Il markup di documenti

Fabio Vitali

«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”

Introduzione

Oggi esaminiamo:

- Cos’è il markup
- Che tipi di markup esistono
- Caratteristiche di markup procedurale e dichiarativo
- Una brevissima storia del markup su computer

Perché tutto questo chiasso?

- Quando si porta una collezione di documenti in forma elettronica, si ha di solito 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.
- I linguaggi di markup derivati da SGML sono i linguaggi più opportuni per strutturare e marcare i documenti in maniera indipendente dall'applicazione, favorendo la ***riusabilità***, la ***flessibilità*** e la ***apertura ad applicazioni complesse***.

Due problemi

Rendere i contenuti accessibili a chiunque

- Gestire in maniera semplice e uniforme tutte le fasi di gestazione di un prodotto editoriale
- Gestire in maniera semplice ed uniforme tutti i possibili usi di un medesimo contenuto: su carta, su video, su terminale Braille, su sintetizzatore vocale, ecc.
- Gestire in maniera semplice ed uniforme tutti i possibili riusi di un medesimo contenuto: libro, indice, sito web, catalogo, archivio, …

Rendere il testo accessibile nel tempo

- Il recupero dei dati del Census Bureau del 1960 (1976-1982)
- Il recupero dei dati per il progetto LUNR del 1969 (1986-????)
- Brewster Kahle http://www.archive.org/
- Bruce Sterling http://www.deadmedia.org/
- Boeing e la digitalizzazione dei manuali degli aerei.

Accessibilità nel tempo (1)

Medium fisico

- La corretta conservazione delle tracce ottiche, elettriche o magnetiche sul supporto fisico prescelto.

Hardware

- La periferica usata per la scrittura e lettura delle informazioni
- Il computer su cui questa periferica funziona

Software

- Sistema operativo in grado di far funzionare il computer
- Driver in grado di far funzionare le periferiche prescelte
- Applicazione in grado di leggere i dati dal medium fisico

Interpretazione

- Il formato dei dati in cui il contenuto è stato scritto(mescolando il contenuto vero e proprio con funzioni dell'applicazione originaria)
- La codifica dei caratteri usata (il meccanismo di conversione dei codici numerici in caratteri di un alfabeto)

Accessibilità nel tempo (2)

Problema HW: migrazione dei dati

- La predominanza degli hard disk sui medium esterni e il costo in diminuzione dell'hardware connesso rendono possibile il trasferimento di tutti i dati sulla nuova macchina appena acquistata

Problema SW: conversione

- Non altrettanto semplice, non altrettanto efficace, non altrettanto automatizzabile, non altrettanto universale.
- Richiede ancora una buona dose di lavoro manuale, in quantità proporzionale alla quantità di dati da trattare.

Entrambe richiedono attenzione costante nel tempo, ma per il software è più difficile.

Livelli diversi di riuso

- Riaccedere in lettura ai dati
  - Corretta lettura del media fisico
- Reinterpretare i dat
  - Corretta identificazione delle caratteristiche del formato dati
- Rieseguire le applicazioni di manipolazione
  - Disponibilità e compatibilità (diretta o via emulazione) delle applicazioni originali
- Modificare e aggiornare i dati
  - Conversione dei dati alle applicazioni odierne senza perdita di informazioni
- Realizzare nuove funzionalità sui dati
  - Adattare le vecchie informazioni alle nuove applicazioni in modo da permettere l'esecuzione di nuove funzionalità

Cos’è il markup? (1)

- 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.
- 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.
- Il markup non è soltanto un’inevitabile e sgradevole risultato della informatizzazione dell’arte tipografica. Non è qualcosa che sta con noi a causa dell’informatica.

Cos’è il markup? (2)

- Quando un autore scrive, da millenni a questa parte, specifica anche i delimitatori di parola (chiamati ***spazi***), i delimitatori di frase (chiamati virgole) e i delimitatori di periodo (chiamati punti).
- La numerazione delle pagine o l’uso dei margini per creare effetti sul contenuto sono noti da centinaia di anni.
- Eppure questo a stretto rigore non fa parte del testo, ma del markup: nessuno dirà ad alta voce ‘virgola’ o ‘punto’ nel leggere un testo, ma creerà adeguati comportamenti paralinguistici (espressioni, toni, pause) per migliorare in chi ascolta la comprensione del testo.

Modi del markup: proprietario vs. pubblico

Un formato proprietario è stato creato da una specifica azienda con uno specifico scopo commerciale. L'azienda ne detiene i diritti, e dunque è in grado di modificarlo, aggiornarlo o rivoluzionarlo in qualunque momento e per qualunque motivo.

Un formato pubblico è stato creato da un gruppo di interesse (individui, aziende, enti non commerciali, ecc.) come modello di armonizzazione tra le esigenze di ciascun partecipante.

Il gruppo tipicamente pubblica le specifiche del formato, permettendo a chiunque di realizzare strumenti software per quel formato. A volte questo si concretizza in uno standard ufficiale, avente valore normativo.

Modi del markup: binario vs. leggibile

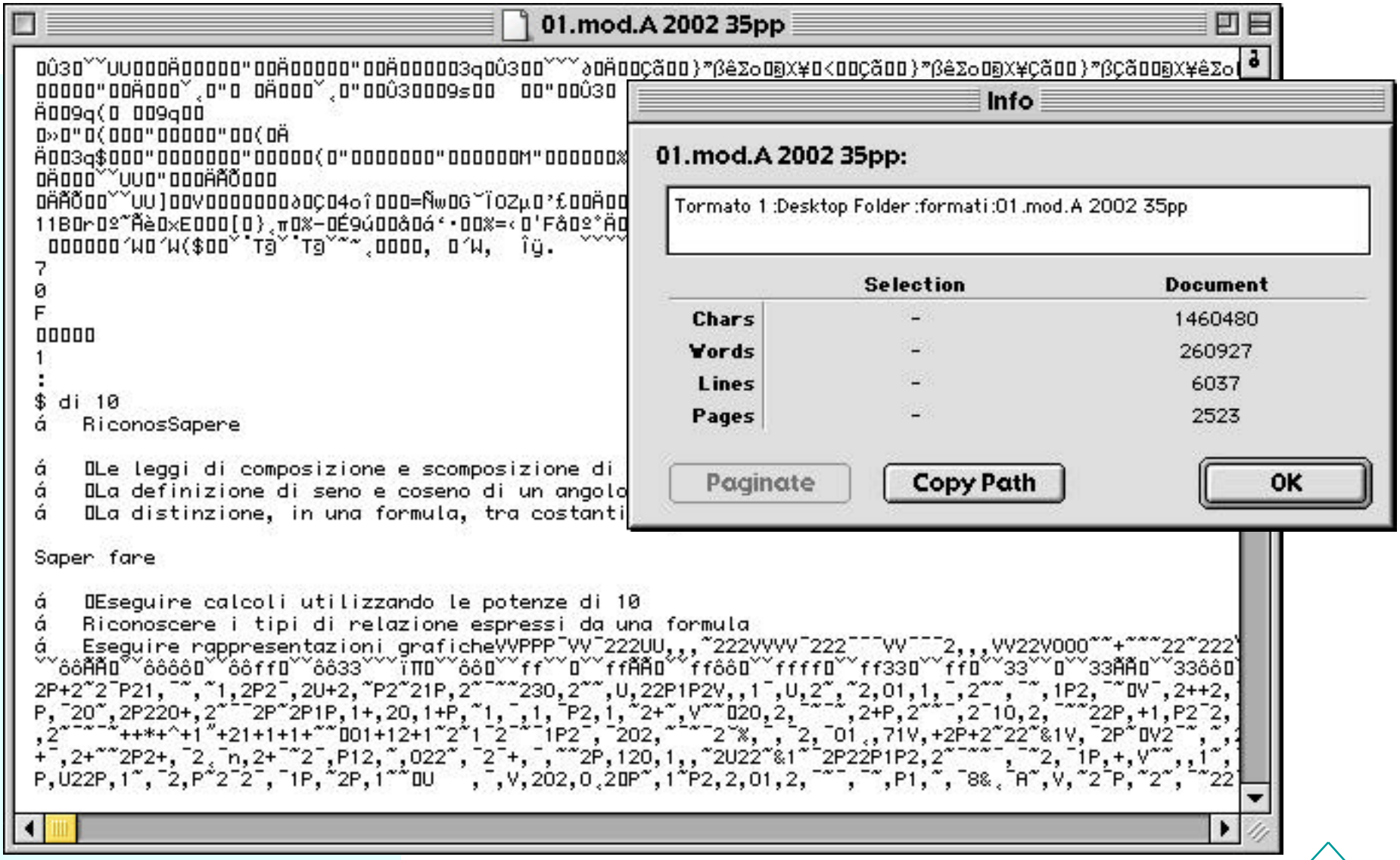
Un formato binario è la memorizzazione esatta delle strutture in memoria dell'applicazione, che niente hanno a che vedere con le esigenze di comprensione di esseri umani. Il testo non è visibile o è visibile per caso.

Un formato leggibile invece è fatto per essere, in casi speciali, letto anche da esseri umani, che possono intervenire per operazioni di emergenza.

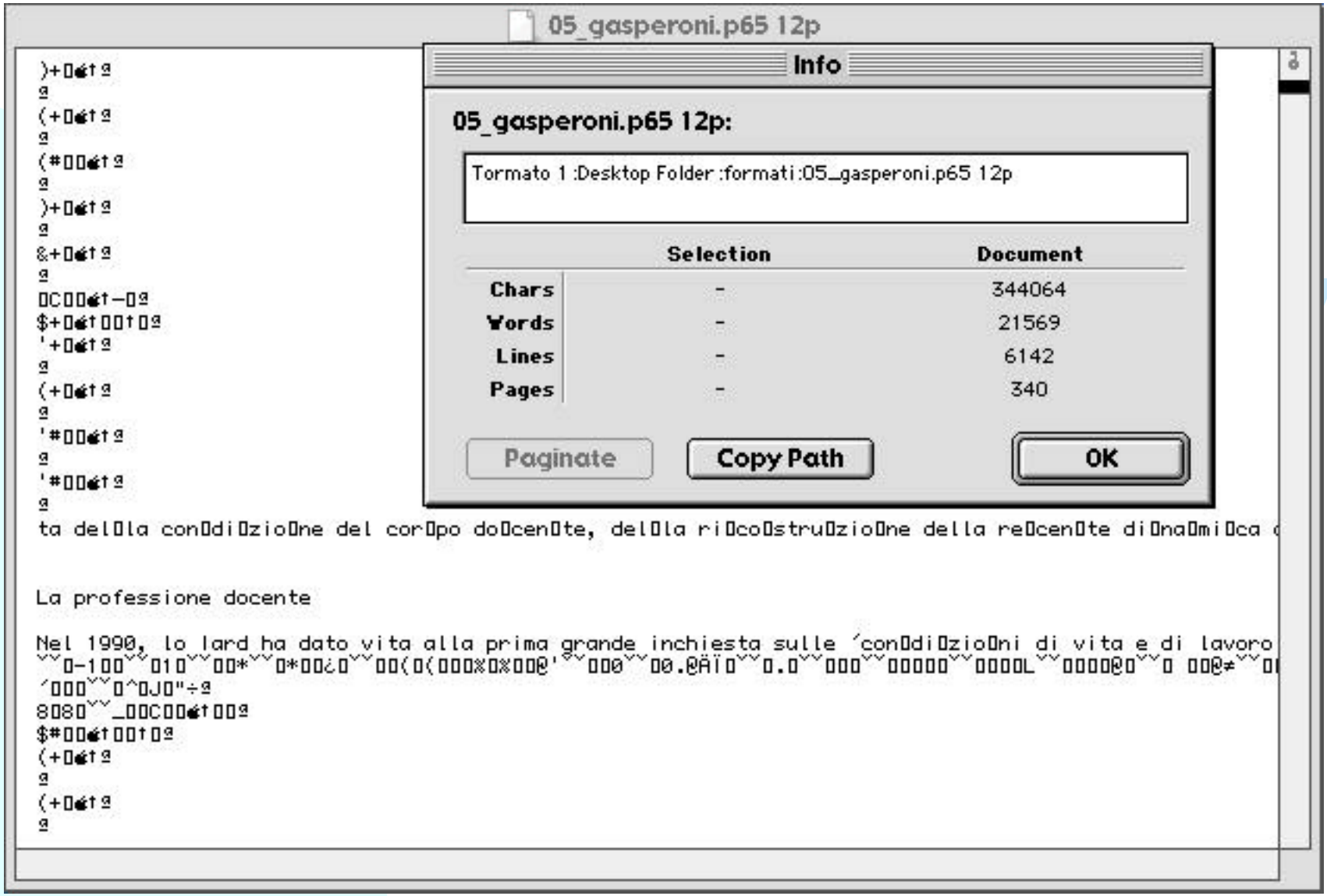
L'applicazione deve trasformare quanto legge in una struttura interna utile per le operazioni di modifica o presentazione. Questa fase si chiama ***parsing***.

Esempio: .doc di MS Word

Esempio: Quark Xpress



Esempio: Adobe PageMaker



Esempio: PDF


Three_men.pdf
%PDF-1.2
%„«”
4 0 obj
<<
/Type /Page
/Parent 5 0 R
/Resources <<
/ProcSet 2 0 R
>>
>>
endobj
9 0 obj
<<
/Length 10 0 R
/Filter /FlateDecode
>>
stream
endstream
endobj
10 0 obj
691
endobj
6 0 obj
<<
/Type /Page
/Parent 5 0 R

Esempio: RTF di MS Word

Three_men.rtf

```
{\creatim\yr2000\mo4\dy9\hr17\min36}{\revtim\yr2000\mo4\dy9\hr17\min36}{\
version2}{\edmins0}{\nofpages3}{\nofwords72}{\nofchars412}{\nofcharsws505}{\
vern117}}\paperw11906\paperh16838\margl1134\margr1134\margt1417\margb1134
\facingp\deftab708\widowctrl\ftnbj\aenddoc\hyphhotz283\margmirror\formshade\
viewkind1\viewscale100\pgbrdrhead\pgbrdrfoot \fet0\sectd
\linex0\headery709\footery709\colsx709\endnhere\sectdefaultcl
\pard\plain \qc\widctlpar\brdrt\brdrtnthsg\brdrw45\brsp20
\brdrl\brdrtnthsg\brdrw45\brsp80 \brdrb\brdrtnthsg\brdrw45\brsp20
\brdrr\brdrtnthsg\brdrw45\brsp80 \adjustright
\lang1033\loch\af4\hich\af4\dbch\f4\cgrid {\page \par \par }{\fs72
\par \par \hich\af0\dbch\af4\loch\f0 Three men in a boat
\par }{\fs48 \hich\af0\dbch\af4\loch\f0 To say nothing of the dog!
\par \par \par \par \par \hich\af0\dbch\af4\loch\f0 Jerome K. Jerome \par
\hich\af0\dbch\af4\loch\f0 FV Books \par }{\fs20
\par \par \par \par }\pard \qc\sl-20\slmult0\widctlpar\adjustright {\caps\fs34
\page
\par }\pard \qc\widctlpar\adjustright {\caps\fs34 \hich\af0\dbch\af4\loch\f0
Chapter 1 \par }\pard \li567\ri851\sb400\sa240\widctlpar\adjustright {\i\fs36
\hich\af0\dbch\af4\loch\f0 \hich\f0 Three invalids - Sufferings of George and
Harris - A victim to one hundred and seven fatal maladies - \u8230\'c9\loch\f0
\par }\pard \qj\widctlpar\adjustright {\fs34 \hich\af0\dbch\af4\loch\f0
T}{\scaps\fs34 \hich\af0\dbch\af4\loch\f0 here}{\fs34 \hich\af0\dbch\af4\loch\f0
were four of us - George, and Willia\hich\af0\dbch\af4\loch\f0
m Samuel Harris, and myself, and Montmorency. We were sitting in my room,
smoking, and talking about how bad we were - bad from a medical point of view I
mean, of course.
\par \hich\af0\dbch\af4\loch\f0 \hich\f0 We were all feeling seedy, and we were
getting quite nervous about it\u8230\'c9
\par }\pard \widctlpar\adjustright {\fs34
\par }}
```


Esempio: PostScript

Three_men.ps

```
119 757 :M
f91 sf
(T)S
162 757 :M
f138 sf
3.525(HERE)A
f91 sf
9.566 .957( were four of us - George, and William Samuel Harris, and)J
119 838 :M
2.105 .211(myself, and )J
470 838 :M
1.978 .198(Montmorency. We were sitting in my room, smoking, and)J
119 920 :M
4.845 .484(talking about how bad we were - bad from a medical point of view I)J
119 1001 :M
(mean, of course.)S
119 1083 :M
6.317 .632(We were all feeling seedy, and we were getting quite nervous about)J
119 1164 :M
<6974C9>S
endp
showpage
%%PageTrailer
%%Trailer
end
%%EOF
```


Esempio: HTML

Three_men.html

```
<HTML>
  <HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
    <TITLE>Three men in a boat</TITLE>
  </HEAD>
  <BODY>
      <P ALIGN="CENTER"> </P>
      <FONT SIZE=7>
         <P ALIGN="CENTER">Three men in a boat</P>
      </FONT>
      <FONT SIZE=6>
         <P ALIGN="CENTER">To say nothing of the dog!</P>
         <P ALIGN="CENTER"> </P>
         <P ALIGN="CENTER">Jerome K. Jerome</P>
         <P ALIGN="CENTER"> </P>
         <P ALIGN="CENTER">FV Books</P>
      </FONT>
      <P ALIGN="CENTER"> </P>
      <FONT SIZE=5>
        <P ALIGN="CENTER">Chapter 1</P>
      </FONT>
      <DIR><DIR>
         <FONT SIZE=5><I>
               <P>Three invalids - Sufferings of George and Harris -
               A victim to one hundred and seven fatal maladies - &#133; </P>
         </I></FONT>
      </DIR></DIR>
      <FONT SIZE=5>
          <P ALIGN="JUSTIFY">There were four of us - George, and William Samuel
           Harris, and myself, and Montmorency. We were sitting in my room,
           smoking, and talking about how bad we were - bad from a medical
           point of view I mean, of course. </P>
          <P ALIGN="JUSTIFY">We were all feeling seedy, and we were getting
           quite nervous about it&#133;</P>
      </FONT>
  </BODY>
</HTML>
```


Esempio: XML

Three_men.xml

```
<?xml version="1.0"?>
<!DOCTYPE NOVEL SYSTEM "novel.dtd">
<NOVEL>
  <FRONT_MATTER>
    <TITLE>Three men in a boat</TITLE>
    <SUBTITLE>To say nothing of the dog!</SUBTITLE>
    <AUTHOR>Jerome K. Jerome</AUTHOR>
    <PUBLISHER>FV Books</PUBLISHER>
  </FRONT_MATTER>
  <CONTENT>
    <CHAPTER>
      <TITLE>Chapter 1</TITLE>
      <SUMMARY>
         <ITEM>Three invalids</ITEM>
         <ITEM>Sufferings of George and Harris</ITEM>
         <ITEM>A victim to one hundred and seven fatal maladies</ITEM>
         ...
      </SUMMARY>
      <BODY>
          <PARA>There were four of us - George, and William Samuel
           Harris, and myself, and Montmorency. We were sitting in my room,
           smoking, and talking about how bad we were - bad from a medical
           point of view I mean, of course. </PARA>
          <PARA>We were all feeling seedy, and we were getting
           quite nervous about it ... </PARA>
      </BODY>
    </CHAPTER>
    <CHAPTER>
      ...
    </CHAPTER>
  </CONTENT>
</NOVEL>
```

Modi del markup: interno vs. esterno

Il markup interno inserisce istruzioni di presentazione all'interno del testo, in mezzo alle parole.

Il markup esterno prevede due blocchi di informazioni: il contenuto e il markup, separati e collegati da meccanismi di indirezione

Il markup interno richiede sintassi particolari per distinguere il markup dal contenuto. Tipicamente si adottano segnalatori particolari che cambiano il tipo di interpretazione del documento. La presenza del carattere segnalatore nel testo richiede l'adozione di tecniche di ***escaping***.

Il markup esterno richiede un meccanismo di indirezione, basato su indirizzi, ***offset*** o identificatori, per associare con correttezza il markup al contenuto.

Modi del markup: procedurale vs. descrittivo

Il markup assolve a diversi ruoli a seconda del sistema di elaborazione, dell'applicazione, dello scopo a cui il documento è soggetto.

- Puntuazionale
- Presentazionale
- Procedurale
- Descrittivo
- Referenziale
- Metamarkup

Markup puntuazionale

- Il markup puntuazionale consiste nell'usare un insieme prefissato di segni per fornire informazioni perlopiù sintattiche sul testo.
- Le regole di punteggiatura sono sostanzialmente stabili, note agli autori, e frequenti nei documenti. Per questo gli autori tipicamente forniscono il loro markup puntuazionale autonomamente.
- Esistono tuttavia notevoli problemi nell'uso della punteggiatura:
  - Incertezze strutturali (virgola, punto e virgola o punto?),
  - Incertezze grafiche (virgolette aperte e chiuse o neutre?),
  - ambiguità procedurali (il punto viene usato sia per segnare la fine di una frase, che l'esistenza di un'abbreviazione, senza contare i tre puntini di sospensione).

Markup presentazionale

- Il markup presentazionale consiste nell'indicare effetti (grafici o altro) per rendere più chiara la presentazione del contenuto.
- Nel testo, possono essere cambi di paragrafo o di pagina, interlinea, pallini per liste, ecc.
- E' altresì markup presentazionale: cambiare pagina all'inizio di una nuova sezione, scrivere "Capitolo 3" in cima alla pagina, ecc.

Markup procedurale

- Il markup procedurale ponsiste nell'indicare con precisione ad un sistema automatico che effetto attivare e che procedura (serie di istruzioni) eseguire nella visualizzazione del contenuto.
- In definitiva, utilizzo le capacità del sistema di presentazione per avere con precisione l'effetto voluto.
- Esempio: ***Wordstar Dot Commands***

  .PL 66
  .MT 6
  .MB 9
  .LH 12

Markup descrittivo

- Il markup descrittivo consiste nell'identificare strutturalmente il tipo di ogni elemento del contenuto.
- Invece di specificare effetti grafici come l'allineamento o l'interlinea, ne individuo il ruolo all'interno del documento, specificando che un elemento è un titolo, un paragrafo, o una citazione.

Markup referenziale

- Il markup referenziale consiste nel fare riferimento ad entità esterne al documento per fornire significato o effetto grafico ad elementi del documento. Per esempio, utilizzare una sigla nota che venga poi sostituita dalla parola intera durante la stampa
- Es.: l'autore scrive "CdL" e il sistema trasforma automaticamente l'input in "Corso di Laurea"

Metamarkup

- Il metamarkup consiste nel fornire regole di interpretazione del markup e permette di estendere o controllare il significato del markup.
- Ad esempio, la possibilità di definire macro per l'interpretazione e la visualizzazione del documento, o il processo di definizione degli elementi e delle procedure valide di un documento.

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

Aggiungiamo markup puntuazionale e presentazionale: maiuscole/minuscole, punteggiatura, spazi e ritorni a capo sono essi stessi elementi di markup.

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. Questo è RTF.

{\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 descrittivo

Sono informazioni (descrizioni) 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>
```


Il 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

Il markup procedurale (2)

- Associato agli individui
  - ogni elemento possiede le proprie procedure per la visualizzazione, che possono anche essere tutte diverse anche per elementi dello stesso tipo.
- 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.
  - Inoltre, e molto importante, non è possibile porre vincoli sulla "correttezza" di un documento.

Il 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 (per esempio) un "titolo" o un "elenco" o un "paragrafo" non dipende dalla sofisticazione del sistema di visualizzazione.

Il 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.
- Ancora, è possibile specificare ***regole di correttezza*** sui documenti, ad esempio che ad un'immagine segua **necessariamente** una didascalia, ecc.

Il markup dichiarativo (3)

Il markup generico sfrutta il ***late binding***: le scelte specifiche si fanno all'ultimo momento utile.

Nel nostro caso, la formattazione viene decisa nel momento in cui il documento viene visualizzato, o stampato, piuttosto che quando il documento viene codificato.

Per contro, il markup dichiarativo richiede l'esistenza di due passaggi distinti, la marcatura e la formattazione, che usano due tecnologie diverse e vengono fatte in due momenti diversi.

Il markup dichiarativo (4)

Ci sono vari vantaggi in questa tecnica:

- **Facilità nella creazione**: l'autore si concentra sul ruolo organizzativo delle singole parti di testo, piuttosto che sul loro aspetto stampato.
- **Indipendenza dalla formattazione**: riformattare un documento secondo nuove regole richiede semplicemente di ricodificare dei parametri esterni, non di modificare in alcuna maniera il documento
- **Flessibilità**: riusare un documento in un nuovo contesto e' facile, perché non è necessario rimuovere la vecchia informazione per far posto alla nuova.
- **Visioni di documenti dinamicamente riconfigurabili**: è possibile evidenziare di volta in volta caratteristiche del documento diverse (caratteristica degli outline processor, per esempio)

Modi del markup (2)

Tradizionalmente, editor (sistemi di inserimento dati) e formatter (sistemi di creazione di output) erano separati. LaTeX ancora riflette questa distinzione, MS Word no.

Il markup viene utilizzato da un formatter per la creazione dell'output, ma l'editor può operare sul markup se ne conosce le caratteristiche. In questo caso il markup sarà:

- **Esposto**: il sistema mostra il markup e ne mostra gli effetti
- **Travestito**: il sistema mostra nel contenuto un simbolo che attiva il markup
- **Nascosto**: il sistema nasconde il markup e ne mostra solo gli effetti
- **Visualizzato**: il sistema non interpreta in markup e lo mostra insieme al contenuto.

Uso del markup (su computer)

- 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.

La storia del markup (1)

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.

La storia del markup (2)

Anni ‘80: WP WYSIWYG e DTP: un passo indietro?

- Si diffondono i primi WP: 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.).

La storia del markup (3)

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, URI, HTTP e HTML
- 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-Namespaces, X-Pointer, X-Link, XSL, ecc.).

TROFF/NROFF (1)

- Nato nel 1973, fa parte della distribuzione Unix standard. Sotto Linux si chiama Groff
- E' ancora usato per documentazione tecnica, in particolare i manuali on-line di Unix
- Esiste un formatter che è in grado di creare documenti stampabili sia su stampante (**troff**) che su schermo a carattere (**nroff**).
- I comandi sono o esterni (compaiono su righe autonome precedute da un punto) o interni (introdotte dal carattere di escape "\")
- Permette la definizione e l'uso di quattro tipi di carattere: roman, italic, bold e symbol.
- Permette la creazione di macro complesse (il nome è al massimo di due caratteri, però)

TROFF /NROFF (2)

```
.\"  Questo è un esempio Troff.
.\"  margine sinistro 4 cm.
.\"  ampiezza del testo 8 cm.
.po 4c
.ll 8c
.\"  Inizia il documento.
.ft B
1. Introduzione a Troff

.ft P
Questo \(`e un esempio di documento s
essere elaborato con Troff.
In questo caso, si presume che verr\
``\fBs\fP'' (con l'opzione \fB\-ms\f

.ft B
1.1 Paragrafi

.ft P
Il testo di un paragrafo termina quando nel sorgente viene incontrata
una riga vuota.

Per la precisione, gli spazi verticali vengono rispettati, per cui le
righe vuote si traducono in spazi tra i paragrafi, anche quando
queste sono pi\(`u di una.
```

1. Introduzione a Troff

Questo è un esempio di documento scritto in modo tale da poter essere elaborato con Troff. In questo caso, si presume che verrà utilizzata lo stile "**s**" (con l'opzione **–ms**).

1.1 Paragrafi

Il testo di un paragrafo termina quando nel sorgente viene incontrata una riga vuota.

Per la precisione, gli spazi verticali vengono rispettati, per cui le righe vuote si traducono in spazi tra i paragrafi, anche quando queste sono più di una.

Questo è l'inizio di un nuovo paragrafo dopo tre righe vuote di separazione.

TeX e LaTeX (1)

Realizzato agli inizi degli anni 80 da Donald Knuth.

Linguaggio di programmazione completo, dotato di comandi per la formattazione di testi (circa 300 comandi fondamentali, detti ***primitive***).

Di notevole complessità, è rivolta unicamente a programmatori e tipografi molto sofisticati.

Metafont è un sistema associato a TeX per la descrizione delle forme dei caratteri di un font attraverso formule matematiche.

TeX permette la generazione di macro che semplificano notevolmente la generazione di testi particolarmente sofisticati.

Esistono librerie di macro che permettono di scrivere documenti arbitariamente complessi: matematica, chimica, musica, grafica vettoriale, grafica bitmap, ecc.

TeX e LaTeX (2)

Il formatter TeX prende in input un documento TeX e genera un documento in formato DVI (DeVice Independent), che rappresenta una formulazione generica dell'aspetto grafico del documento.

Appositi programmi convertono poi il DVI in un formato utile per la stampante (ad esempio il PostScript).

Nel 1985 Leslie Lamport sviluppò LaTeX, una raccolta di macro TeX per la generazione di una ventina circa di tipi di documento particolarmente comuni: articolo, libro, lettera, annuncio, ecc.

TeX e LaTeX (3)

```
documentclass{article}

% Inizia il preambolo.

\setlength{\textwidth}{7cm}
\setlength{\textheight}{7cm}

% Fine del preambolo.

\begin{document}

% Inizia il documento vero e proprio.

\section{Introduzione a TeX/LaTeX}

Questo \`e un esempio di documento scritto con LaTeX.
Come si pu\`o vedere \`e gi\`a stato definito uno stile
generale del documento: article.

\subsection{Gli ambienti}

LaTeX utilizza gli ambienti per definire dei comportamenti
circoscritti a zone particolari del testo.
Per esempio, la centratura si ottiene utilizzando l'ambiente
center.

\begin{center}
Questo \`e un esempio di testo centrato.
\end{center}

% Fine del documento.
```


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.

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.

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.

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.

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.

Markup dichiarativo

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.

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.

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.

I documenti SGML

Un documento in un linguaggio di markup definito sulla base di SGML è sempre composto delle seguenti tre parti:

- Dichiarazione SGML
- DTD
- Istanza del documento

Un esempio di SGML

```
<!SGML “ISO 8879:1986”  ...>
```

```
<!DOCTYPE NOVEL [
  <!ELEMENT NOVEL   (FRONT,CONTENT) >
  <!ELEMENT FRONT (TITLE, SUBTITLE?, AUTHOR)>
  <!ELEMENT CONTENT (CHAPTER)+ >
  <!ELEMENT CHAPTER (TITLE, PARA+)>
  <!ELEMENT TITLE   #PCDATA >
  <!ELEMENT SUBTITLE    #PCDATA >
  <!ELEMENT AUTHOR   #PCDATA >
  <!ELEMENT PARA    #PCDATA >
]>
```

```
<NOVEL>
  <FRONT>
    <TITLE>Three men in a boat</TITLE>
    <SUBTITLE>To say nothing of the dog!</SUBTITLE>
    <AUTHOR>Jerome K. Jerome</AUTHOR>
  </FRONT>
  <CONTENT>
    <CHAPTER>
      <TITLE>Chapter 1</TITLE>
      <PARA>There were four of us ... </PARA>
      <PARA>We were all feeling ... </PARA>
    </CHAPTER>
</CONTENT>
</NOVEL>
```


SGML Declaration

```
<!SGML “ISO 8879:1986” car. spec.>
```

- La dichiarazione SGML contiene le istruzioni di partenza delle applicazioni SGML.
- Essa permette di specificare valori fondamentali come la lunghezza dei nomi degli elementi, il set di caratteri usati, le specifiche caratteristiche di minimizzazione ammesse, ecc.)
- Una dichiarazione SGML è lunga varie centinaia di righe.
- Non è obbligatoria. Se è assente, viene usata una dichiarazione di default detta “***Reference Concrete Syntax***”.
- La RCS definisce lunghezze e sintassi standard (come l'uso del carattere "<" per indicare l'inizio del tag).

Document Type Declaration

```
<!DOCTYPE nome TIPO [markup] >
```

- La dichiarazione del tipo del documento serve a specificare le regole che permettono di verificare la correttezza strutturale di un documento.
- Vengono cioè elencati ***[i file che contengono]*** gli elementi ammissibili, il contesto in cui possono apparire, ed altri eventuali vincoli strutturali.
- Nella terminologia SGML, si parla di modellare una classe (cioè una collezione omogenea) di documenti attribuendogli un tipo.

La Document Instance

L’istanza del documento è quella parte del documento che contiene il testo vero e proprio, dotato del markup appropriato.

Esso contiene una collezione di elementi (tag), attributi, entità, PCDATA, commenti, ecc.

Le applicazioni SGML sono in grado di verificare se l’istanza del documento segue le regole specificate nel DTD, e di identificare le violazioni.

I componenti del markup

Un documento con markup di derivazione SGML (inclusi HTML, XML, ecc.) contiene una varietà dei seguenti componenti

- Elementi
- Attributi
- Entità
- Testo (detto anche #PCDATA)
- Commenti
- Processing Instructions

Elementi, Attributi, Entità, #PCDATA, Commenti, Processing Instructions

- Gli elementi sono le parti di documento dotate di un senso proprio.
- Il titolo, l'autore, i paragrafi del documento sono tutti elementi.
- Un elemento è individuato da un tag iniziale, un contenuto ed un tag finale.
- **Non confondere i tag con gli elementi!**

```
<TITOLO>Tre uomini in barca</TITOLO>
```


Elementi, **Attributi**, Entità, #PCDATA, Commenti, Processing Instructions

- Gli attributi sono informazioni aggiuntive sull'elemento che non fanno effettivamente parte del contenuto (meta-informazioni).
- Essi sono posti dentro al tag iniziale dell'elemento. Tipicamente hanno la forma nome="valore"

```
<romanzo file="threemen.sgm">…</romanzo>
<capitolo N="1">Capitolo primo</capitolo>
```


Elementi, Attributi, **Entità**, #PCDATA, Commenti, Processing Instructions

- Le entità sono frammenti di documento memorizzati separatamente e richiamabili all'interno del documento.
- Esse permettono di riutilizzare lo stesso frammento in molte posizioni garantendo sempre l'esatta corrispondenza dei dati, e permettendo una loro modifica semplificata.

```
Oggi &egrave; una bella giornata.
Come dice &FV;: "divertitevi!"
```


Elementi, Attributi, Entità, **#PCDATA**, Commenti, Processing Instructions

- Rappresenta il contenuto vero e proprio del documento.
- Esso corrisponde alle parole, gli spazi e la punteggiatura che costituiscono il testo.
- Viene anche detto **#PCDATA** (Parsed Character DATA) perché i linguaggi di markup definiscono ***character data*** (CDATA) il contenuto testuale vero e proprio, e quello degli elementi è soggetto ad azione di parsing (perlopiù per identificare e sostituire le entità).

Elementi, Attributi, Entità, #PCDATA, **Commenti**, Processing Instructions

- I documenti di markup possono contenere commenti, ovvero note da un autore all'altro, da un editore all'altro, ecc.
- Queste note non fanno parte del contenuto del documento, e le applicazioni di markup li ignorano.
- Sono molto comodi per passare informazioni tra un autore e l'altro, o per trattenere informazioni per se stessi, nel caso le dimenticassimo.

```
<!-- Questo è ignorato dal parser -->
```


Elementi, Attributi, Entità, #PCDATA, Commenti, **Processing Instructions**

- Le **processing instructions** (PI) sono elementi particolari (spesso di senso esplicitamente procedurale) posti dall'autore o dall'applicazione per dare ulteriori indicazioni su come gestire il documento XML nel caso specifico
- Per esempio, in generale è l'applicazione a decidere quando cambiare pagina. Ma in alcuni casi può essere importante specificare un comando di cambio pagina (oppure tutti i cambi pagina di un documento già impaginato).

<?NEWPAGE?>

Conclusioni

Oggi abbiamo parlato di

- Importanza del markup nel testo
- Rilevanza del markup dichiarativo per applicare scopi multipli allo stesso testo
- Qualche distinzione tra tipi di markup
- Un po’ di storia del markup
- Cos'è SGML e di che cosa è fatto

Riferimenti

- ***J. H. Coombs, A. H. Renear, S. J. DeRose, Markup Systems and the future of Scholarly Text Processing, Communications of the ACM, 30(11), November 1987.***
- "I.2 A Gentle Introduction to SGML", in C.M. Sperberg-McQueen and L. Burnard (eds.), ***Guidelines for Electronic Text Encoding and Interchange***, 1994, http://etext.virginia.edu/TEI.html
- E. Maler, J. El Andaloussi, ***Developing SGML DTDs***, Prentice Hall, 1996