Potenziali applicazioni del linguaggio XML

Cos'è un linguaggio XML e come funziona?

- Acronimo di Extensible Markup Language, è un *linguaggio a marcatori estensibile*, in grado di realizzare una descrizione di un contenuto, un elemento caratteristico di un documento, in forma testuale: *es. la sottotitolatura è una descrizione XML della componente audio di un video*.
- Fu introdotto e sviluppato dal **World Wide Web Consortium (W3C)** nel 1996: si decise di avere un linguaggio di mark-up in grado di descrivere delle risorse in rete che fosse sia estensibile che di uso generale.

Caratteristiche dell'XML (1)

- Ha una **sintassi rigorosa**, ma **flessibile**: ha regole di base molto stringenti, ma permette, al tempo stesso, di cambiarle laddove c'è necessità.
- Deriva dal **SGML**, un *linguaggio generalizzato di mark-up*, creato nel 1986 dal W3C quale primo esempio di linguaggio in grado di realizzare descrizioni di contenuti, ma era **troppo generico** → il che comportava una serie di caratteristiche troppo pesanti.

Caratteristiche dell'XML (2)

- E' un linguaggio a **marcatori** o **tag** → è in grado di descrivere un elemento caratteristico di un documento.
- I marcatori sono inseriti all'interno del documento.
- Ogni **marcatore** (coppia di marcatori di inizio e fine) **identifica** un **elemento** (o componente) del documento: racchiudono il contenuto del documento.
- Sia il testo, sia i marcatori sono memorizzati in **formato ASCII** (American Standard Code for Infromation Interchange).
- E' un linguaggio che utilizza la **codifica dei caratteri Unicode**.
- Un documento XML è **leggibile** da un utente umano **senza la mediazione** di software specifici.

Esempio di marcatori

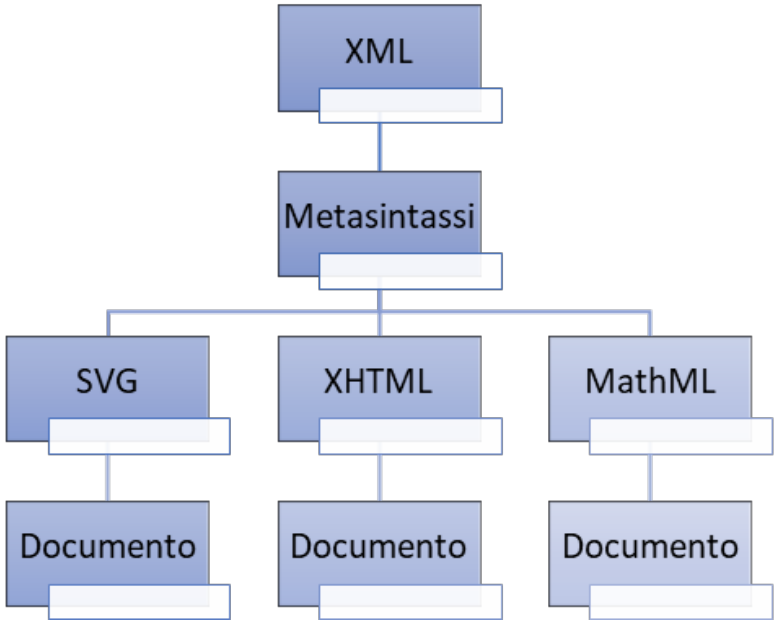
- <Gatto> → è il **marcatore di inizio** che definisce l'elemento «Gatto»
- <Specie> → ogni marcatore definisce un elemento (nel ns. caso l'elemento è «Specie»)
- Soriano → testo del documento
- </Specie> → i marcatori si aprono e si chiudono sempre in ordine inverso: l'ultimo che si è aperto è il primo a chiudersi. Questo approccio è chiamato ***LIFO***: ***Last Input First Output***.
- </Gatto> → **marcatore di fine**

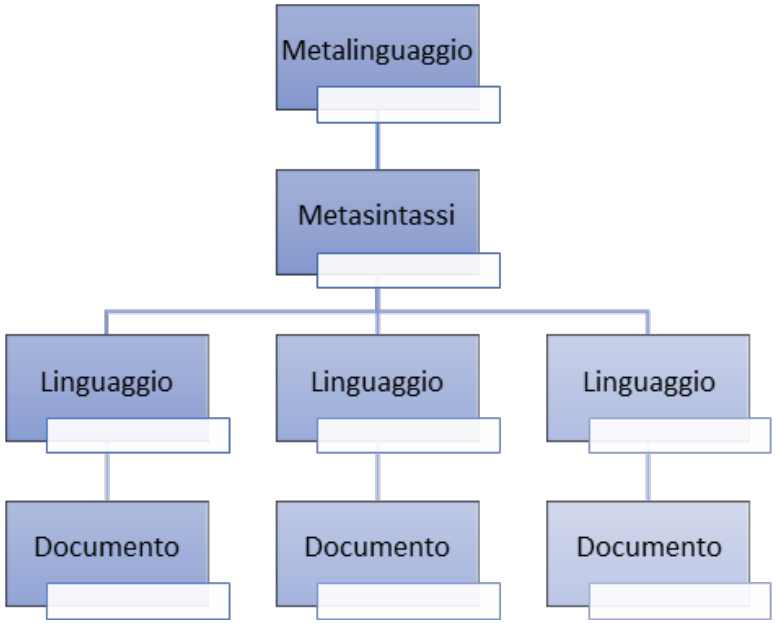
Esempio di Elemento

- Un **elemento** è un **frammento di testo** racchiuso fra uno *start tag* e un *end tag.*
  - Uno marcatore iniziale (start tag) è costituito da un **nome** più eventuali **attributi** racchiusi dai simboli ‘<’, ‘>’ *<TagName attribute-list>*
    - *Es. <Gatto Microchip = "s000 1234AX" Genere ="maschio" Pelo ="lungo" Vaccino ="sì">*
  - Un marcatore finale (end tag) è costituito da un nome (lo stesso dello start tag) racchiuso da ‘</’,’>’: *</TagName>*
    - *Es. </Gatto>*
- Il **contenuto** di un elemento può essere un **valore atomico** (elemento semplice) o un **valore strutturato** (elemento complesso) attraverso altri elementi XML (**elementi figli**):
  - *Es: <Gatto> contiene gli elementi: <Nome>, <Età>, <Padrone>, <Specie>*
- Un elemento può essere **vuoto**: privo di contenuto
  - *Es. <Gatto _ />* → equivale a scrivere: *<Gatto> </Gatto>*

XML come *metalinguaggio*

- È un linguaggio che ha bisogno di ulteriori dettagli per essere meglio definito.
- Ha una serie di **regole di base** di carattere generale, dette **metasintattiche**, ma necessita di **ulteriori dettagli** per poter diventare un **linguaggio** vero e proprio, con **regole sintattiche (come scrivere le informazioni all'interno del documento)** proprie, ereditate dal linguaggio XML di base, e una **semantica (cosa possiamo scrivere)**.





N.B. ogni Linguaggio ha una propria sintassi (insieme definito di regole che derivano dal linguaggio XML di base) che sarà rispettata dai differenti Documenti che verranno prodotti, i quali, a loro volta, rispetteranno la metasintassi del metalinguaggio di partenza (in tal caso XML).

- Un documento XML che rispetta le regole sintattiche si dice *well-formed* (ben formato).
- Un documento XML che rispetta le regole sintattiche ed eventuali regole semantiche si dice *valido*.
- Ciò **implica** che un documento ben formato può non essere valido, ma un documento valido è necessariamente ben formato.
- Un documento XML è un semplice **file di testo** che si salva con un'estensione ***.xml***
- XML è **case-sensitive**, cioè è sensibile alla differenza tra maiuscole e minuscole

Struttura logica di un documento XML

- Un documento XML:
- è strutturato in **modo gerarchico;**
- è composto da **Elementi;**
  - Un elemento:
  - rappresenta la componente logica del documento
  - può contenere un **frammento di testo**, oppure altri **elementi** chiamati ***sotto-elementi***, ossia tag contenuti all'interno di un altro tag.
  - ad un elemento possono essere associati degli **attributi** → informazioni descrittive rappresentate sottoforma di coppia *nome-valore* all'interno del tag.
  - gli elementi sono organizzati ad albero con **radice** → *root*

  N.B. in un documento well formed l'elemento radice è sempre unico!
- Ogni documento XML può essere rappresentato come un albero → ***document-tree***

- Un documento è costituito da due parti:
- **Prologo**: contiene una **dichiarazione XML** (*XML Declaration* con la quale dichiariamo quale versione del metalinguaggio XML usiamo; es. *xml version = "1.0"*) ed il riferimento (opzionale) ad altri documenti che ne definiscono la struttura o direttive di elaborazione (es. *encoding = "UTF-8",* che indica con quale tipologia di testo e di caratteri scriviamo il documento).
- **Corpo**: è il documento XML vero e proprio.

DTD: Document Type Definition

- È un linguaggio per determinare in modo non ambiguo la struttura dei dati che vengono rappresentati, prima di costruire il file XML.
- Fornisce uno strumento per la validazione dei documenti XML e può essere inserito direttamente all'interno del file XML, oppure può essere salvato in un file con estensione ***.dtd***, poi collegato al file XML.

Per applicare un DTD ad un documento XML, nel Prologo del documento XML dobbiamo inserire una dichiarazione (***Doctype Declaration***) con questa sintassi:

<!DOCTYPE root-element SYSTEM "filename">

- Dove:
  - **root-element** è il nome dell'elemento radice ;
  - **SYSTEM** definisce documenti di utilizzo locale ;
  - **filename** è il file che contiene il DTD.
- In alternativa a SYSTEM si può usare la parola chiave PUBLIC che serve per definire documenti di utilizzo pubblico.
- Es di Prologo completo:

<?xml version="1.0" encoding ="UTF –8"?>

<!DOCTYPE filename SYSTEM "filename.dtd">

XML Declaration

Doctype Declaration

Esempio di documento XML

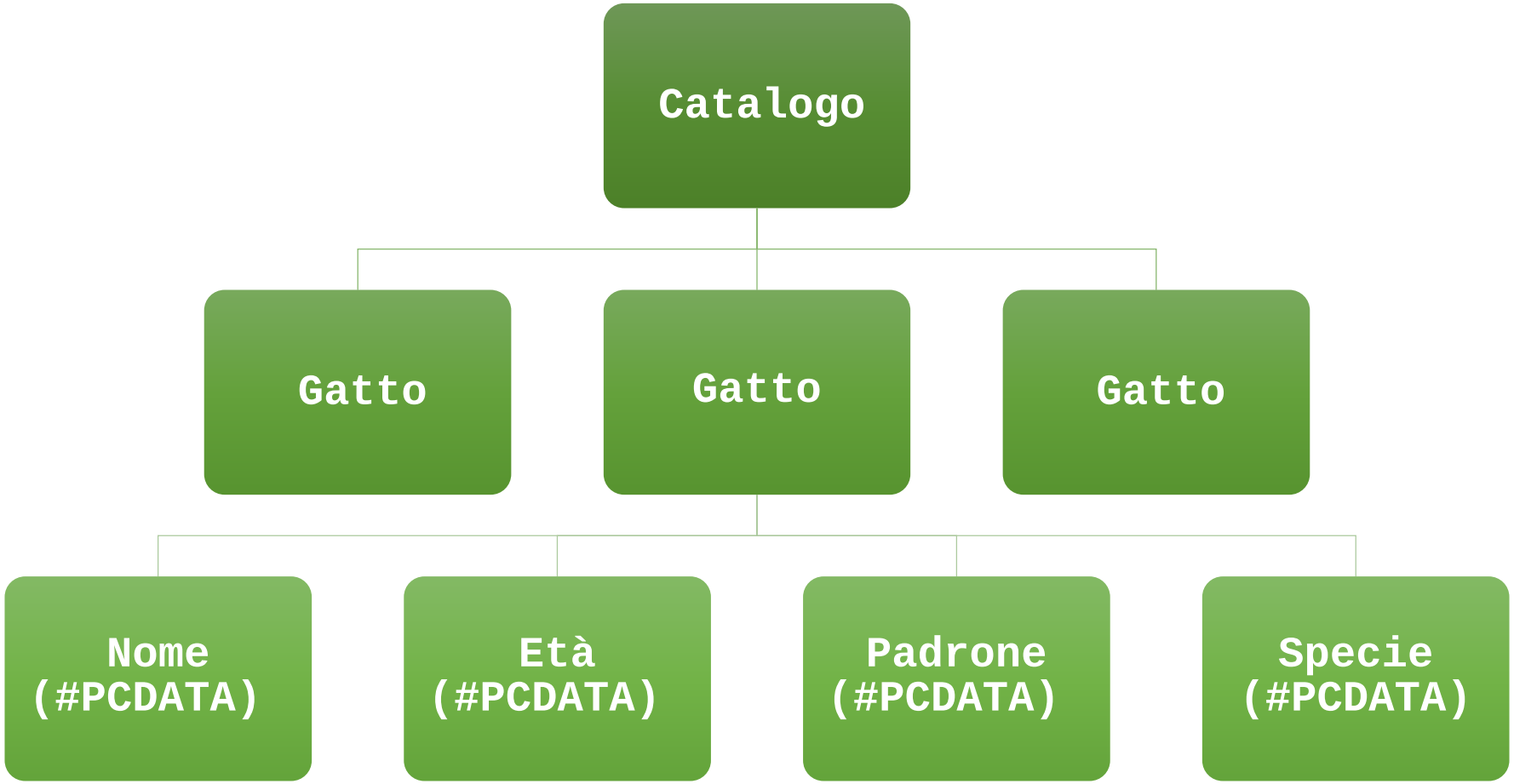
Modellizzare un documento XML di catalogazione Gatto e relativo DTD di validazione in cui:

- Un Catalogo può contenere zero o più Gatto.
- Un Gatto è descritto eventualmente da un Nome, Età, zero o più Padrone e una Specie.
- Un Gatto è dotato di proprietà quali un codice identificativo univoco (Microchip), un'indicazione di Genere (maschio| femmina) obbligatoria, un’indicazione di Pelo (lungo| medio |corto) se richiesta, un’indicazione di Vaccino (obbligatoria) che può essere sì – no.

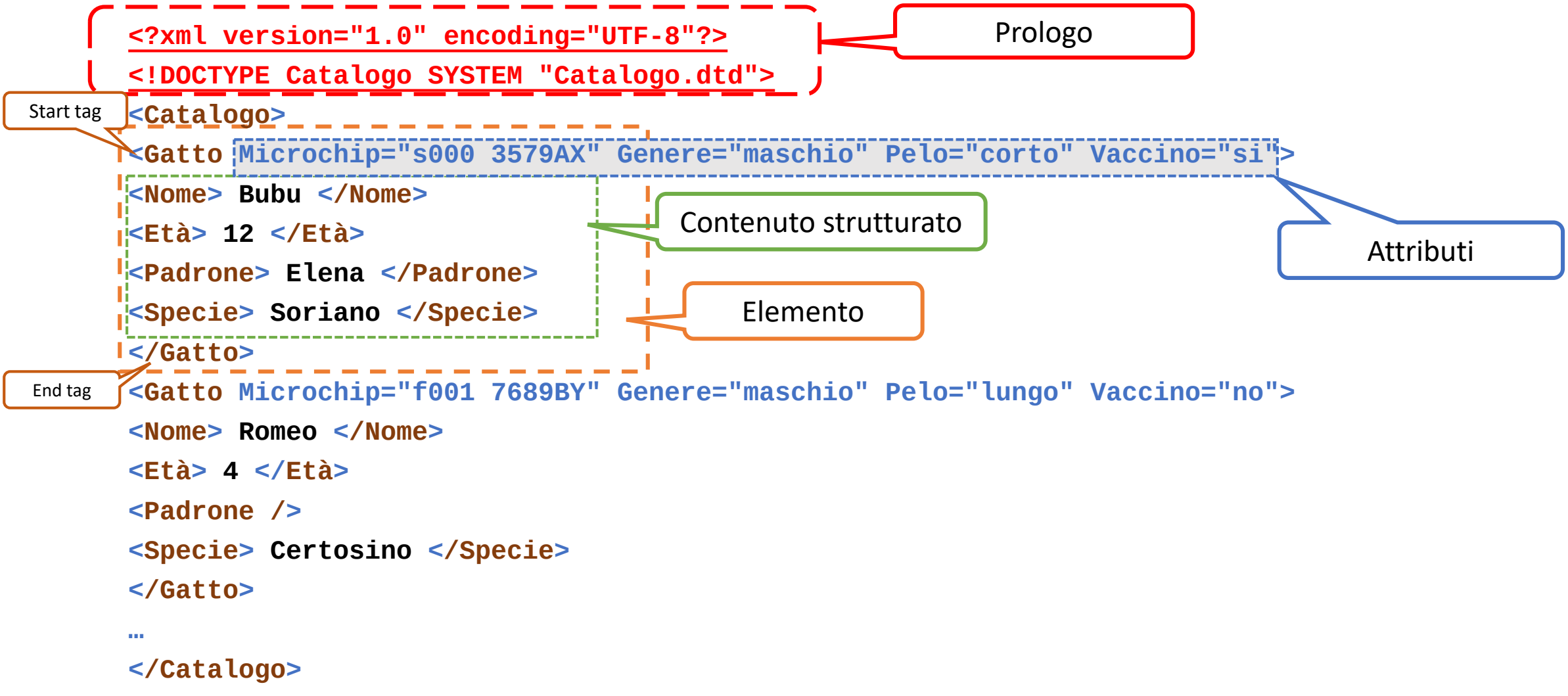
DTD di validazione

```
<!DOCTYPE Catalogo[
<!ELEMENT Catalogo (Gatto*)>
<!ELEMENT Gatto
(Nome, Età, Padrone+, Specie)>
<!ELEMENT Nome (#PCDATA)>
<!ELEMENT Età (#PCDATA)>
<!ELEMENT Padrone (#PCDATA)>
<!ELEMENT Specie (#PCDATA)>
<!ATTLIST Gatto
Microchip ID #REQUIRED
Genere (maschio | femmina) #REQUIRED
Pelo (lungo | medio | corto) #IMPLIED
Vaccino (sì | no) #REQUIRED >
]>
```

Document-tree



Documento XML




```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Catalogo SYSTEM "Catalogo.dtd">
<Catalogo>
<Gatto Microchip="s000 3579AX" Genere="maschio" Pelo="corto" Vaccino="si">
<Nome> Bubu </Nome>
<Età> 12 </Età>
<Padrone> Elena </Padrone>
<Specie> Soriano </Specie>
</Gatto>
<Gatto Microchip="f001 7689BY" Genere="maschio" Pelo="lungo" Vaccino="no">
<Nome> Romeo </Nome>
<Età> 4 </Età>
<Padrone />
<Specie> Certosino </Specie>
</Gatto>
…
</Catalogo>
```