***,

tramite il linguaggio di **XSLT** (*XSL Transformations*), usato anche al di fuori di ***XSL*** tanto da diventare uno standard separato;

- **Formattazione**: le informazioni da elaborare sono divise in blocchi, delimitate dalle etichette, ciascuno con opportune proprietà di visualizzazione, attraverso l' utilizzo del il linguaggio di formattazione **XSL FO** (*XSL Formatting Objects*) per specificare come visualizzare un documento in base alla sua struttura.
  XSLT permette la trasformazione di documenti ***XML*** in ***HTML***, senza l'utilizzo del linguaggio di formattazione.

Inoltre, permette la strutturazione di documenti ***XML***, così che differenti tipi di documenti possano essere fatti corrispondere fra loro, permettendo la traduzione delle informazioni da un vocabolario ***XML*** all'altro.
Risulta, quindi, facile trasformare documenti conformi ad una certa ***DTD*** in documenti conformi ad un'altra ***DTD***.
Indicando come devono essere visualizzati gli elementi di formattazione si ottiene una specifica sulla visualizzazione di un documento conforme alla ***DTD*** originaria.

2.2.5 Collegamenti e riferimenti ipertestuali

I collegamenti ed i riferimenti ipertestuali permettono di collegare documenti localizzati in un qualsiasi punto del Web.
In ***HTML***, i "*collegamenti ipertestuali*" mettono in relazione il documento che li contiene ad un altro presente sul Web, con relazioni di tipo unidirezionale, "*uno-a-uno*", dove gli ***URL*** puntano solo ad un documento e non si può effettuare l'indirizzamento a porzioni di un determinato documento.
XML estende i "*collegamenti ipertestuali*", visti in ***HTML***, tramite tre linguaggi , fruibili separatamente, ma che permettono di ottenere migliori risultati attraverso un utilizzo congiunto:

1) **XPointer** (*XML Pointer Language*) definisce i costrutti per indirizzare strutture interne di un documento ***XML***; la navigazione nel documento è effettuata tramite i meccanismi definiti in ***XPath***.
2) **Xlink**[18]o **XLL** (*eXstensible Linking Language*) descrive come collegare due documenti;

3) **XPath** (*XML Path Language*) definisce i meccanismi di base per navigare attraverso la struttura di un documento ***XML***.

XPointer è un linguaggio per determinare dove è localizzata la risorsa che si vuole collegare, tra le novità introdotte:

- possibilità di referenziare un punto interno ad un documento, anche se sprovvisto dell'identificatore come in ***HTML***;
- possibilità di referenziare in maniera più precisa gli elementi, le stringhe e tutti gli oggetti contenuti in un documento;
- indica chiaramente le entità referenziate e le relazioni gerarchiche fra le varie parti della struttura di un documento ***XML***, rendendo i collegamenti più facili da interpretare.

Xlink permette di realizzare "*collegamenti ipertestuali*" fra vari tipi di risorse[12], più potenti e flessibili di quelli disponibili in ***HTML***.
In tale linguaggio un riferimento ad un altro documento è costituito da un URI e da un ***XPointer***. Altre caratteristiche di ***Xlink*** sono:

- "*collegamenti ipertestuali*" che puntano a più documenti: si possono realizzare dei collegamenti che referenziano più di un documento, quindi relazioni "*uno-a-molti*";
- "*collegamenti ipertestuali bidirezionali*": non si ha un punto di partenza e uno di arrivo il che permette di spostarsi da un documento all'altro e viceversa;
- raccolte di "*collegamenti ipertestuali*": forniscono la possibilità di filtrare, ordinare, analizzare, elaborare una collezione di "*collegamenti ipertestuali*".
- "*collegamenti ipertestuali*" possono essere memorizzati fuori di ogni risorsa che essi collegano;
- "*collegamenti ipertestuali*" possono avere la loro semantica, tipicamente assegnata attraverso attribuite regole.

Infine, **XPath** è stato progettato per muoversi in maniera semplice e non ambigua all'interno dei documenti ***XML*** e per selezionare porzioni di informazioni.

[12] La risorsa è qualcosa che è indirizzabile tramite un URL e un link fra due risorse rende rende comprensibile la relazione fra le due risorse.

Tale caratteristica lo rende uno strumento utile per la ricerca all'interno dei documenti ***XML***, anche per altri linguaggi come ***XSLT*** e ***XPointer***.
Un tipico documento ***XML,*** concettualmente, è visto come un albero di nodi, dove c'è un nodo radice che contiene tanti altri tipi di nodi e, un'espressione ***XPath*** è composta da una serie di passi attraverso l'albero ***XML***, dove ogni passo è una dichiarazione di un criterio di ricerca (ad es. trova le parole in corsivo nella pagina attuale).

2.2.6 DOM : documento di modellazione degli oggetti

Ogni documento ***XML*** può essere associato ad una struttura ad albero, definita dal ***DOM***[19]*(Document Object Model*), data dall'annidamento degli elementi e degli attributi associati agli elementi.
La ***DOM*** è un'interfaccia ***API*** *(Application Programming Interfac*e) per ***XML*** indipendente dalle piattaforme e dai linguaggi ed attualmente è l'approccio più utilizzato dai "*Parser*".
La metodologia ***DOM*** permette l'analisi di documenti ***XML*** per garantire l'accesso alle informazioni, mediante un insieme di interfacce standard (metodi, oggetti e così via) che permettono di manipolare il contenuto dei documenti ***XML***.
I documenti ***XML*** sono strutturati gerarchicamente, quindi, il ***DOM*** li rappresenta con una struttura ad albero e l'applicazione che si basa sulla ***DOM*** per l'analisi dei documenti ***XML***, costruisce in memoria una rappresentazione ad albero e permette di navigare attraverso l'albero, grazie ad una serie di semplici regole.
Il vantaggio delle ***DOM*** sulla metodologia "*Event-Driven*" è che ci si può riferire ad un qualsiasi nodo dell'albero a prescindere dal nodo in esame.
Di contro, si ha che è necessario che il software tenga in memoria l'albero per tutto il tempo dell'analisi, con conseguente dispendio di risorse, a scapito di velocità ed efficienza, c'è però il controllo totale sui dati.

2.3 XML Voice

2.3.1 Introduzione

La multimodalità sta passando da interesse puramente accademico ad un interesse commerciale, spinta da tre aspetti:

- il miglioramento dell'accuratezza e della robustezza del riconoscimento vocale avvenuto negli ultimi anni che rende la realizzazione di applicazioni vocali più realistiche;
- lo sviluppo della telefonia mobile con la progressiva miniaturizzazione dei dispositivi cellulari che rende la digitazione su tastiera lenta e scomoda;
- le applicazioni vocali legati all'infrastruttura Web e la presenza di ambienti di sviluppo potenti rendono lo sviluppo di applicazioni più semplici ed efficienti.

Rispetto agli anni passati in cui le applicazioni multimodali erano basate su tecnologie proprietarie, oggi, si sta cercando di definire uno standard unico per l'interazione multimodale nel Web.
Le applicazioni di tale tecnologia sono molteplici: si va dai sistemi di apprendimento delle lingue straniere, agli strumenti per disabili, ma il campo più promettente è quello della telefonia mobile dove si potranno realizzare interfacce solo vocali, che permetteranno l'interazione con apparecchi telefonici senza schermo.
La voce, però, deve essere utilizzata non tanto per il suo carattere innovativo, ma perché dà alle applicazioni un valore aggiunto importante, specialmente in dispositivi piccoli con tastiere scomode, oppure in applicazioni usate quando l'utente ha le mani e gli occhi impegnati (ad esempio durante la guida).[13]
Nel futuro, i browser vocali aiuteranno l'accesso sui servizi basati sul Web e saranno utilizzati nella prossima generazione dei servizi cliente (che diventeranno portali Web voce) in cui gli utenti potranno interagire mediante i tasti e comandi vocali e utilizzare i dialoghi naturali.

[13] In tale ambito quindi risulta interessante la cosiddetta navigazione vocale, che permette di percorrere i livelli di menù.

Per raggiungere l'obiettivo, il **Voice Browser Working Group**[20]del ***W3C***, sta rilasciando specificazioni per quattro diversi linguaggi di marcatura, conosciuti come ***W3C Speech Interface Framework***.

Tali linguaggi, progettati per integrarsi con i linguaggi esistenti, coprono diversi aspetti delle applicazioni vocali: dialogo, chiamata di comandi, sinterizzazione e riconoscimento del parlato.

Gli elementi del ***W3C Speech Interface Framework*** sono:

- **XMLVoice 2.0** [21] che specifica le strutture di base dei dialoghi di conversazione che caratterizzano il parlato sintetizzato, l'audio digitalizzato, il riconoscimento del parlato e gli ingressi a chiavi DTMF, registrazione del parlato, telefonia, e iniziative di conversazione miste.
- **Speech Recognition Grammar Specification** (**SRGS**) [22] che è la grammatica relativa al parlato e agli ingressi DTMF[14]. Tale grammatica abilita gli sviluppatori a descrivere le parole e le frasi che possono essere riconosciute da un motore di riconoscimento del parlato. La sintassi può essere specificata in due forme equivalenti che utilizzano o XML o una sintassi equivalente Augmented BNF (ABNF).
- La **Speech Synthesis Markup Language** (SSML)[23] definisce un linguaggio di marcatura per specificare istruzioni al sintetizzatore del parlato circa la pronuncia di parole e frasi: può essere selezionata la caratteristica della voce e la velocità del parlato, volume, tono ed enfasi. SSML mette a disposizione delle etichette che saranno utilizzate dagli sviluppatori per migliorare le caratteristiche di presentazione di riferimento del sintetizzatore del parlato.
- La **Semantic Interpretation Specification** [24]descrive le annotazioni delle regole grammaticali per estrarre i risultati semantici del riconoscimento. L'obiettivo per l'uscita XML è l'Extensible Multimodal Annotation Markup Language (EMMA)[25] il cui sviluppo è ancora in corso.
- **Call Control XML** (CCXML) (Attuale Bozza di Lavoro Giugno 2003) [26] permette di specificare controlli fini sul parlato, su risorse e su risorse telefoniche in una piattaforma telefonica XMLVoice.

[14] DTMF è particolarmente prezioso in condizioni di rumore o quando non è possibile parlare ad esempio perché ci sono problemi legati alla riservatezza

2.3.2 Il linguaggio XMLVoice

XMLVoice [27](*Voice eXtensible Markup Language*) è un linguaggio di marcatura di dialogo, sviluppato dal ***W3C***, per diventare il linguaggio di riferimento di tutte le applicazioni basate sul Web che implicano interazioni con il parlato sintetizzato e il suo riconoscimento, con l'audio digitale ed interazioni con gli ingressi DTMF[15]e più in generale con i servizi voce.

Tale linguaggio potrà essere usato per costruire applicazioni vocali avanzate, in una forma facilmente integrabile con gli altri linguaggi definiti dal ***W3C***. Invece, eventuali necessità particolari nei servizi voce, richiederanno sempre applicazioni dedicate.

Le applicazioni possibili di tale linguaggio sono molteplici: accesso automatico alle informazioni aziendali, come il servizio clienti, gestione automatica degli ordini telefonici; accesso alle informazioni pubbliche o ad informazioni personali (agende personali); assistenza durante la comunicazione.

XMLVoice vuole portare il potere delle applicazioni Web nelle applicazioni vocali, permettendo di trasportare i contenuti, senza dovere realizzare applicazioni per la gestione delle risorse che devono operare ad un basso livello di programmazione.

Obiettivi di ***XMLVoice*** sono:

- sintetizzazione del parlato (da testo a parlato);
- sintetizzazione di file audio;
- Riconoscimento e registrazione del parlato;
- Riconoscimento di DTMF;
- Controllo del flusso di dialogo.

Particolari funzionalità della telefonia, ad esempio il trasferimento di chiamata;

L'architettura prevista è:

[15] DTMF (Dual Tone Multi Frequency) è il bitono a frequenza molteplice del telefono digitale, che permette di assegnare un suono specifico ad ogni tasto dell'apparecchio telefonico (composto da due frequenze: alta e bassa) e che consente la sua individuazione da parte di apparecchiatura automatiche

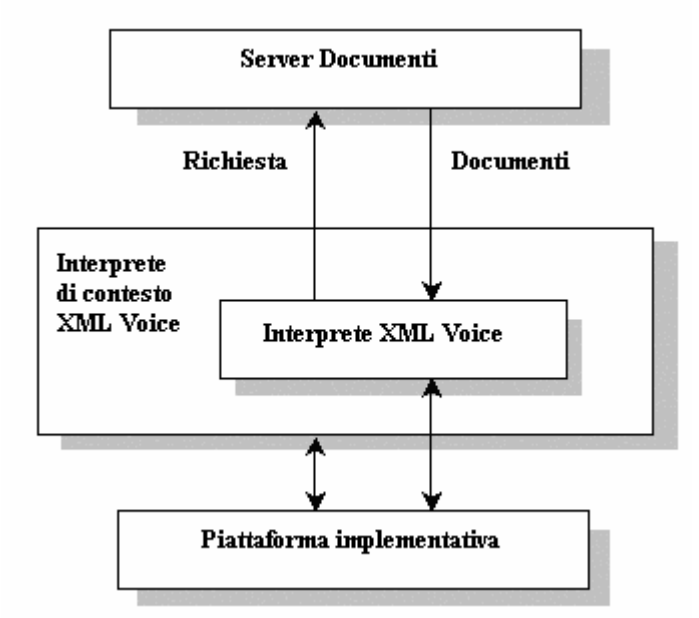

Figura 2.2: Architettura prevista per XML Voice

Un servizio voce è costituito da una sequenza di dialoghi d'interazione fra un utente e la "*Piattaforma implementativa*".

I dialoghi sono realizzati dal "*server dei documenti*" che è un modulo che può essere interno od esterno alla piattaforma d'implementazione e specifica ogni dialogo d'interazione che deve essere condotto da un "*Interprete XMLVoice*", mantiene la logica di servizio ed esegue operazioni sulle basi di dati.

L'utente influenza l'interpretazione dei dialoghi ed è associato alle richieste sottoposte al "*server dei documenti*" che risponde realizzando un altro documento *XMLVoice* per continuare la sessione utente.

Il "*server dei documenti*" elabora le richieste dell'"*interprete XMLVoice"*, attraverso "*l'interprete di contesto XMLVoice*" che controlla gli ingressi degli utenti parallelamente all'Interprete XMLVoice.

La "piattaforma di implementazione" è controllata dall' "interprete di contesto XMLVoice" e dall'"interprete XMLVoice".

2.3.2.1 Principi di progettazione

XMLVoice è una applicazione XML che:

1. promuove la portabilità dei servizi fra diverse piattaforme di implementazione, attraverso l'astrazione delle risorse e un linguaggio comune a tutti;
2. funge da intermediario fra diverse piattaforme, adattando diversi formati audio, grammaticali per il parlato e per gli schemi URI; in tal modo le applicazioni non dovranno occuparsi dei dettagli specifici delle singole piattaforme e per favorire l'interoperabilità è utilizzato un formato grammaticale comune del W3C, il **SRGR** (Speech Recognition Grammar Specification).[28]
3. facilita gli autori nella realizzazione dei più comuni tipi di interazione e fornisce gli strumenti per supportare forme di dialogo complesse;
4. ha una semantica ben definita che preserva l'intento degli autori durante l'interazione con altri utenti;
5. riconosce interpretazioni semantiche dalla grammatica e rende le informazione disponibile alle applicazioni;
6. ha un meccanismo di controllo del flusso;
7. abilita la separazione della logica di servizio dal comportamento di interazione;
8. non è utilizzata per elaborazioni intensive o operazioni sulle basi di dati; tali operazioni sono svolte esternamente all'interprete dei documenti (ad esempio da un server dei documenti);
9. i servizi generali, la gestione dello stato, generazione di dialoghi sono svolti esternamente all'interprete di documenti;
10. fornisce diversi modi per collegare documenti usando URI;
11. fornisce modi per identificare esattamente i dati da sottoporre al server per presentarli;
12. permette agli autori di documenti di non occuparsi della gestione delle risorse di dialogo o del controllo di interazione, delegando ciò alla piattaforma di implementazione.

2.4 RDF

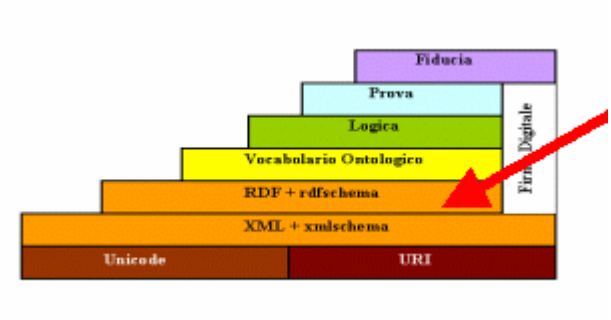

Figura 2.2: Posizione di RDF e RDF Schema nell'architettura del Web Semantico

RDF (***Resource Description Framework***) [29] costituisce il primo passo compiuto dal ***W3C*** nella direzione del Web Semantico dopo ***XML***, per giungere ad una sintassi per codificare i dati.

RDF è un meccanismo per fare affemazioni circa i dati, ma, come dice lo stesso nome non è un linguaggio, ma un modello, dove il tipo di dati cui si fa riferimento è chiamato "*metadato*".[15]

RDF, quindi, rappresenta un meccanismo per descrivere le risorse associate ad ***URI*** (presenti o meno sul Web), in maniera indipendente dal dominio di applicazione ("*general purpose*").

Il fatto che il meccanismo per la descrizione di risorse sia neutrale rispetto ai domini applicativi, implica che non è rivolto ad un particolare dominio di applicazione, né definisce a priori il vocabolario e la semantica di qualche dominio, ma che il suo scopo essenziale è proprio quello di essere adattabile alla descrizione di informazioni relative ad un qualsivoglia dominio.

RDF si propone di:

- rappresentare meta-informazioni relative alle risorse;
- creare una struttura comune che permette di esprimere le informazioni da scambiare fra applicativi, senza perdita di significato.

Per realizzare il suo compito, non conosce la sintassi di ciò che rappresenta.

Esso dà solo un modello per rappresentare "metadati", definendo una convenzione sintattica e un semplice modello dati per rappresentare dati semantici in modo processabile dalle macchine[30].

La definizione formulata del ***W3C*** propone, per avere un formato scambiabile dalle macchine, ***XML*** come metalinguaggio per la rappresentazione del modello ***RDF***, quindi ***RDF*** risulta basato su ***XML***.

Con ***RDF*** è possibile condividere conoscenza fra applicazioni differenti, senza perdita di significato, con un formato di rappresentazione indipendente dall'applicazione.

Mediante ***RDF***, quindi, sono stabilite nuove relazioni fra documenti, immagini e qualunque altro tipo di risorsa Web, in modo da consentire l'interoperabilità fra le applicazioni.

RDF ha un sistema a classi, simile a quelli della programmazione ad oggetti: una collezione di classi è detta "*schema*" e le classi formano una gerarchia che, tramite il meccanismo delle sottoclassi, consente la specializzazione dei concetti e la condivisione delle definizioni dei "*metadati*".

Il modello dati di ***RDF*** consiste in tre tipi di oggetti e si basa sull'idea che le "cose" da descrivere hanno proprietà con dei valori associati e che su questa base si vuole fornire un semplice meccanismo per definire dei fatti relativi alle risorse:

- **Risorsa:** è una qualunque cosa nominabile tramite un ***URI*** e un identificativo addizionale; risorsa può inoltre essere un oggetto materiale non direttamente accessibile via Web o anche una entità del tutto immateriale.
- **Proprietà:** specifica una caratteristica di una risorsa, definendo una relazione binaria tra la risorsa stessa ed un'altra risorsa, oppure tra la risorsa e dei valori atomici; mediante le proprietà si definiscono i valori ammessi, il tipo delle risorse che possono essere descritte, le relazioni esistenti con altre proprietà; quindi le proprietà definiscono caratteristiche, attributi o relazioni usate per descrivere una risorsa;
- **Espressioni**: assegnano un valore per una proprietà in una specifica risorsa.

Si chiama "*espressione **RDF***" o "*tripla*" l'insieme di una specifica risorsa con una specifica proprietà ed il valore di questa proprietà per la risorsa: rispettivamente gli elementi sono detti soggetto, predicato e oggetto.

Il soggetto e il predicato devono essere obbligatoriamente delle risorse, mentre l'oggetto può essere sia una risorsa sia un letterale.

Ad esempio volendo affermare che “Giovanni Canfora” è l’autore di una tesi presente sul Web all’indirizzo http://www.tesicanfora.it/tesiWeb.doc allora si ha in RDF che:

Risorsa : http://www.tesicanfora.it/tesiWeb.doc
Proprietà: autore
Valore : Giovanni Canfora
e si esprime ciò mediante la dichiarazione:

http://www.tesicanfora.it/tesiWeb.doc **ha autore** Giovanni Canfora

Un “*modello* ***RDF***” può essere rappresentato come un grafo diretto etichettato: graficamente ogni risorsa è identificata con un ovale, le proprietà mediante frecce e i valori mediante scatole.

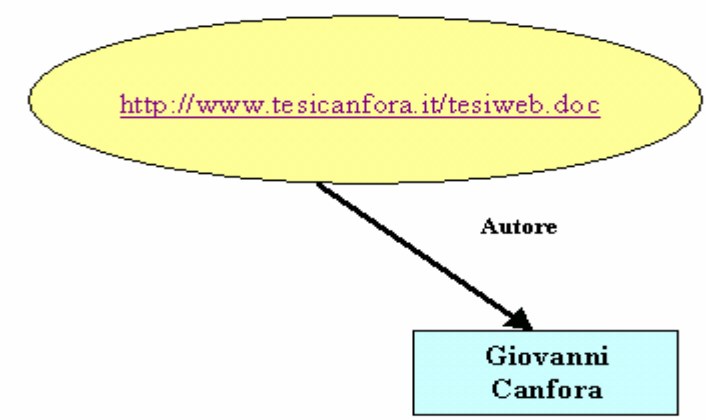

Figura 2.3 : Esempio rappresentazione mediante grafo di una espressione RDF

Un insieme di proprietà che fanno riferimento alla stessa risorsa è detta **descrizione**.
Risulta interessante che l’oggetto, può a sua volta essere il soggetto di una nuova tripla, premettendo di rappresentare situazioni complesse.

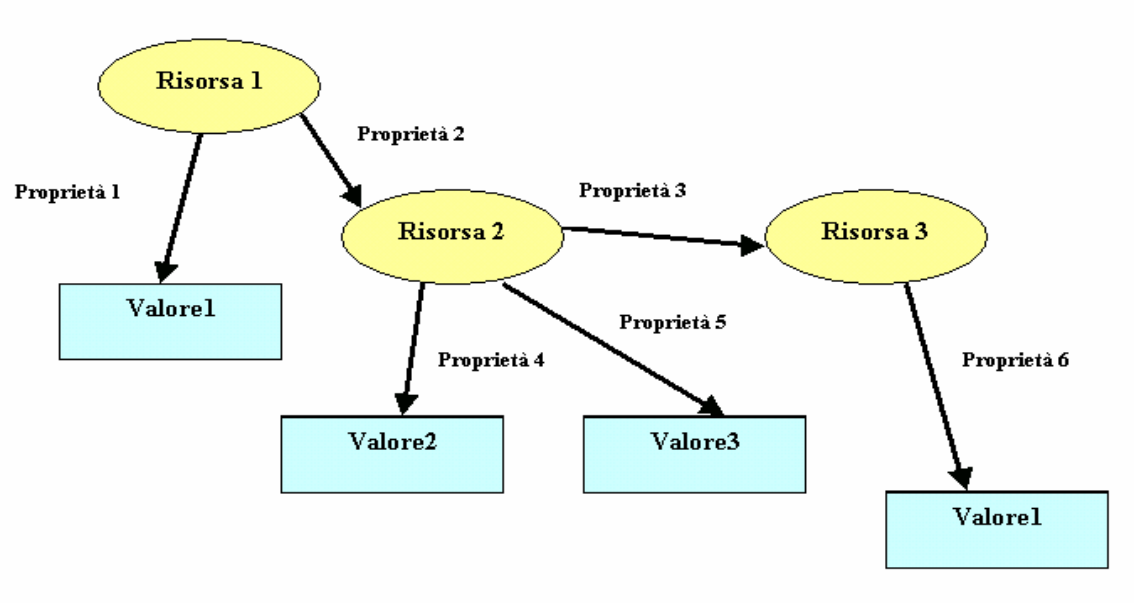

Figura 2.4 : Esempio rappresentazione mediante grafo di una espressione RDF multipla

Inoltre tale rappresentazione ha il vantaggio di essere incrementale, perché permette facilmente l'aggiunta di nuova conoscenza, senza richiedere la modifica delle terne precedentemente costruite.

Quando è necessario fare riferimento a più di una risorsa ***RDF*** definisce tre tipi di **contenitori** (container):

- ***bag***: è una lista non ordinata di risorse o costanti, utilizzato per dichiarare che una proprietà ha valori multipli e sono ammessi valori multipli;
- ***sequenze***:è una lista ordinata di risorse o costanti, utilizzato per dichiarare che una proprietà ha valori multipli e sono ammessi valori multipli;
- ***alternative***: è una lista ordinata di risorse o costanti, che rappresentano una alternativa per un valore (singolo) di una proprietà;

Le proprietà possono appartenere al contenitore o ai singoli elementi.

La presenza degli ***URI*** garantisce che i concetti siano legati ad una definizione univoca che chiunque può trovare nel Web.

Inoltre si ha che:

- la proprietà può essere descritta dettagliatamente all'interno della pagina identificata dall'***URI***, utilizzando meccanismi che permettono di aggiungere semantica;

- la proprietà identificata dall'***URI*** potrà essere usata da chiunque, in qualsiasi pagina del Web Semantico per descrivere una qualsiasi cosa, mantenendone però il significato originale;
- è risolto il problema del conflitto sui nomi;
- avendo trasformato un letterale in una risorsa, si può utilizzarla come soggetto o oggetto in una asserzione ***RDF***;

RDF rappresenta conoscenza, però non ha nessun meccanismo di deduzione automatica, quindi eventuali inferenze si potranno fare solo avvalendosi di una semantica esplicita e di motori inferenziali esterni.

Riepilogando, RDF è lo strumento basilare per la codifica, lo scambio e il riutilizzo di "metadati" strutturati, e consente la condivisione di conoscenza tra applicazioni che interagiscono tra loro, scambiandosi sul Web informazioni comprensibili dalle macchine.

2.5 RDF Schema

RDF consente di definire un semplice modello dati per descrivere le proprietà di risorse e le relazioni fra le risorse stesse, ma non permette di definire le relazioni fra proprietà e risorse o di organizzare le risorse in tipi o "classi" con loro proprietà specifiche.

Inoltre, non fornisce alcun meccanismo per dichiarare le proprietà, per definirne i vincoli d'applicabilità, per organizzarle gerarchicamente.

A tale scopo ***RDF*** è stato arricchito con un semplice sistema di tipi, ***RDF Schema*** (*Resourse Description Framework Schema*) [31] che definisce le primitive per definire modelli di conoscenza legati ad un approccio basato sui modelli.

Per esempio, una risorsa può essere definita come istanza di una o più classi e le classi possono essere organizzate in modo gerarchico, derivando per ereditarietà nuova conoscenza.

Attraverso uno *Schema RDF,* è possibile definire tutti i termini usati nelle asserzioni ***RDF***, assegnando loro un significato specifico come in una sorta di "*vocabolario*" liberamente creato dalle singole comunità di utenti.

Tali "vocabolari" risiederanno in opportuni documenti, reperibili e accessibili sul Web.

Uno *Schema RDF* raccoglie la dichiarazione delle categorie alle quali possono appartenere le entità che si vogliono descrivere e le relazioni che le legano.

Il linguaggio di specifica ***RDF(S)*** è un linguaggio dichiarativo semplice che si esprime attraverso la sintassi di serializzazione ***RDF/XML*** e si avvale del meccanismo dei ***namespace XML*** per la formulazione delle ***URI***.

I vantaggi introdotti dai namespace giocano un ruolo cruciale nello sviluppo degli schemi RDF perché consentono il riutilizzo di termini definiti in altri schemi, permettendo di impiegare concetti e proprietà già dichiarati per un dominio o precisati per incontrare le esigenze di una particolare comunità di utenti.

Le primitive di modellazione di base utilizzabili per costruire uno schema per un dominio specifico ***RDF(S)*** sono:

- **classi e sottoclassi di dichiarazioni**: permettono la definizione di classi di gerarchie;
- **proprietà e sottoproprietà di dichiarazioni**: per costruire gerarchie di proprietà;
- **domini e intervalli di dichiarazioni**: per restringere le possibili combinazioni di proprietà e classi;
- **dichiarazioni di tipi**: per dichiarare una risorsa come una istanza di una specifica classe.

In particolare, una Classe è una qualunque risorsa che abbia una proprietà "***rdf:type***" ed il cui valore sia "***rdfs:Class***".

In uno Schema RDF vengono, inoltre, documentate le definizioni ed i vincoli d'uso delle proprietà che caratterizzano le classi.

In ***RDF*** tutte le Proprietà devono essere definite come istanze della classe "***rdf:Property***", mentre ***RDF(S)*** definisce un modo per specializzare le proprietà, come avviene per le classi, attraverso la proprietà "***rdfs:subPropertyOf***"

Inoltre, attraverso le proprietà "***rdfs:range***" e "***rdfs:domain***", limita il campo di applicabilità delle proprietà definite a livello utente, ponendo vincoli sul loro d*ominio* (quali classi sono lecite come s*oggetto* della proprietà) e sul loro c*odominio* (quali classi sono lecite come v*alore* della proprietà).

RDF(S) fornisce strumenti per aggiungere ulteriori descrizioni sulle risorse, ma senza prescrivere come tali descrizioni debbano essere utilizzate da un' applicazione.

Riepilogando, le dichiarazioni ***RDF(S)*** sono solo delle *descrizioni* che possono essere considerate vincolanti, soltanto se l'applicazione che li tratta è stata progettata per seguire quest'interpretazione.

RDF(S) è un linguaggio dichiarativo che si limita a fornire un mezzo per fornire informazioni addizionali sui dati, ma sono le singole applicazioni a stabilire individualmente come affrontare l'eventualità che i dati non siano conformi agli Schemi.

Bibliografia

[1] Fensel D. and Musen M.A., "*The Semantic Web: A Brain for Humankind*" IEEE INTELLIGENT SYSTEMS, pp 24-25 (March/April 2001)

[2] Benjamins V. R , Contreras J., Corcho O., and Gómez-Pérez A.,"*Six Challenges for the Semantic Web"* http://www.isoco.com/isococom/whitepapers/files/SemanticWeb-whitepaper-137.pdf (April 2002)

[3] Berners-Lee T. and Fischetti M., "*Weaving the Web: The Original Design and UltimateDestiny of the World Wide Web by Its Inventor.*", Harper (San Francisco, 1999)

[4] Berners-Lee, Tim; Hendler, James; Lassila, Ora. "*The Semantic Web*". Scientific American 284(5), pp. 34-43. May 2001

[5] Hendler J., "*Agents and the Semantic Web*"; IEEE INTELLIGENT SYSTEMS, pp 30-37 (March/April 2001)

[6] Berners-Lee Tim, "*Principles of design*" http://www.w3.org/DesignIssues/principles.html (1998)

[7] Miller E., "*Semantic Web Principles"* http://www.w3.org/2001/09/06ecdl/slide140.html

[8] http://www.w3c.org/2000/Talks/1206-xml2k-tbl/slide10-0.html

[9] Swartz A., "*The Semantic Web In Breadth*", http://logicerror.com/semanticWeb-long (May 2003)

[10] *"Resource Description Framework (RDF): Model and Syntax specification"*, www.w3.org/TR/REC-rdf-syntax-19990222/ (W3C recommendation 22 February 1999)

[11] Decker S., Fensel D. et al., "*The Semantic Web: The Roles of XML and RDF*", IEEE Internet Computing, pp 63-74 (September/ October 2000)

[12] Bergolz A., "*Extending Your Markup: An XML Tutorial* ", IEEE Internet Computing, pp.74-79 (July/August 2000)

[13] Grado-Caffaro M. A., "*The Challenge that XML Faces*", IEEE Tecnology News, pp.15-18 (October 2001)

[14] "*Introduction to XML and Related Tecnologies*", Student Notebook, IBM Learning Services (2001)

[15] Klein M., "*XML ,RDF, and Relativies*", IEEE Intelligent Systems, pp 26-28 (March/April 2001)

[16] Fallside D.C., "*XML Schema part 0. Primer –W3C candidate recommendation*", Tecnical report, World Wide Web Consortium (October 2000)

[17] http://www.w3.org/Style/XSL/ (W3C Recommendation 27 June 2001)

[18] http://www.w3.org/TR/xlink/. (W3C Recommendation 27 June 2001)

[19] Wood L. et al., "*Document Object Model (DOM), level 1 specification 1.0-W3CRecommendation 1 Ottobre 1998"*, Technical Report, W3C (1998)

[20] http://www.w3c.org/Voice/Group/

[21] http://www.w3c.org/TR/2004/PR-voicexml20-20040203/ (W3C Proposed Recommendation 3 February 2004)

[22] http://www.w3c.org/TR/speech-grammar/(W3C Recommendation 16 March 2004)

[23] http://www.w3c.org/TR/speech-synthesis/ (W3C Candidate Recommendation 18 December 2003)

[24] http://www.w3c.org/TR/semantic-interpretation/ (W3C Working Draft 1 April 2003)

[25] http://www.w3c.org/TR/emma/ (W3C Working Draft 18 December 2003)

[26] http://www.w3c.org/TR/ccxml/ (W3C Working Draft 12 June 2003)

[27] http://www.w3c.org/TR/voicexml20/ (W3C Recommendation 16 March 2004)

[28] http://www.w3c.org/TR/2004/PR-voicexml20-20040203/#ref_SRGS(W3C Proposed Recommendation, December 2003)

[29] Lassila O. and Webick R., *"Resource DescriptionFramework (RDF) Model and Syntax Specification."*, www.w3.org/TR/PR-rdf-syntax (W3C Recommendation January 2002)

[30] Fensel D., "*The semantic Web and its languages*", IEEE Intelligent Systems, pp.67-73 (November/December 2000)

[31] Brickley D. and Guha R.V., *"Resource Description Framework (RDF) Schema Specification",*www.w3.org/TR/PR-rdf-Schema(W3C Proposed Recommendation Marzo 1999)

[32] http://www.w3.org

Appendici

Appendice A: SGML

La nascita di quello che poi diventerà l'SGML è da datare alla fine degli anni sessanta.In quel periodo all'IBM ci si pose il problema di costruire un sistema che consentisse lo scambio e l'elaborazione dei documenti legali.
Dallo studio condotto si stabilirono tre fatti importanti:

- i programmi devono consentire l'utilizzo di una rappresentazione comune dei documenti; indipendentemente dal contenuto e dal sistema di elaboratori, devono poter essere descritte con un linguaggio comune, per garantire l'uniformità delle applicazioni che elaborano i vari tipi di documenti.
- il formato comune deve essere sufficientemente flessibile, in modo da garantire una rappresentazione per ogni documento, qualunque sia la sua struttura.
- per ogni tipo di documento è necessario stabilire delle regole alle quali ogni istanza di quel tipo si deve attenere, in modo da generale la struttura di quel particolare tipo di documento.

Nel 1969 ,un gruppo di ricercatori dell'IBM, composto da C. Goldfarb, E. Mosher e R. Lorie, mise a punto un linguaggio chiamato GML (Generalized Markup Language).
Per tale linguaggio decisero di adottare un *linguaggio di marcatura*.
Il concetto di marcatura è molto semplice: il documento è costituito da testo e da comandi di marcatura che stabiliscono in che modo deve essere trattato il testo. In genere i comandi di marcatura sono racchiusi fra i simboli di minore (<) e maggiore (>) ed il testo è racchiuso fra due comandi di marcatura (<comando> testo </comando>). I comandi di marcatura sono chiamati anche tag (<tag>).
Il linguaggio prototipo ideato identificava gli elementi strutturali che specificavano la natura astratta (anziché gli attributi di formattazione) delle informazioni che contenevano.
Gli attributi di formattazione erano inclusi in file distinti, chiamati *fogli stil*e, con i quali gli elaboratori potevano formattare gli elementi e rappresentare un documento finito.

I documenti potevano essere elaborati in modo affidabile soltanto se erano conformi a uno standard, quindi, a delle regole per formalizzare la struttura dei vari tipi di documenti.

Le regole erano racchiuse nelle Document Type Definition (in breve DTD) e rappresentavano la definizione formale del tipo di documento in esame. Ogni documento doveva rispettare la relativa DTD, in tal caso si affermava che il documento era valido, diversamente si affermava che quello non era un documento valido per la DTD data.

Le DTD stabilivano, in linea di massima, quale tipo di tag era possibile utilizzare all'interno di un documento e quale doveva essere il loro ordine.

La separazione fra elementi di rappresentazione ed elementi di conformità di un documento offriva una grande flessibilità agli sviluppatori, poiché i file DTD e i fogli stile potevano essere facilmente modificati senza intervenire direttamente sul codice di marcatura dei dati.

Nel 1974, Goldfarb dimostrò che era possibile utilizzare un *parser* (il software capace di analizzare la struttura e la sintassi di un documento) per verificare la conformità di un documento a un determinato standard, senza elaborare realmente il documento.

Nel 1986, GML diventa lo standard *ISO*[16](ISO 8879) per registrare e scambiare i dati gestionali ed è noto come SGML (Standard Generalized Markup Language).

L'elaborazione dei documenti SGML era definita da un altro standard internazionale: *Document Style Semantics and Specification Language* (*DSSS*L).

Nel 1992, fu realizzato il linguaggio *HyTime (Hypermedia/Time-based Structuring Languag*e), che aveva funzioni specifiche per rappresentare gli oggetti SGML multimediali e potenti capacità di "link"ing.

[16] ISO =*International Organization for Standard*s è l'organizzazione internazionale per gli standard

Appendice B: il W3C

Nel 1994 Tim Berners-Lee fondò l'organizzazione **W3C**[32] (World Wide Web Consortium) per sviluppare tecnologie destinate al World Wide Web: si tratta di un consorzio internazionale di imprese, neutrale rispetto ai venditori, che collabora con organizzazioni di vario tipo e sviluppa tecnologie (specifiche, linee guida, software, e strumenti), definendo protocolli comuni[17].

Il W3C si occupa anche di indicare gli obiettivi a lungo termine dell'evoluzione del WWW, ed in particolare il suo scopo è **portare il Web al suo massimo potenziale**, mediante lo sviluppo di tecnologie.

Gli obiettivi e i principi strategici del W3C possono essere sintetizzati in tre punti:

1. **"Accesso Universale"**: Rendere il Web accessibile in tutti i continenti a tutti, promuovendo tecnologie che tengano conto delle notevoli differenze in termini di cultura, formazione, capacità, risorse materiali e disabilità fisiche degli utenti
2. **Semantic Web**: Sviluppare un ambiente software che consenta ad ogni utente di fare il miglior uso possibile delle risorse disponibili sul Web
3. **"Web della Fiducia"**: Guidare lo sviluppo del Web tenendo in attenta considerazione tutti gli aspetti innovativi che la tecnologia solleva in campo legale, commerciale e sociale

Altra attività del W3C è quella della standardizzazione: una tecnologia Web standardizzata dal W3C si chiama *"raccomandazione"* e sono i documenti che specificano il ruolo, la sintassi, le regole e altre caratteristiche di una tecnologia.

Fra le attuali "raccomandazione" del W3C figurano *HTM*L (*HyperText Markup Language*) e XML (*Extensible Markup Language*).

Prima di diventare *"raccomandazione"*, un documento deve superare tre fasi: *working draf*t, una bozza del documento in evoluzione che può subire delle modifiche; "raccomandazione candidata", una versione stabile del documento che può essere implementata dall'industria del software; "raccomandazione proposta", una specifica considerata matura (perché è stata implementata e collaudata per un certo periodo) e pronta per essere convertita in *"raccomandazione"* del W3C.

[17] Il W3C è costituito da tre *ospiti :* MIT (Massachusetts Institute of Technology), INRIA (Institut National de Recherche en Informatique et Automatique) e Keio University of Japan e da oltre 400 *membri*.