XML

Mario Bravetti

Introduzione

- XML (eXtensible Markup Language) è un un linguaggio di markup che lascia completa libertà nell'utilizzo di tag (aggiungibili a piacere): nei loro nomi e nel loro significato.
- XML è stato standardizzato dal W3C nel 1998 e attualmente sono disponibili le versioni 1.0 e 1.1 (la 1.1 non è ancora largamente supportata):
  - http://www.w3.org/TR/REC-xml;

  traduzione ufficiale in italiano:
  - http://www.iat.cnr.it/xml/REC-xml-19980210-it.html

XML vs. HTML

- HTML attribuisce un significato predefinito ad un insieme predeterminato di tag (es. p,b,i, ecc..) e anche storicamente modalità di visualizzazione.
- XML invece è orientato esclusivamente a definire la struttura e il significato dei dati (invece che trattare formati di visualizzazione):
  - i tag indicano i contenuti che lo compongono, non definiscono il modo con cui questi devono apparire
  - per definire caratteristiche di visualizzazione si deve ricorrere ad altre tecnologie, es. fogli di stile: i CSS o fogli XSLT (trasformazione di documenti XML)

Rappresentazione dei dati

- XML consente rappresentazione dati di qualsiasi tipo in formato strutturato ai fini di archiviazione (come un database) o trasmissione (es. sulla rete).
  - qualsiasi database rappresentabile con tags, anche se non sempre possibile riprodurre tutti i vincoli, es.:

    <persona>
    <nome> … </nome>
    <cognome> … </cognome>
    <indirizzo> … </indirizzo>
    </persona>
  - documenti XML sono semplici file di testo: dati al loro interno non legati a formato proprietario, ma manipolabili da ogni piattaforma (interoperabilità).

Esempio di tipo di dati (con DTD)

```
<! ELEMENT persona (nome, cognome, indirizzo)>
<! ELEMENT nome (#PCDATA)>
<! ELEMENT cognome (#PCDATA)>
<! ELEMENT indirizzo (#PCDATA)>
```

- XML valido rispetto a questo tipo di dato:

```
<persona>
        <nome>Silvia</nome>
        <cognome>Rossi</cognome>
        < indirizzo >Via Sacchi</ indirizzo >
</persona>
```

Definizione di nuovi linguaggi

- Tramite tecnologie per la definizione di tipo (insieme di tag ammissibili e regole su loro utilizzo),
  - esempi sono DTD (visto per HTML) o XML-Schema

  XML consente: sviluppo di linguaggi di markup specifici per particolari domini applicativi, es.:
  - XHTML (eXtensible HTML) con DTD analoghe a quelle viste per HTML
  - CML (Chemical Markup Language)
  - MathML (Mathematical Markup Language)
  - SMIL (Synchronized Multimedia Integration Language)
  - VML (Vector Markup Language)
  - MusicML (Music Markup Language)
  - OFX (Open Financial eXchange)
  - RDF (Resource Description Framework).

In definitiva..

- Deriva da SGML ma rispetto ad esso adotta una sintassi semplificata che lo rende utilizzabile in maniera efficiente in ambito Web.
- Specifiche XML definiscono anche: serie di tecnologie e linguaggi correlati per descrizione dei vari aspetti connessi con rappresentazione dei dati
  - oltre a tipi di documento e fogli di stile: meccanismi di ricerca (XPath), link (XLink e Xpointer), ecc.
- XML si propone di integrare, arricchire HTML come linguaggio di markup standard per il World Wide Web (si proponeva anche di sostituirlo!).

XML e i browser

- Browser XML-enabled: Internet Explorer 5 e superiori, Firefox, e tutti i browser moderni.
- Siccome in XML non esiste un insieme precostituito di tag ed un significato ad essi associato
  - in assenza di fogli di stile, visualizzano il documento XML mostrando il sorgente testuale
  - con fogli di stile, es. CSS, invece:
    - si può ottenere un risultato analogo all'uso di HTML + CSS in cui non si sfruttano i tag presentazionali di HTML (es. HTML con soli tag DIV e SPAN)
    - Tramite fogli di stile XSLT si possono definire vere e proprie "viste parziali" sui dati e loro trasformazioni in HTML

Parsing dei documenti XML

- XML più che altro utilizzato per scambio dati tra applicativi (es. servizi web) e per archiviazione.
- Applicativi di un dato linguaggio utilizzano parser XML per trasformare documenti XML in/da strutture dati del linguaggio. Operano controllando:
  - in generale, la correttezza del documento XML
  - la conformità rispetto ad un eventule foglio di stile, es. DTD, dichiarato nel documento
- Parser solitamente forniti come librerie (API) di un dato linguaggio: solitamente linguaggi ad oggetti che consentono la rappresentazione DOM.

Struttura di un documento XML

- Sono semplici file di testo (con estensione .xml)
- Documento XML deve aver la seguente struttura:
  - prologo: sezione iniziale che contiene le istruzioni di elaborazione, la dichiarazione del tipo di documento
  - corpo: sezione che contiene la struttura di markup e le informazioni che si desidera rappresentare.

Esempio di documento XML

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

Prologo

```
<!-- Esempio di documento XML -->
<persona>
   <nome> Andrea </nome>
   <cognome> Rossi </cognome>
   <data_nasc> 01.07.1975 </data_nasc>
</persona>
```

Corpo del documento

Istruzioni di elaborazione

- Istruzioni di elaborazione (PI, process instruction) rappresentano attività da eseguire sul corpo.
- Quelle maggiormente utilizzate sono le seguenti:
  - la dichiarazione XML (deve essere nella prima riga): specifica proprietà del documento (versione, codifica dei caratteri, modalità relativa a tipo di documento)

    <?xml version="1.0" encoding="UTF-8" standalone="no"?>
  - le dichiarazioni dei fogli di stile da applicare al documento (definizioni applicate per ultime hanno la precedenza rispetto a quelle applicate per prime).

    <?xml-stylesheet href="style.css" type="text/css" ?>

DTD interne

- Dichiarate prima del corpo (nel prologo) con:

  <!DOCTYPE root_name [

  ... regole di definizione

  ]>

- root_name: nome elemento radice del corpo
- Se non vi sono anche riferimenti a DTD esterne dichiarazione XML deve avere standalone="yes"

DTD esterne

- Le DTD esterne possono essere:
  - pubbliche (keyword "PUBLIC"): sono DTD standardizzate disponibili pubblicamente sul Web, tipicamente create da enti di standardizzazione
    - come quelle per HTML e XHTML
  - private (keyword "SYSTEM"): sono le DTD create dall'autore/sviluppatore del sito/sistema
    - non sono disponibili pubblicamente e non rappresentano alcun particolare standard

DTD esterne: uso

- Dichiarate prima del corpo (nel prologo) con:
  - DTD pubblica

    <!DOCTYPE root_name PUBLIC ident url>
  - DTD non pubblica

    <!DOCTYPE root_name SYSTEM url>
- Dichiarazione XML deve avere standalone="no"
- Eventualmente prima di ">" vi può essere una parte di regole di definizione interne tra "[" e "]"

DTD esterne pubbliche: uso

- Caratterizzate da:
  - un identificativo univoco (formal public identifier, FPI) che specifica in forma di URN (Universal Resource Name) informazioni su origine della DTD;
  - un URL che consente di localizzare la DTD specificando la locazione del file ".dtd".
- Esempio (XHTML transitional):

```
<!DOCTYPE html PUBLIC
"-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

FPI

URL

Documenti XML Well-formed

- Documenti XML devono essere conformi a una serie di regole prestabilite (di “composizione)
  - indipendentemente dal fatto di essere validi o meno rispetto ad una eventuale DTD
- Un documento che rispetta queste regole si dice "ben formato" (well-formed)
  - se un documento non è ben formato, qualsiasi tentativo di leggerlo o visualizzarlo sarà destinato al fallimento, in quanto i parser si basano su queste regole per effettuare il riconoscimento del testo.
- Anche i browser danno errori a differenza di html

Regole di composizione

- Affinché un documento XML sia ben formato, deve rispettare queste regole:
  - il documento deve iniziare con la dichiarazione XML
  - nomi dei tag sono case sensitive
  - gli elementi non vuoti devono avere sia il tag di apertura che il tag di chiusura
  - gli elementi vuoti che usano un solo tag devono terminare con />
  - il documento deve sempre avere un unico elemento che contiene tutti gli altri (elemento root o elemento documento)

Regole di composizione

– gli elementi possono essere annidati ma non sovrapposti (overlapping):
  - non si possono sovrapporre i tag di apertura e chiusura dei nodi di pari livello
  - non si possono sovrapporre i tag di apertura e chiusura dei nodi figli con quelli del nodo padre

  Cioè i documenti XML hanno una struttura ad albero definita tramite i tag senza errori.

– i valori di attributo devono essere inseriti tra virgolette (semplici o doppie)

Regole di composizione

– nei contenuti non possono essere usati i caratteri "<" e "&", che sono riservati (per iniziare rispettivamente tag e riferimenti a entità).

– gli unici riferimenti a entità ammesi sono

  • < (<) & (&) > (>) ' (‘) " (“)

  per altri (come ©, ®, ecc.) si usano codici unicode inseriti tramite numeric character reference (NCR)

  • Esempio: Σ inserito con Σ o Σ (esadecimale)

– i commenti e le istruzioni di elaborazione non possono comparire all’interno dei marcatori

– dentro un elemento non devono comparire multipli attributi con stesso nome

Contenuti di un documento XML

- A differenza di HTML in XML il contenuto degli elementi è sempre soggetto a parsing
  - in HTML ciò non accade per alcuni elementi (es. script, style,…)
  - ma XML: elementi non hanno semantica predefinita!
- Se nei contenuti del documento si devono usare simboli confondibili con marcatura:
  - inserire il corrispondente riferimento a entità
  - inserire il testo all'interno di una sezione "protetta" (CDATA)

Sezioni CDATA

- Dette anche "sezioni di dati di tipo carattere": consentono di specificare aree di un documento XML che contengono un testo che non deve essere elaborato dal parser XML
- In queste regioni è possibile scrivere testo composto da simboli di qualsiasi tipo (marcatori HTML, simboli riservati, istruzioni di linguaggi di programmazione, ecc.)

Sezioni CDATA

- Le sezioni di tipo carattere sono delimitate da:
  - un identificativo di inizio: <![CDATA[
  - un identificativo di fine:]]>
- Le sezioni CDATA non possono essere innestate una dentro l'altra (ma ve ne può essere più d'una)

Esempio sezione CDATA

```
<![CDATA[
    // ciclo
    if (!error & (nextistruction<=NUMCELL)
        fetch(..);
        decode(..);
        execute(..);
]]>
```

- Questa sezione viene visualizzata con gli stessi spazi, tabulazioni, cr e lf con cui è scritta.