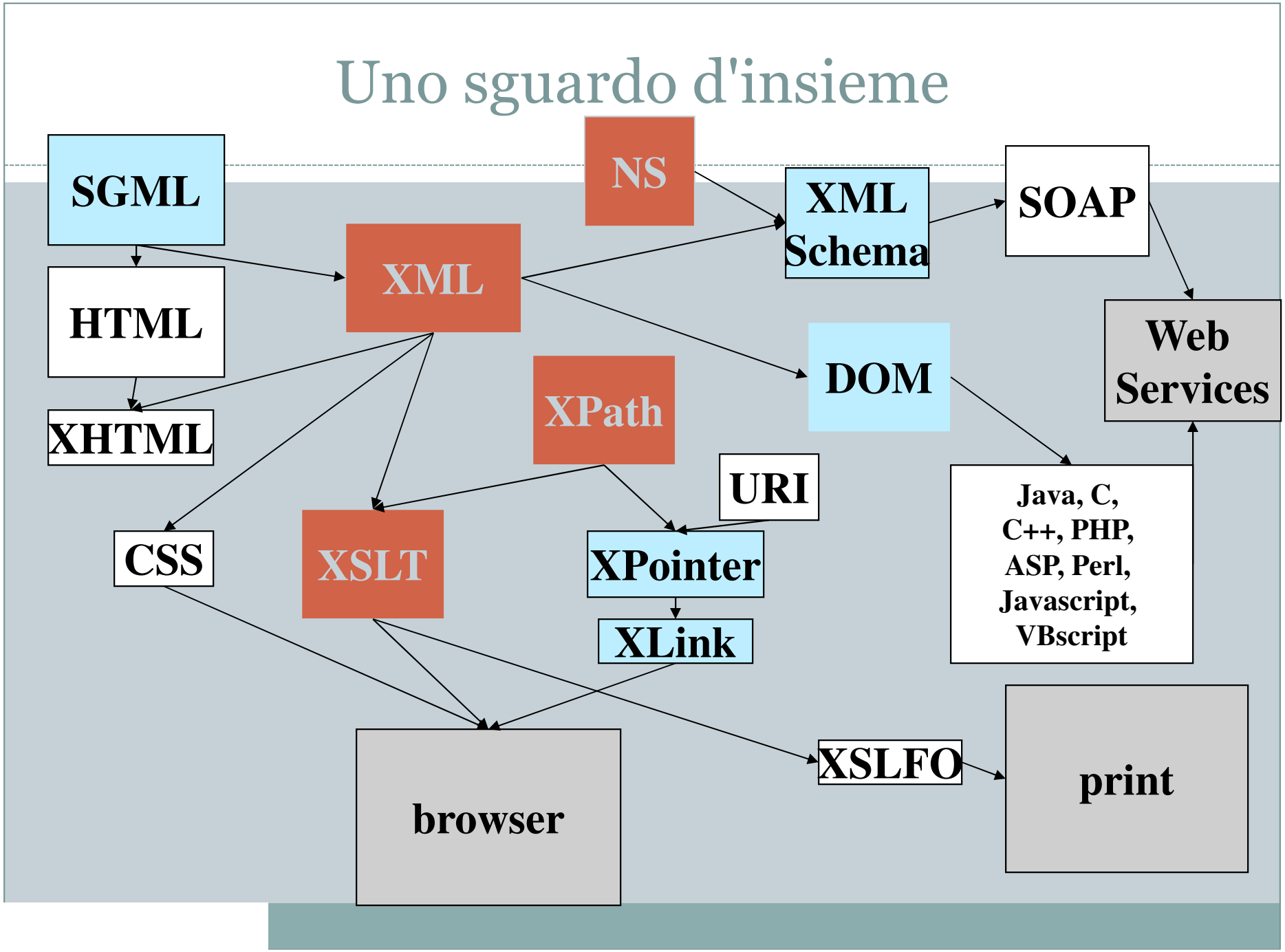
Introduzione ad XML

GIORDANI ALESSANDRA
ITT SERALE
"G.MARCONI"

XML

- XML (Extensible Markup Language) è un insieme **standard** di regole sintattiche per modellare la **struttura** di documenti e dati, progettato per lo scambio e la interusabilità di documenti su Internet
- È UNA derivazione del noto linguaggio SGML (Standard Generalized Markup Language) ma prevede una sintassi semplificata e una definizione più lunga di linguaggi associati
- XML si propone di integrare, arricchire e, nel lungo periodo, sostituire HTML come linguaggio di markup standard per il World Wide Web.

Uno sguardo d'insieme



Perché XML?

- HTML nacque come un DTD di SGML, che permetteva di mettere in rete documenti di un tipo molto specifico, semplici documenti di testo con qualche immagine e dei link ipertestuali.
- Con il successo del WWW, HTML venne iniziato ad usare per molti scopi, molti più di quelli per cui era stato progettato.
- Si iniziò ad abusare dei tag di HTML per gli effetti grafici che forniva, più che per gli aspetti strutturali o semantici.
- Si iniziarono a desiderare elaborazioni sofisticate sui dati HTML, elaborazioni che non era possibile fornire.

I vantaggi di XML (1)

- Documenti auto-descrittivi
  - La scelta dei nomi degli elementi può essere fatta per facilitare la comprensione del ruolo strutturale dell'elemento.
  - Inoltre, l'uso di un DTD può esplicitare le regole di composizione ed i rapporti possibili tra le varie parti dei documenti.
- Struttura navigabile dei documenti
  - La rigida struttura ad albero rende semplice la visualizzazione e l'analisi della struttura del documento, e la possibilità di visualizzare il documento è indipendente dal foglio di stile che vi si applica.

I vantaggi di XML (2)

- Platform-independence
  - XML è uno standard aperto, e chiunque può realizzare strumenti che lo usino come formato di dati.
- Facile convertibilità a formati Web
  - La totale interdipendenza tra XML, SGML, HTML etc. fa sì che la conversione tra formati interni e formati per il Web sia facile.
- Strutturazione gerarchica dei documenti
  - Esistono molti formati di dati generici per l'intescambio di dati, ma sono tutti organizzati linearmente. XML permette strutture ad albero.

I vantaggi di XML (3)

- Ripetibilità degli elementi
  - XML permette di definire formalmente elementi ripetibili. Questo permette strutture più flessibili e complesse di altri formati di dati
- Content model misti
  - XML trova un punto di equilibrio tra i formati dati per l'interscambio di dati e i formati per la strutturazione di documenti di testo. I content model misti (elementi che possono contenere sia altri elementi che testo) infatti permettono di inserire caratterizzazioni semantiche non solo per interi elementi, ma anche all'interno di elementi di testo contenitori (ad esempio, i paragrafi).

I vantaggi di XML (4)

- Meta-linguaggio
  - XML non è una grammatica (cioè un vocabolario di termini riservati e regole di utilizzo), ma una *grammatica di grammatiche*, un modo per generare grammatiche personalizzate sulle esigenze del progettista
- Sintassi universale, minimale e rigorosa
  - XML definisce una sintassi di scrittura (per dati, metadati, categorie, etc.) utile per qualunque applicazione, vocabolario, linguaggio umano, sistema operativo. La totale mancanza di eccezioni e la equivalenza tra struttura dati e rappresentazione linearizzata favoriscono l'adozione in ambienti estremamente eterogenei.

Quali applicazioni XML?

- Data Interchange
  - Ogni volta che più programmi si debbono scambiare dati, ci sono problemi di compatibilità. Ogni programma ha le proprie assunzioni in termini di caratteri, separatori, ripetibilità di elementi, differenza tra elementi vuoti e assenti, ecc.
  - XML si propone come la sintassi intermedia più semplice per esprimere dati anche complessi in forma indipendente dall'applicazione che li ha creati.
- Document publishing
  - XML è ideale come linguaggio per esprimere documenti strutturati o semi strutturati, e per esprimerli in maniera indipendente dalla loro destinazione finale.
  - Lo stesso documento XML può essere preso e trasformato per la stampa, il Web, il telefonino, l'autoradio.

Documenti ben formati o validi

- XML distingue due tipi di documenti rilevanti per le applicazioni XML: i documenti ***ben formati*** ed i documenti ***validi***.
- In SGML, un DTD è necessario per la validazione del documento. Anche in XML, un documento è **valido** se presenta un DTD ed è possibile validarlo usando il DTD.
- Tuttavia XML permette anche documenti **ben formati**, ovvero documenti che, pur essendo privi di DTD, presentano una struttura sufficientemente regolare e comprensibile da poter essere controllata.

Documenti XML ben formati

- Un documento XML si dice ben formato se:
  - Tutti i tag di apertura e chiusura corrispondono e sono ben annidati
  - Esiste un elemento radice che contiene tutti gli altri
  - I tag vuoti (senza contenuto) utilizzano un simbolo speciale di fine tag: <vuoto/>
  - Tutti gli attributi sono sempre racchiusi tra virgolette
  - Tutte le entità sono definite.

Parser validanti e non validanti

- Il cuore di un applicazione XML è il parser, ovvero quel modulo che legge il documento XML e ne crea una rappresentazione interna utile per successive elaborazioni (come la visualizzazione).
- Un parser validante, in presenza di un DTD, è in grado di verificare la validità del documento, o di segnalare gli errori di markup presenti.
- Un parser non validante invece, anche in presenza di un DTD è solo in grado di verificare la buona forma del documento.
- Un parser non validante è molto più semplice e veloce da scrivere, ma è in grado di fare meno controlli. In alcune applicazioni, però, non è necessario validare i documenti, solo verificare la loro buona forma.

Sintassi dei DTD

- Una precisazione
- <!DOCTYPE … >
- <!ELEMENT … >
- <!ATTLIST … >
- <!ENTITY … >: Entità generali
- <!ENTITY % … >: Entità parametriche
- Altre caratteristiche di XML

La dichiarazione di tipo

- Il <!DOCTYPE … > è la dichiarazione del tipo di documento. Essa permette alle applicazioni SGML di determinare le regole sintattiche da applicare alla verifica e validazione del documento.
- La dichiarazione non è, ma contiene o fa riferimento alla Document Type Definition, o DTD, ove vengono elencati gli elementi validi e i loro vincoli.
- Il DTD può essere posto in un file esterno, internamente al documento, o in parte esternamente ed in parte internamente.

- **N.B.: In XML il nome del DOCTYPE deve essere il nome del tag radice.**

Dichiarazione del DTD: <!DOCTYPE … >

- **1** **<!DOCTYPE mydoc SYSTEM "document.dtd">**
- **2** **<!DOCTYPE mydoc [**
  <!ELEMENT …]>
- **3** **<!DOCTYPE mydoc SYSTEM "document.dtd" [**
  <!ELEMENT …]>

- La prima forma di dichiarazione indica che il DTD è contenuto in un file esterno (per esempio, condivisa con altri documenti). Il DTD viene chiamato *external subset* perché è posto in un file esterno.
- La seconda forma precisa il DTD internamente (cioè nello stesso file), che quindi non può essere condiviso da altri file. Il DTD si chiama in questo caso *internal subset*.
- La terza forma precisa una parte del DTD come contenuta in un file esterno (e quindi condivisibile con altri documenti), e una parte come propria del documento, e non condivisibile.

Specifica di elementi: <!ELEMENT …>

- **<!ELEMENT nome content-model >**
- **<!ELEMENT para (#PCDATA | bold)* >**

- In SGML:
- <!ELEMENT nome ST ET content-model >
- <!ELEMENT para - - (#PCDATA | bold)* >

- ST & ET: minimizzazione del tag iniziale (ST) e finale (ET): può assumere i valori '-' (*obbligatorio*) o 'o' (*omissibile*). In XML mancano.
- Content-model: la specificazione formale del contenuto permesso nell'elemento, secondo una sintassi specifica di gruppi di modelli.

Content model

- Tramite il content model posso specificare quali sono gli elementi leciti all'interno di un elemento, in quale numero e quale posizione rispetto agli altri.
- Un content model (CM) è 'ANY', 'EMPTY', oppure un gruppo di CM più elementari.
- Un gruppo di CM è sempre circondato da parentesi. Può contenere la specifica #PCDATA, la specifica di un elemento SGML, o di un altro gruppo di CM più elementare. Ogni specifica è separata da un separatore. Alla fine ci può essere un operatore di ripetizione.

ANY, EMPTY, #PCDATA

- ANY: significa che qualunque content è ammesso. Ogni elemento definito nel DTD può comparire qui dentro in qualunque ordine e numero.
- EMPTY: Questo è un elemento vuoto, o senza contenuto. In questo caso nel documento esso appare come tag semplice, senza tag finale.

  Def.: <!ELEMENT HR EMPTY>
  Uso: "<HR/>"
- #PCDATA: (Parsed Character Data): il contenuto testuale del documento. Include caratteri ed entità generali. È naturalmente in numero multiplo.

Separatori

- *S*eparano specifiche determinando l'ordine o l'obbligatorietà:
  - **',' (virgola)**: richiede la presenza di entrambe le specifiche nell'ordine precisato.
    Es.: **(a , b)**: ci devono essere sia a che b, e prima ci deve essere a e poi b.
  - **'|' (barra verticale)**: ammette la presenza di una sola delle due specifiche.
    Es.: **(a | b)**: ci può essere o a, oppure b, ma solo uno di essi.
- In SGML anche:
  - **'&' (ampersand)**: richiede la presenza di entrambe le specifiche, ma in qualunque ordine.
    Es.: **(a & b)**: ci debbono essere sia a che b, ma in qualunque ordine.

Operatori di ripetizione (1)

- Permettono di specificare se un gruppo può comparire esattamente una volta, almeno una volta, oppure zero o più volte.
  - ***Niente***: la specifica precedente deve comparire esattamente una volta.
    Es.: c, (a, b): a e b devono comparire in quest'ordine esattamente una volta. È lecito solo: cab. Si dice che è una specifica *richiesta* e *non ripetibile*.
  - ***? (punto interrogativo)***: la specifica precedente può e può non comparire, ma solo una volta.
    Es.: c, (a , b)?: a e b possono comparire una volta, ma possono non comparire. Sono lecite: c, cab. Si dice che è una specifica *facoltativa* e *non ripetibile*.

Operatori di ripetizione (2)

- + ***(più)***: la specifica precedente deve comparire almeno una volta. Es.: c, (a , b)+: a e b devono comparire almeno una volta, ma possono comparire anche più di una. Sono lecite: cab, cabab, cababababab, ma non c, ca, cb, cba, cababa. Si dice che è una specifica *richiesta* e *ripetibile*.
- * ***(asterisco)***: la specifica precedente deve comparire zero o più volte. Es.: c, (a, b)*: a e b possono comparire o no, a scelta e in numero libero. Sono lecite: c, cab, cabab, cababababab, ma non ca, cb, cba, cababa. Si dice che è una specifica *facoltativa* e *ripetibile*.

Character content

- **<!ELEMENT para (#PCDATA) >**
- Un elemento contiene soltanto caratteri stampabili e entità. Nessun altro elemento è ammesso all'interno.
- <para>Questo è lecito</para>

Lista di attributi: <!ATTLIST … >

- La dichiarazione <!ATTLIST… > permette di definire una lista di attributi lecita ad un elemento SGML dichiarato in precedenza.

- <!ATTLIST nome

  nome-attributo-1 tipo-1 default-1
  nome-attributo-2 tipo-2 default-2
  nome-attributo-3 tipo-3 default-3
  …

  >

Attributo di tipo stringa

- <!ATTLIST doc
  linguaggio CDATA “HTML” >

- CDATA significa “character data”, e indica qualunque sequenza di caratteri (tranne “<“ e le virgolette già usate come delimitatore), ma non entità o elementi.

Attributi di tipo lista

- **<!ATTLIST doc**
  stato (bozza | impaginato | finale) “bozza”
  >
- Solo uno dei valori elencati nella lista può essere accettato. Ogni altro valore genererà un errore.