SGML e XML

Fabio Vitali
Università di Bologna

Una citazione

«Quando io uso una parola», disse Humpty Dumpty in tono piuttosto sprezzante, «significa quello che io scelgo che significhi - né più, né meno.»

Lewis Carroll

“Attraverso lo specchio”

Perché SGML?

- Quando si porta una collezione di documenti in forma elettronica, si ha in mente in generale una specifica applicazione (metterla in rete, prepararla per la stampa, ecc.).
- Di solito, si cerca di trasformare il documento nella forma più opportuna perché venga utilizzato nell'applicazione suddetta.
- Spesso per questo si fanno delle scelte che impediscono o ostacolano notevolmente un ulteriore riuso della stessa collezione per una diversa applicazione. L'impaginato poco si adatta ad un'indicizzazione per la rete, o viceversa i linguaggi di visualizzazione su rete sono troppo poco sofisticati per una produzione tipografica di buon livello, ecc.
- SGML è il linguaggio più opportuno per strutturare e marcare i documenti in maniera indipendente dall'applicazione, favorendo la riusabilità, la flessibilità e la apertura ad applicazioni complesse.

Sommario

- Cos’è il markup?
- Storia di SGML
- Analisi dei documenti
- La sintassi di SGML
- XML e standard connessi

Il markup

Definizione e tipi

SGML

Cos’è il markup?

Definiamo markup ogni mezzo per rendere esplicita una particolare interpretazione di un testo. Per esempio, tutte quelle aggiunte al testo scritto che permettono di renderlo più fruibile.

Lo scopo del markup

Oltre a rendere il testo più leggibile, il markup permette anche di specificare ulteriori usi del testo. Con il markup per sistemi informatici (il nostro caso), specifichiamo le modalità esatte di utilizzo del testo nel sistema stesso.

Esempi di uso del markup

- Possiamo sistemare, riorganizzare e formattare un testo per la stampa.
- Possiamo sistemare, riorganizzare e formattare un testo per la lettura a video.
- Possiamo sistemare, riorganizzare e formattare un testo per l'uso di handicappati fisici (ad esempio ciechi con l'aiuto di un lettore Braille)
- Possiamo sistemare, riorganizzare e formattare un testo per la lettura del testo ad alta voce in situazioni di impedimento temporaneo (ad esempio, mentre stiamo guidando).
- Possiamo permettere facilmente una ricerca sui contenuti del testo.
- Possiamo verificare la adeguatezza del testo rispetto a regole più o meno formali di strutturazione.

Un testo su carta

Tre Uomini in Barca

Jerome K. Jerome

1889

Capitolo primo

Tre invalidi - Le sofferenze di George e Harris - La vittima di centosette malattie inguaribili - […]

Eravamo in quattro: George, William Samuel Harris, e io, Montmorency. Standocene seduti in camera mia, fumavamo e parlavamo di quanto fossimo malridotti… malridotti, dal punto di vista della salute, intendo, naturalmente.
Ci sentivamo tutti piuttosto giù di corda, ...

3

Il testo senza markup

- Questo è il testo completamente senza markup, come poteva essere scritto su un papiro della biblioteca di Alessandria, nel II o III secolo a.C.

Treuominiinbarcajeromekjerome1889capitoloprimotreinvalidilesofferenzedigeorgeeharrislavittimadicentosettemalattieinguaribilieravamoinquattrogeorge,williamsamuelharriseiomontmorencystandocenesedutiincameramiafumavamoeparlavamodiquantofossimomalridottimalridottidalpuntodivistadellasaluteintendonaturalmentecisentivamotuttipiuttostogiùdicorda

Markup metabolizzato

- Maiuscole/minuscole, punteggiatura, spazi e ritorni a capo sono elementi di markup. Su di essi non riflettiamo più di tanto, perché sono ormai metabolizzati nel sistema di scrittura.

Tre Uomini in Barca
Jerome K. Jerome (1889)
Capitolo primo
Tre invalidi - Le sofferenze di George e Harris - La vittima di centosette malattie inguaribili - […]
Eravamo in quattro: George, William Samuel Harris, e io, Montmorency. Standocene seduti in camera mia, fumavamo e parlavamo di quanto fossimo malridotti… malridotti, dal punto di vista della salute, intendo, naturalmente.
Ci sentivamo tutti piuttosto giù di corda, ...

Markup procedurale

- Sono comandi, o istruzioni che il sistema di lettura (umano o elettronico) deve eseguire sul testo. Ad esempio, istruzioni su come andare a capo, come decidere i margini, ecc.

{\rtf1 \mac \ansicpg10000 \uc1 \pard \plain \s15 \qc \widctlpar \adjustright \f4 \fs48 \cgrid {**Tre Uomini in Barca** \par } \pard \plain \widctlpar \adjustright \f4 \cgrid { \line \par \par \par \par \par } \pard \plain \s1 \qc \keepn \widctlpar \outlinelevel0 \adjustright \i \f4 \fs36 \cgrid {**Jerome K. Jerome** \par } \pard \plain \qc \widctlpar \adjustright \f4 \cgrid { \fs36 […] \par **1889** \line \par } \pard \widctlpar \adjustright { \page } { \b \fs36 **Capitolo primo**} { \par \par \par \line } { \i **Tre invalidi - Le sofferenze di George e Harris - La vittima di centosette malattie inguaribili - [** \u8230 \'c9**]** \par } { \line } { \fs28 **Eravamo in quattro: George, William Samuel Harris, e io, Montmorency. Standocene seduti in camera mia, fumavamo e parlavamo di quanto fossimo malridotti** \u8230 \'c9 **malridotti, dal punto di vista della salute, intendo, naturalmente. \line Ci sentivamo tutti piuttosto gi** \u249 \'9d **di corda, ...** \par }}

Markup dichiarativo

- Sono informazioni (dichiarazioni) sugli elementi del documenti, che ne specificano il ruolo, la giustificazione, la relazione con gli altri elementi.

```
<ROMANZO><TITOLO>Tre Uomini in Barca</TITOLO>
<AUTORE>Jerome K. Jerome</AUTORE>
<ANNO>1889</ANNO>
<CAPITOLO><TITOLO>Capitolo primo</TITOLO>
<INDICE><EL>Tre invalidi</EL><EL>Le sofferenze di George e Harris
</EL><EL> La vittima di centosette malattie inguaribili </EL>[…]</INDICE>
<PARA>Eravamo in quattro: George, William Samuel Harris, e io,
Montmorency. Standocene seduti in camera mia, fumavamo e parlavamo di
quanto fossimo malridotti… malridotti, dal punto di vista della salute, intendo,
naturalmente. </PARA>
<PARA>Ci sentivamo tutti piuttosto giù di corda, …</PARA>
</CAPITOLO>… </ROMANZO>
```

Caratteristiche del markup procedurale (1)

- Basato sull'aspetto
  - Ad ogni elemento del documento viene associata la procedura per visualizzarlo in maniera voluta: font, dimensione, corsivi, grassetti, margini, interlinea, ecc.
- Dipendente dal sistema
  - ogni sistema di visualizzazione impone le proprie regole e la propria sintassi, dipendendo da:
    - Filosofia sintattica (comandi-punto per `troff`, comandi-barra per RTF, ecc.)
    - Capacità di raggruppamento (parentesi graffe in RTF, comando di disattivazione esplicita in troff, ecc.)
    - Supporto di specifiche funzionalità

Caratteristiche del markup procedurale (2)

- Associato agli individui
  - ogni elemento possiede le proprie procedure per la visualizzazione, che possono anche essere tutte diverse.
- Non contestuale
  - Le regole di visualizzazione non dipendono dal contesto in cui vengono fatte, ma ognuna fa specie a sé.
  - Ad esempio, una lista in `troff` è fatta come segue:

```
.li
 .it elemento 1
 .it elemento 2
.el
```

  - Nessun controllo impone di chiudere la lista alla fine, o di usare i comandi .it solo dentro alla lista.

Caratteristiche del markup dichiarativo (1)

- Basato sul ruolo
  - Di ogni elemento viene descritto il ruolo all'interno del testo, più che le regole per la sua visualizzazione:
  - Ad esempio: "questo è un titolo, questo è un paragrafo, questo è il nome dell'autore, questa è una citazione."
- Indipendente dal sistema
  - Poiché il markup dichiarativo assegna **ruoli** (e non regole di visualizzazione) agli elementi del testo, questi sono intrinseci agli elementi stessi, e non alle funzionalità disponibili nel sistema di visualizzazione.
  - Un sistema incapace di variare l'interlinea, o con un elenco limitato di font e dimensioni, può aver problemi ad interpretare un markup procedurale troppo ricco, ma la differenza tra un titolo o un elenco e un paragrafo non dipende dalla sofisticazione del sistema di visualizzazione.

Caratteristiche del markup dichiarativo (2)

- Basato su categorie
  - I ruoli sono categorie. Ogni elemento è associato ad una categoria, e ne riflette tutte le caratteristiche automaticamente.
- Contestuale
  - Con il markup dichiarativo è possibile definire delle regole che permettano o impediscano l'assegnazione di una categoria (ruolo) ad un elemento del testo a seconda del contesto.
  - Ad esempio, si può richiedere che il titolo vada all'inizio del testo, o che una lista sia composta solo di elementi della lista, e non da paragrafi, ecc.

SGML

- SGML (Standard Generalized Markup Language) è uno ***standard***.
- SGML è un ***meta-linguaggio non proprietario*** di markup dichiarativo.
- Facilita markup ***leggibili***, ***generici***, ***strutturali***, ***gerarchici***.

Glossario: ***linguaggio standard***

- SGML è uno standard ISO (International Standard Organization) n. 8879 del 1986. ISO è l'organizzazione mondiale degli standard, la più importante.
- Essere uno standard per SGML significa che esso è il risultato di discussioni e compromessi di rappresentanti di tutte le comunità interessate, che è un investimento ***nel tempo*** di queste comunità, e che non dipende dai piani commerciali o dai capricci di una singola casa produttrice.

Glossario: ***meta-linguaggio di markup***

- Un meta-linguaggio è un linguaggio per definire linguaggi, una grammatica di costruzione di linguaggi.
- SGML non è un linguaggio di markup, ma un linguaggio con cui definiamo linguaggi di markup.
- SGML non dà dunque tutte le risposte di markup che possano sorgere a chi vuole arricchire testi per qualunque motivo, ma fornisce una ***sintassi*** per definire il linguaggio adatto.
- SGML non sa cos'è un paragrafo, una lista, un titolo, ma fornisce una grammatica che ci permette di definirli.

Glossario: ***linguaggio non proprietario***

- Non proprietario significa che non esiste un'unica ditta o casa produttrice che ne detiene il controllo.
- Viceversa RTF è © Microsoft, mentre PostScript e Acrobat sono © Adobe.
- Essendo standard non proprietario, non dipende da un singolo programma, da una singola architettura. Gli stessi dati possono essere portate da un programma all'altro, da una piattaforma all'altra senza perdita di informazione.
- Inoltre, anche l'evoluzione nel tempo è assicurata per lo stesso motivo.

Glossario: ***markup leggibile***

- In SGML il markup è posto in maniera leggibile a fianco degli elementi del testo a cui si riferiscono.
- Essi possono sia essere usati da un programma, sia letti da un essere umano. Possono sia essere aggiunti da un programma (***editor***), sia da un essere umano con un programma non specifico.
- Il markup in SGML è semplice testo facilmente interpretabile.

Glossario: ***markup generico***

- Il markup in SGML non è pensato unicamente per la stampa su carta. E’ possibile combinare markup utile per scopi o applicazioni diverse, ed in ogni contesto considerare o ignorare di volta in volta i markup non rilevanti.

Glossario: ***markup strutturato***

- SGML permette di definire delle strutture, suggerite o imposte, a cui i documenti si debbono adeguare.
- Ad esempio, si può imporre che un testo sia diviso in capitoli, ognuno dei quali dotato di un titolo, una breve descrizione iniziale e almeno un paragrafo di contenuto.
- È cioè possibile definire una serie di regole affinché il testo sia considerabile strutturalmente corretto.

Glossario: ***markup gerarchico***

- Le strutture imposte da SGML sono tipicamente a livelli di dettaglio successivi.
- Gli elementi del testo possono comporsi gli uni con gli altri, permettendo di specificare la struttura in maniera gerarchica.
- Ad esempio, si può imporre che il libro sia fatto di capitoli, e che questi a loro volta siano fatti di un una descrizione e molti paragrafi. Una descrizione sarà quindi fatta di elementi, ciascuno dei quali sarà una frase composta di testo; i paragrafi saranno testo eventualmente contenenti anche elementi in evidenza o figure.

Storia di SGML

Come si è arrivati a SGML

Anni ‘60: primo WP in IBM

- Uomini e animali sono già nello spazio grazie ai computer, le aziende telefoniche ed elettriche già calcolano le bollette con i computer, eppure prima del 1964 nessuno aveva ancora trattato i testi con i computer.

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.

Anni ‘80: WP WYSIWYG e DTP

- Si diffondono i primi Word Processor, come WordStar, Word Perfect, MS Word
- Nasce il concetto di WYSIWYG (***What You See Is What You Get***): MacWrite e poi MS Word, ecc. L’enfasi è sulla verosimiglianza tra ciò che si vede sullo schermo e ciò che risulta sulla carta.
- Nascono i primi prodotti da editoria professionale (DTP: ***Desk Top Publishing***): PageMaker, Ventura, Quark Xpress, ecc.

1986: SGML è standard ISO 8879

- Giudizi misti: "***Sounds Great, Maybe Later***": manca il software, è considerato complicato e sofisticato.
- Nel 1988 il dip. della difesa americano (DoD) adotta SGML per l'iniziativa CALS (***Continuous Aided Logistic Support***)
- 1990: TEI (***Text Encoding Initiative***): un gruppo di umanisti generano linee guida per la strutturazione dei testi umanistici (prosa, teatro, poesia, ecc.).

1991: HTML

- Tim Berners Lee (CERN - Ginevra) inventa un sistema ipertestuale per la rete Internet chiamato “World Wide Web”. Il WWW è basato su tre protocolli:
  - HTTP (***HyperText Transfer Protocol***) è un semplice protocollo di comunicazione TCP/IP tra client (browser) e server per accedere a documenti remoti.
  - URL (***Universal Resource Locator***) è un formato di denominazione delle risorse accessibili in rete.
  - HTML (***HyperText Markup Language***) è un linguaggio di markup per semplici documenti ipertestuali.
- HTML non nasce come un’applicazione SGML, ma solo come “ispirato” ad SGML. Solo in seguito verrà corretto per adeguarsi a SGML.

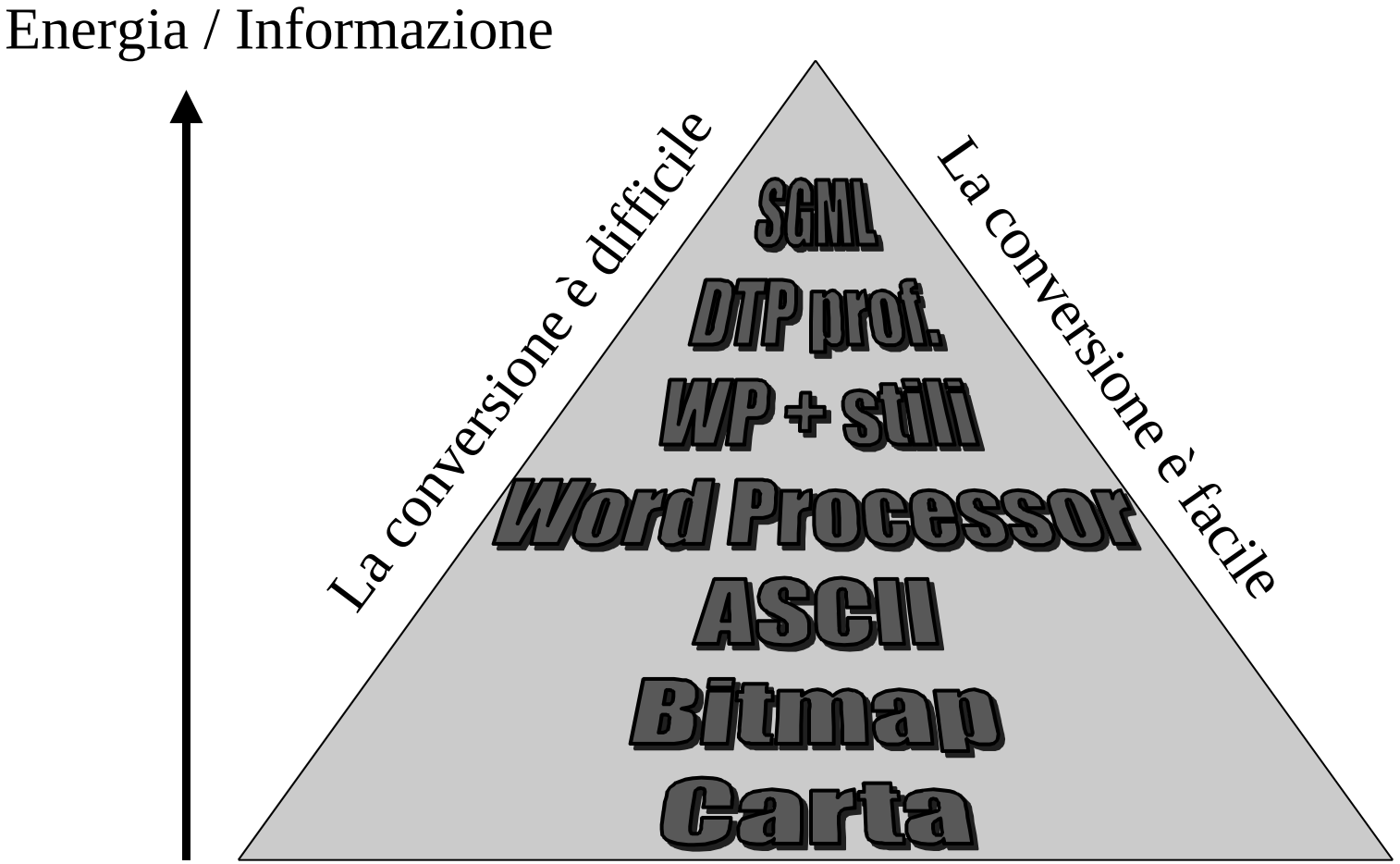
1997: la convergenza su XML

- Lo sviluppo e la correttezza degli standard connessi con il WWW sono gestiti dal W3C (World Wide Web Consortium).
- Nel 1995 la commissione sui linguaggi di markup decise di creare un nuovo linguaggio di markup con la completezza di SGML e la semplicità di HTML: ***Extended Markup Language*** (XML).
- Nel 1997 è uscito il primo standard per il linguaggio di markup (XML 1.0). In seguito i linguaggi connessi (XML-Naming, X-Pointer, X-Link, XSL, ecc.).

Analisi dei documenti

Analisi dei documenti, classificazione degli elementi, uso delle applicazioni

Perché conviene SGML?



Analisi dei documenti

- Questa sezione è dedicata alla analisi dei documenti da trasformare in SGML (XML), e alla preparazione del DTD più adatto allo scopo.
- Vi sono quattro fasi nella analisi dei documenti e preparazione del DTD:
  - Identificazione e classificazione dei componenti e verifica
  - Selezione dei componenti, costruzione della gerarchia, dei blocchi informativi e degli elementi di dati
  - Identificazione delle connessioni
  - Verifica e miglioramento iterativo delle specifiche

Classificazione dei componenti

- La parte più importante del lavoro di progettazione di un'applicazione SGML è l'identificazione delle strutture e del significato delle parti dei documenti e delle loro relazioni.
- L'identificazione semantica dei componenti avviene quando si evidenzia l'esigenza di distinguere tra un tipo di dati ed un altro.
- Se due pezzi diversi di un documento contengono lo stesso tipo di informazione, li si deve considerare appartenenti alla stesso componente semantico, anche se sono separati tra loro.
- Viceversa, se due pezzi contengono due tipi diversi di informazione, o è necessario distinguerli in qualche maniera per i fini dell'applicazione, allora debbono essere distinti in due componenti semantici separati.
- Possiamo distinguere tra tre tipi di classificazione dei componenti di un documento: ***contenuto, struttura*** e ***presentazione***.

Classificazione basata sul contenuto

- Si identificano i componenti per il significato che essi hanno, indipendentemente dalla loro posizione nel documento o dal loro aspetto grafico.
- Ad esempio:
  - indirizzi, città, codici postali;
  - ricette, ingredienti, tempi di preparazione;
  - termini, sviluppo grammaticale, significato.

Classificazione basata sulla struttura

- Si identificano i componenti per il loro ruolo all'interno del documento, per il senso che hanno in quella posizione e in quella forma
- Ad esempio:
  - sezioni, capitoli, liste, paragrafi, titoli

Classificazione basata sulla presentazione

- Si identificano i componenti per le variazioni nel modo in cui debbono apparire graficamente, senza implicazioni sul loro “vero significato”.
- Ad esempio:
  - Frasi con un determinato font o dimensione
  - Blocchi da mantenere sulla stessa pagina
  - Posti dove spezzare una pagina

Caratteristiche delle classificazioni

- Questi tre modi di classificare i componenti di un documento sono presenti contemporaneamente bell'analisi di un documento. Persone diverse, sugli stessi documenti, possono identificare questa o quella classe a seconda di professione, ***forma mentis***, esigenze.
- I tre tipi di classificazione hanno anche caratteristiche diverse di ***identificabilità, flessibilità*** e ***durata***.

Classificazione basata sul contenuto

- È la classificazione più complessa da realizzare, ma la più flessibile, identificabile, flessibile e duratura.
- Poiché identifico i componenti basati sul loro significato, è immediato identificare il senso di un componente.
- La classificazione è indipendente dalla struttura del documento e dall'aspetto grafico, così posso cambiare idea su queste decisioni in qualunque momento, e anche fornire soluzioni diverse sugli stessi componenti.
- Poiché un componente avrà sempre quel significato in qualunque contesto, anche cambiamenti di stile, organizzazione del documento ecc. non impediranno a questa classificazione di sopravvivere.

Classificazione basata sulla struttura

- La classificazione strutturale identifica l'organizzazione di un documento in maniera sufficientemente generale, ma rozza per quel che riguarda il senso del documento.
- È possibile in qualunque momento modificare la resa grafica degli elementi, ma non il loro ruolo nella struttura globale del documento.
- Cambiamenti stilistici non preoccupano (il font, la larghezza di un paragrafo, l'esistenza o meno di un bordo in una tabella), cambiamenti strutturali importanti invece sì (ad esempio, passare da una forma a lista ad una a tabella, ecc.).

Classificazione basata sulla presentazione

- In generale, ci sono più classi di contenuto che classi di presentazione. Per uniformità grafica e facilità di lettura molti componenti aventi significato diverso vengono resi graficamente nello stesso modo.
- L'identificazione delle sole classi grafiche fa sparire immediatamente l'identificabilità di elementi di significato diverso ma resa grafica uguale.
- Decisione successive di cambiamenti grafici di solo alcuni componenti, e non altri, saranno impossibili.
- Utilizzi del documento per scopi diversi dalla presentazione (ad esempio, la creazione di un indice, l'inserimento in un motore di ricerca, ecc.), saranno impossibili.

Regole guida per la classificazione

- Identificare il più possibile i componenti per il loro significato e contenuto. Richiede più lavoro ma ne vale la pena. È necessario, ovviamente, fermarsi ad un livello ragionevole di specificità.
- Attribuire a questi componenti significati strutturali (tabelle, liste, paragrafi, organizzazioni gerarchiche tipo sezioni, sotto-sezioni, ecc.).
- Specificare la resa grafica dei componenti. Quest'ultima specificazione tipicamente avviene al di fuori del contesto di SGML, con appositi strumenti e linguaggi di stylesheet.

Altri suggerimenti di classificazione: 1

- **Scartare elementi puramente presentazionali**: numero di pagina, elementi che si ripetono pagina dopo pagina (un logo, una decorazione, il nome di un capitolo) possono tipicamente essere aggiunti automaticamente dal formattatore e non è necessario considerarli come componenti del documento.
- **Identificare classi generali di informazioni**. Anche se presenti in varie parti del documento, alcune informazioni possono avere lo stesso significato e lo stesso ruolo, e quindi debbono essere identificati nella stessa maniera.

Altri suggerimenti di classificazione: 2

- **Identificare informazioni ripetute in varie parti del documento.** Alcune informazioni (nomi propri o di organizzazioni, riferimenti ad immagini, date importanti, elementi ripetuti di una struttura, ecc.) debbono essere presenti in varie parti del testo in maniera identica, e debbono cambiare in maniera coerente.È utile avere un componente unico che registri una sola volta l'informazione da stampare, e venga usato ovunque necessario.
- **Identificare i componenti che provengono da sistemi informativi esistenti**. Tipicamente un database ha già distinzioni di elementi basate sul contenuto. Se alcune informazioni provengono da un database è comodo e si risparmia tempo usare o basarsi sulla strutturazione dei dati già esistenti nel sistema informativo.

Altri suggerimenti di classificazione: 3

- **Identificare i componenti etichettati**: in certe classi di documenti, etichette descrittive introducono ripetutamente i componenti del documento. Ad esempio: manuali di riferimento, testi di legge, strutturazioni in capitoli, ecc.
- **Identificare i contenitori di altri sottocomponenti**. Forniscono la base iniziale di identificazione delle strutture gerarchiche di un documento.
- **Identificare i blocchi funzionali**. A volte componenti possono essere raggruppati in unità più grandi in cui la presenza e l'ordine di successione dei singoli elementi è importante.

Le applicazioni SGML

- Quali applicazioni SGML esistono e sono necessarie nella produzione di un sistema di elaborazione di documenti SGML dipende ovviamente da quale uso è previsto per i documenti.
- Detto questo, esistono molti strumenti di elaborazione di documenti SGML, che possono essere classificati nelle tipologie seguenti:
  - Tool di scrittura e modifica
  - Motori di conversione e trasformazione
  - Sistemi di gestione dei documenti
  - Sistemi di formattazione ed impaginazione
  - Sistemi di indicizzazione e ricerca

Uso dei documenti SGML

- Ci sono tre usi principali di un documento (o di una classe di documenti): la creazione e modifica, la organizzazione e memorizzazione, e l'uso finale.
- **Creazione e modifica**: che tipo di documenti? Che tipo di dati (testo, immagini, tabelle, schemi, equazioni, ecc.)? Che linguaggio? Di quale provenienza? Che rapporto c'è tra creatore e "editore"? Quanti creano? Quanti editano? Con quali scadenze? Quali caratteristiche di revisione?
- **Organizzazione e memorizzazione**: quanti tipi di documento? Che tipo di uso e riuso va previsto? Che tipo di processo deve essere applicato? Come vengono memorizzati? Che tipo di controllo di accesso deve essere applicato?
- **Uso**: che tipo di funzionalità va previsto (lettura e visione, ricerca e navigazione, ulteriori processi)? Su quale media (carta, video, CD-ROM, rete)? In quale forma (stampa, Braille, voce)?

Editor SGML

- Tipicamente, si usa una combinazione di varie classi di programmi: editor SGML, word processor tradizionali, applicazioni ad hoc, motori di conversione, ecc.
- Questi sistemi si combinano a seconda dei compiti, delle professionalità e dell’esperienza degli utenti per trovare il miglior compromesso tra facilità d’uso e potenza di funzioni.

Motori di conversione

- Possiamo distinguere tre tipi di conversione:
  - Conversione da dati pre-esistenti a dati SGML
  - Conversione da dati SGML in un formato a dati SGML in un altro formato
  - Conversione da dati SGML a dati in un formato non SGML
- Tipicamente i dati pre-esistenti avranno delle regolarità, o delle strutture già esistenti, o delle formattazioni sistematiche che possono essere usate. Il metodo allora è sfruttare queste regolarità e strutture per creare un mapping ai componenti SGML. Questo prta a conversioni lunghe, faticose e con una grossa componente manuale.
- Viceversa per una conversione degli altri due tipi tutte le informazioni necessarie alla conversione dovrebbero già essere presenti nel documento come markup o altrimenti. È allora necessario sfruttarle.

Document Management System

- Sono database specializzati nel gestire documenti e in particolare documenti SGML. Un buon database di documenti deve fornire:
  - Gestione di tutti i tipi di documenti, inclusi quelli non testuali
  - Funzioni di controllo degli accessi e lock
  - Funzioni di versionamento dei documenti
  - Funzioni di indicizzazione e ricerca
  - Funzioni di estrazione e composizione selettiva
  - Funzioni di ricerca complesse
  - Supporto per granularità variabile

Sistemi di impaginazione e formattazione

- Sono questi casi speciali di motori di conversione, che generano un documento convertito nel linguaggio usato da sistemi di fotocomposizione, stampa offset, ecc.
- In sostanza è a questo livello che si producono delle specifiche grafiche precise da associare ai componenti del documento. E a questo livello che si dimostra che il sistema SGML effettivamente serve.
- Sono proposti vari sistemi che utilizzano vari linguaggi di specifica più o meno standard, tra cui FOSI (***Formatting Output Specificazion Instance***) e DSSSL (Document Style Semantics and Specification Language).
- Questi linguaggi (detti linguaggi di stylesheet) permettono di generare documenti in Postscript, RTF o sistemi proprietari per la stampa tipografica.

Motori di indicizzazione e ricerca

- L’identificazione delle parti di un documento SGML che possono essere soggette a indicizzazione (perché più facilmente usate per la ricerca) è parte integrante dell’attività di analisi dei documenti.
- Questo garantisce correttezza del markup, validazione della struttura, correzione degli errori e del “rumore” ancora prima di creare le strutture utilizzate dal motore di ricerca.