Gestione documentale

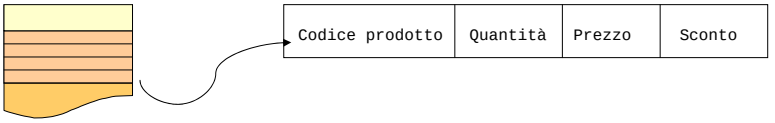
Lezione n.8

Information retrieval

Lezione n.8.1

Parliamo di testi

- Il testo è una forma *destrutturata*
- *La semantica delle parti non è esplicitata*
- Necessita di tecnologie apposite per essere rappresentato, elaborato, archiviato, ricercato
- Non si può pensare di utilizzare le stesse tecniche dei *database* a meno di scomporre il testo in *campi*
- Es: una fattura è un *documento* ma la posso comporre in parti strutturate (testa, righe, coda). Il testo viene *mappato* in campi. La semantica è espressa attraverso la posizione che assume nella tabella (coppia: campo,valore), ma si perde *l'integrità* del documento



Il documento

- Il documento contiene soprattutto testo ma
- contiene anche dati multimediali:video, audio, foto, immagini, grafici, etc.
- A differenza del dato il documento è solitamente un oggetto non strutturato
- Difficilmente suddivisibile in parti omogenee
- Variabile in lunghezza, contenuto, struttura

I sistemi documentali

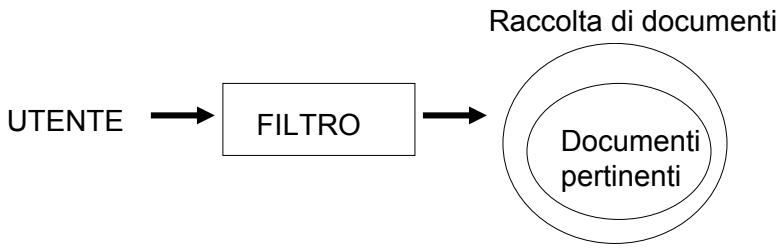
- I documenti sono di difficile gestione mediante i DBMS
- Alternative:
  - Word processor - automazione d'ufficio
  - IRS - sistemi di Information Retrieval
  - DMS - Document management system
  - Ipertesti
  - Linguaggi di marcatura - SGML, HTML, XML

I sistemi documentari e di information retrieval

- Sistemi documentari (information retrieval): banche dati di testi liberi
  - Forniscono l'accesso ai testi (documenti) in essi contenuti
  - Composti di unità documentali (formato = campi che identificano e descrivono il documento, più il contenuto del documento in testo libero)

Definizione di IRS

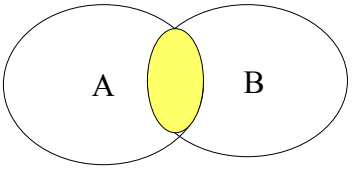
- **Information reterival system** è un sistema software per la gestione, l'indicizzazione e il reperimento interattivo di documenti che soddisfano una certa esigenza informativa dell'utente



Obiettivi di Information Retrieval

- Efficienza: prestazione buone
- Efficacia: rispondere all'esigenza informativa, la rilevanza della risposta in confronto alla richiesta

- Documenti rilevanti = documenti che sono pertinenti con la domanda posta dall'utente = A
- Documenti ritrovati = documenti che costituiscono la risposta del sistema = B



Parametri di misura di IRS

- Precision = A ∩ B/ B - misura il rumore ovvero quanti documenti non rilevanti sono stati proposti dal sistema fra i documenti risultanti
- Recall = A ∩ B/ A - misura il silenzio ovvero quanti documenti rilevanti non sono stati estratti e sono quindi rimasti all'interno del sistema
- Il sistema perfetto ottiene P=R=1 ossia dove tutti i documenti ritrovati sono pertinenti, numeri piccoli es. 0,03 indicano valori non buoni
- Non esistono IRS perfetti

Obiettivi della ricerca:

minimizzare il *rumore* e massimizzare la *pertinenza*

- L'efficacia si misura confrontando il numero di documenti ritrovati con il numero di documenti rilevanti:
- ***caso 1***: B=numero ritrovati 250, A=pertinenti 50, A ∩ B=30 → significa che 20 documenti pertinenti non sono stati trovati P=A ∩ B/B=30/250=0,12 → troppo rumore
- ***caso 2***: B=numero ritrovati 10, A=pertinenti 200, A ∩ B= 10 → significa che almeno 190 non sono stati ritrovati R=A ∩ B/A= 10/300= 0,05 → troppo silenzio
- ***caso 3***: B=numero ritrovati 11, A=pertinenti 10, A ∩ B=9 → P=A ∩ B/B=9/11=0,8 R=A ∩ B/A=9/10=0,9 ottimo!!

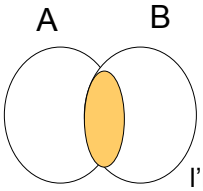
Gli operatori booleani - (I)

Operatori booleani:

- **and** "canta" e "danza" ossia la condizione è vera quando il soggetto canta e danza contemporaneamente
- **or** "canta" o "danza" - in modo alternativo ossia sono valide le seguenti situazioni: "canta e non danza", oppure "danza e non canta", oppure "canta e danza"
- **not** non "canta" - negazione
- **xor** "canta" o "danza" - in modo esclusivo ovvero sono valide le seguenti situazioni "canta e non danza" oppure "danza e non canta"

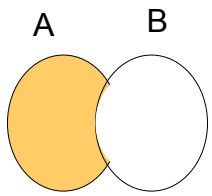
Gli operatori booleani - (II)

A = l'insieme degli avvocati
B = l'insieme dei professori



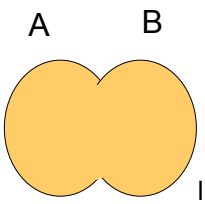
A AND B

l'insieme degli avvocati che sono anche professori



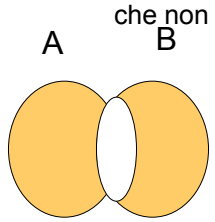
NOT B

l'insieme degli avvocati che non sono professori



A OR B

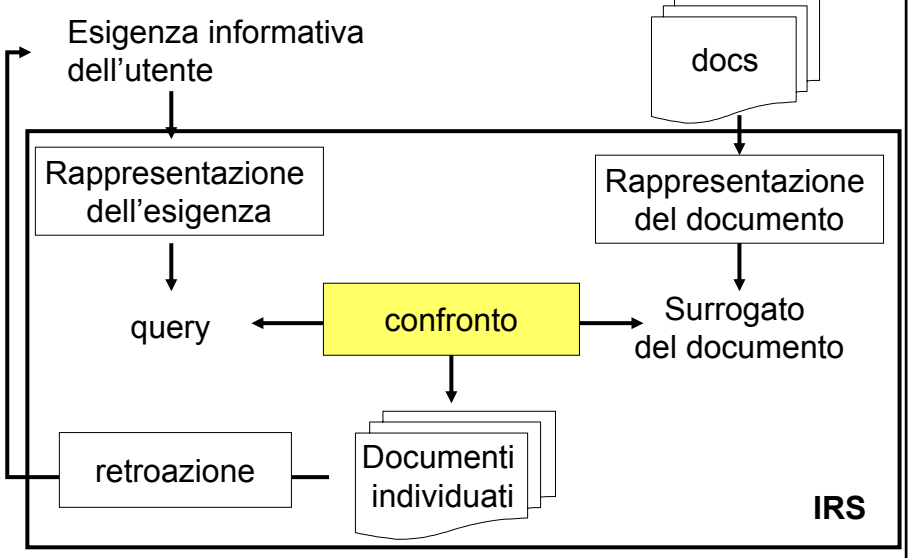
l'insieme di tutti e due



A XOR B

l'insieme dei soli avvocati e dei soli professori

Modello di un IRS



Architettura di un IR

- un IR è composto da:
  - componente per la memorizzazione dei documenti
  - componente per la rappresentazione del contenuto dei documenti
  - componente per la rappresentazione dei dati strutturati
  - componente di interfaccia con l'utente

Linguaggi di marcatura e l'XML
eXtensible Markup Language

Lezione n.8.2

Linguaggio di marcatura

- Per *linguaggio di marcatura* intendiamo un *qualsiasi meccanismo di annotazione* che consenta di evidenziare una parte del *testo* ed assegnargli un *significato*
- Il processo di marcatura arricchisce il *testo* con *elementi* aggiuntivi i quali racchiudono parti di contenuto
- Gli *elementi* che consentono di marcare il testo si dicono *marcatori* e vengono chiamati *tag*
- I *tag* sono composti da un elemento di apertura e da un elemento di chiusura
- <title>La divina commedia</title>

apertura **chiusura**

Linguaggio di marcatura

- Solitamente il linguaggio di marcatura cerca di:
  - lasciare inalterato il documento originale nel suo contenuto rispettando l'autore
  - distinguere i tag che servono alla stampa/presentazione da quelli che servono per rappresentare il contenuto
  - apporre in modo evidente e separato tutti gli elementi aggiuntivi di arricchimento del testo ossia di informazioni sul documento che non sono contenute nel documento stesso (*metadati*)
  - non *piegare* la rappresentazione del documento a nessuna funzione di processo in particolare ma cercare di rappresentare l'essenza ontologica del documento

Esempi

- Quando si stampa un libro si appongono delle marcature *tipografiche* (grassetto, corsivo, sottolineato, ecc.)
- Quando si sottolinea con un evidenziatore le parti rilevanti di un testo si applica un processo di marcatura
- I glossatori della scuola di *Irnerio* apponevano segni di distinzione nel testo arricchendo il testo stesso con note
- HTML è un linguaggio di marcatura

Tipi di Markup

La marcatura può avere diverse funzioni:

- Puntuazionale – di punteggiatura
- Presentazionale – tipografiche
- Procedurale – di impaginazione, per l'elaborazione
- Descrittivo – per descrivere parti significative del testo: titolo, autore, editore di un testo
- Referenziale – per collegare parti di testo interne fra loro (note a piè di pagina, riferimenti incrociati) o collegare parti di un testo con un altro (link)
- Meta-markup – per descrivere il *modello* di marcatura

Breve storia del markup

- GCA-GenCode e IBM-GML (1968-70)
  - GenCode è il risultato della standardizzazione dei codici di tipografia (*Graphics Communications Association*)
  - *Generalized Markup Language* di IBM è il linguaggio di markup per la documentazione interna e il prodotto BookMaster
- 1986: SGML è standard ISO 8879
  - Nel 1988 il dip. della difesa americano (DoD) adotta SGML per l'iniziativa CALS (*Continuous Aided Logistic Support*)

Breve storia del markup

- 1991: HTML
  - Tim Berners Lee (CERN - Ginevra) inventa il “World Wide Web”
  - HTML è “ispirato” ad SGML. Solo in seguito verrà corretto per adeguarsi a SGML
- 1997: XML
  - Nel 1995 il W3C decise di creare un linguaggio di markup con la completezza di SGML e la semplicità di HTML: *Extensible Markup Language* (XML)
  - Nel 1997 è uscito il primo standard per il linguaggio di markup (XML 1.0). In seguito i linguaggi connessi (XML-Namespaces, X-Pointer, X-Link, XSL, XPath, ecc.)

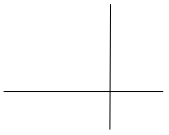
Alcuni esempi

- Rich text format RTF – linguaggio di marcatura presentazionale di proprietà della Microsoft
- LaTex – linguaggio di marcatura a scopi tipografici
- HTML – linguaggio di marcatura presentazionale e referenziale (creare ipertesti)
- XML – linguaggio di marcatura descrittivo, meta-linguaggio

HTML

- L'HTML è un linguaggio di marcatura
- Contiene un numero limitato di *tag* per questo si dice *chiuso,* non estensibile
- Non distingue lo strato di *rappresentazione* del contenuto con lo strato di *presentazione* grafica del contenuto:
  - <title> è un tag HTML che definisce il titolo della finestra di un file HTML
  - <b> è un tag HTML che definisce la marcatura in bold della parte racchiusa dal tag
- Non è gerarchico
- Non è rigoroso (es: vi possono essere tag aperti)

XML



- XML – eXstansible Markup Language
- Nato per la descrizione documentale viene poi applicato con successo anche nella descrizione di dati strutturati
- E' un *meta-linguaggio* (linguaggio per creare linguaggi) per questo estensibile ossia si possono definire nuovi tag
- Distingue lo strato di *rappresentazione* del contenuto con lo strato di *presentazione* grafica del contenuto:

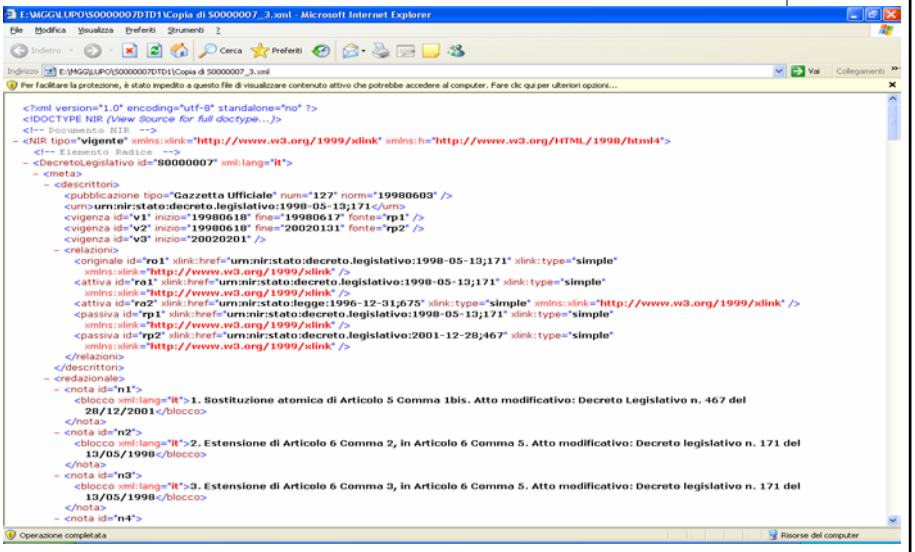
XML

- E' *gerarchico*
- E' *rigoroso*
- E' un formato *aperto* non proprietario
- E' *indipendente* dalla piattaforma hardware e software
- E' *indipendente* dagli applicativi

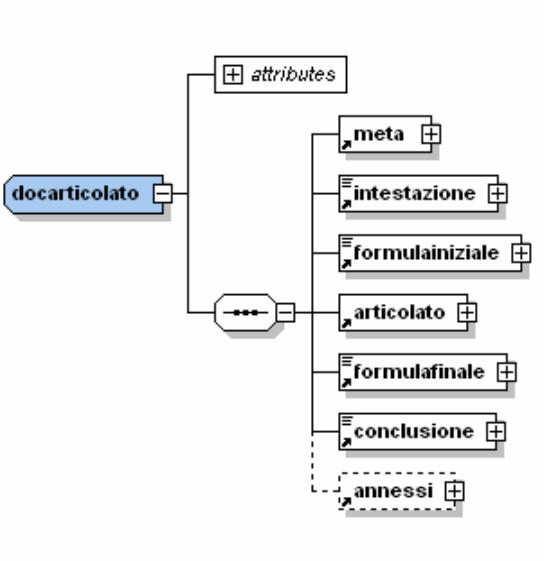
DTD e XML-SCHEMA

- Mediante appositi strumenti di possono definire delle grammatiche per la descrizione di un certo tipo di documento ossia si creano dei "template" o "modelli"
- Questi modelli divengono la base per comporre i testi all'interno di una certa comunità (standard)
- Es. Standard NormeInRete, Circolare AIPA/41/2001

La struttura ad albero



Modello di un frammento di NormeinRete: docarticolato



Stessa definizione con un altro linguaggio - DTD

- DTD – document type definition
- Definizione di uno schema o modello mediante regole
- Esempio di una regola per definire un documento articolato
- In questa regola si dice che l'elemento docarticolato è composto da altri elementi quali meta, intestazione, formulainiziale, articolato, formulafinale, conclusione, annessi
- In particolare l'ordine è definito dalla "," e il simbolo "?" indica la facoltatività dell'elemento

```
<!ELEMENT docarticolato
(meta,intestazione,formulainiziale,articola
to,formulafinale,conclusione,annessi?) >
```

Stessa cosa in sintassi XML-schema

```
<xsd:complexType name="docarticolato">
        <xsd:sequence>
                <xsd:element ref="meta"/>
                <xsd:element ref="intestazione"/>
                <xsd:element ref="formulainiziale"/>
                <xsd:element ref="articolato"/>
                <xsd:element ref="formulafinale"/>
                <xsd:element ref="conclusione"/>
                <xsd:element ref="annessi" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attributeGroup ref="globalinorma"/>
   </xsd:complexType>
```

Componenti essenziali dell'XML

- Elementi
  - Definiscono l'ossatura dell'XML
  - Sono le etichette che si mettono intorno al testo per qualificarlo e dargli un significato
  - Il database entra così nel testo
- Attributi
  - Proprietà o qualità che dell'elemento che non sono presenti necessariamente nel testo
- Entità
  - abbreviazioni
- Namespace
  - Prefisso posto davanti ai tag per non confonderli in caso di omonimia

Gli elementi

Esiste sempre un elemento radice detto root
L'elemento viene definito tramite i due *tag* di apertura e chiusura

<tipoDoc>**Decreto legislativo**</tipoDoc> →elemento tipoDoc
<numDoc>**171**</numDoc> →elemento numDoc
<titoloDoc>**Disposizioni in materia di tutela** →elemento titoloDoc
della vita privata nel settore delle telecomunicazioni,
in attuazione della direttiva 97/66/CE del Parlamento
europeo e del Consiglio, ed in tema di attività
giornalistica
</titoloDoc>
<dataDoc norm="**19980513**">**13 maggio 1998**</dataDoc> →elemento dataDoc

Elementi e sotto_elementi

<intestazione>
<tipoDoc> **Decreto legislativo**</tipoDoc>
<numDoc>**171**</numDoc>
<titoloDoc >**Disposizioni in materia di tutela della vita privata nel settore delle telecomunicazioni, in attuazione della direttiva 97/66/CE del Parlamento europeo e del Consiglio, ed in tema di attività giornalistica**</titoloDoc>
<titoloDoc tipo="breve" nome="Codice Privacy"/>
<dataDoc norm="**19980513**">**13 maggio 98**</dataDoc>
</intestazione>

intestazione è formato dai sotto elementi

tipoDoc, numDoc, titoloDoc, dataDoc

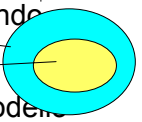
La gerarchia è data dalla chiusura dei tag e dal meccanismo di inclusione degli stessi

Attributi

- Gli attributi sono informazioni note del testo ma non presenti nel testo e quindi devono essere inseriti in appositi spazi mediante il meccanismo denominato "nome-valore"
- Il nome è fisso
- Il valore varia
- <pubblicazione tipo="**Gazzetta Ufficiale**" num="**127**" norm="**19980603**" />
- Attributi dell'elemento `pubblicazione`:
  - Tipo
  - Num
  - Norm (data pubblicazione normalizzata ossia annomesegiorno)

Documenti ben formati e validi

- un documento XML è ***ben formato*** (well-formed) quando rispetta le regole di sintassi del linguaggio stesso
- un documento XML è ***valido*** se è conforme ad un modello prestabilito che ne descrive gli elementi, la struttura, i vincoli fra gli elementi
- tali vincoli vengono definite mediante una ***grammatica***
- definire un modello consente di passare dallo strato di meta-linguaggio al linguaggio
- vi sono due diverse tecnologie per definire modelli:
  - DTD – document type definition
  - XML - Schema



Differenze fra DTD e XML-schema

- DTD
  - sono nate prima
  - sono più leggibili all'uomo
  - sono più adatte a modellare testi
  - non possono definire le tipologie di dati
  - non possono definire classi ed ereditarietà fra classi
  - non si possono imporre limitazioni ai tipi
  - liste di attributi uguali non si possono definire
  - usano un linguaggio di definizione non XML

Differenze fra DTD e XML-schema

- XML-schema
  - sono nate dopo per sanare le precedenti lacune
  - sono meno leggibili all'uomo più leggibili alla macchina
  - sono più adatte a modellare dati strutturati
  - possono definire tipologie di dati complessi
  - possono definire classi ed ereditarietà fra classi
  - possono imporre limitazioni ai tipi
  - si possono definire liste di attributi uguali
  - sono definite in XML

Perché definire modelli

- per creare **standardizzazione**
- per incentivare ***l'interoperabilità*** sia fra gli uomini sia fra gli applicativi
- per dare regole ***prescrittive***
- per formalizzare regole ***descrittive***
- per creare schemi di comportamento uniformi (***processi***)
- per condividere ***metadati*** all'interno di una comunità

Parser

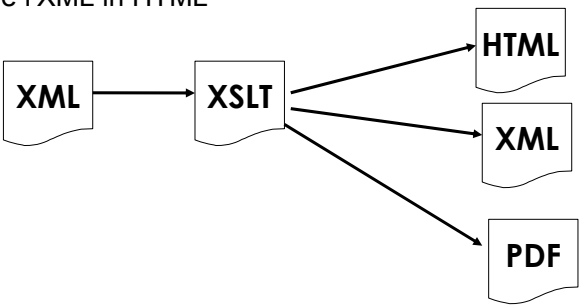
- Parser non validanti
- Parser validanti
  - Dom
    - Carica tutto l'albero in memoria e lo esamina in modo atomico
  - Sax
    - Carica solo la parte di albero sollecitata da un evento e quindi ha in memoria solo un frammento del documento
  - altri

XSLT

- XSLT è un linguaggio della famiglia XML per la trasformazione di un file XML in altri file o XML o HTML o altri formati
- XSLT trasforma mediante un processo il file originale XML
- XSLT viene utilizzato per diversi scopi:
  - per trasformare un XML in un altro XML - per esempio quando si cambia versione della DTD occorre trasformare il file precedente in un nuovo file XML valido per la nuova DTD
  - per inserire delle istruzioni di processo – per esempio per poi dare il risultato ad DBMS
  - per visualizzare graficamente il file XML e renderlo quindi fruibile tramite i browser

Meccanismo XSLT

- Alcuni browser supportano l'engine XSLT standard e quindi fornendo le specifiche di layout si può direttamente dal browser visualizzare il documento nel formato di presentazione desiderato
- Altri browser non supportano l'engine e quindi occorre trasformare l'XML in HTML



Domande possibili

- Cosa è l'XML? Sue caratteristiche
- Cosa vuol dire che l'XML è un meta-linguaggio?
- Da quali componenti è formato (elementi, attributi)?
- Cosa sono le DTD e gli XML-schema? E che beneficio apportano alla società dell'informazione?
- Cosa significa che un XML è valido e che differenza c'è con well-formed?
- Cosa è un parser?
- Come si visualizza sul Web un XML?