

Corso di informatica umanistica

Prof. Pierluigi Feliciati

MODULO 3

- La codificazione della struttura: XML
- Uno schema XML all'opera: *Dublin Core*
- Elaborare testi: lo standard TEI
- La digitalizzazione e i metadati gestionali

Elaborare testi: XML

Quache parola in più su XML

abbiamo accennato qualcosa sui linguaggi testuali (MIME type:text) di marcatura (la famiglia dei -ML) e sulla differenza tra HTML, XHTML e XML.

Vediamo ora con più attenzione **XML**, **eXtensible Markup Language**, base di tutti i progetti attuali di trattamento di testi (e non solo) in ambito umanistico.

La sua versione 1.0 risale al 1998 e la versione italiana del linguaggio è in

<http://www.xml.it:23456/XML/REC-xml-19980210-it.html>, la seconda specifica del 2000, la terza e la versione 1.1 (compatibile con Unicode 4) del 2004

<http://www.w3.org/TR/REC-xml>

Dai 4 aspetti della pagina web...

- Come ricorderete, abbiamo accennato in apertura del modulo 2, che nella pagina web possiamo individuare 4 aspetti: CONTENUTO, STRUTTURA, PRESENTAZIONE e COMPORTAMENTO
- Con l'esercitazione HTML abbiamo sperimentato un linguaggio di marcatura che ci permettesse di coprire tutti insieme i primi tre aspetti, pur sapendo che la presentazione deve essere gestita a parte, con i fogli di stile
- Ora ci concentreremo su un linguaggio DI MARCATURA che consente SOLO di STRUTTURARE CONTENUTI (non solo per il Web) e non serve a fare presentazioni: **XML**

XML

oltre HTML: XML

XML, acronimo di **eXtensible Markup Language** è un metalinguaggio creato e gestito dal World Wide Web Consortium (W3C). È una semplificazione e adattamento dell'**SGML**, da cui è nato nel 1998, e permette di definire la grammatica di diversi linguaggi specifici derivati.

Rispetto all'HTML, l'XML ha uno scopo ben diverso: **XML è un linguaggio utile allo scambio dei dati, quindi di back-office e non di front-office, o di (rap)presentazione che dir si voglia.** L'XML è oggi molto utilizzato anche per l'esportazione di dati tra DBMS.

La sua versione 1.0 risale al 1998 e la versione italiana del linguaggio è in <http://www.xml.it:23456/XML/REC-xml-19980210-it.html>, la seconda specifica del 2000, la terza e la versione 1.1 (compatibile con Unicode 4) del 2004 <http://www.w3.org/TR/REC-xml>

XML

Il World Wide Web Consortium (W3C), in seguito alla cd. *guerra dei browser* (nei tardi anni '90 Microsoft e Netscape introducevano, con ogni nuova versione del proprio browser, un'estensione proprietaria all'HTML), fu costretto a scegliere **quali caratteristiche standardizzare e quali lasciare fuori** dalle specifiche dell'HTML. Fu in questo contesto che iniziò a delinearsi la necessità di **un linguaggio di markup che desse maggiore libertà nella definizione dei tag**, pur rimanendo all'interno di uno standard.

Il "**progetto XML**" suscitò un così forte interesse a tal punto che si creò un XML Working Group, composto da esperti mondiali delle tecnologie SGML, ed una commissione, XML Editorial Review Board, deputata alla redazione delle specifiche del progetto. Nel febbraio del 1998 le specifiche divennero una raccomandazione ufficiale con il nome di *Extensible Mark-up Language, versione 1.0*. Ben presto ci si accorse che XML **non era solo limitato al contesto web**, ma era qualcosa di più: uno strumento che permetteva di essere utilizzato nei più diversi contesti, dalla definizione della struttura di documenti, allo scambio delle informazioni tra sistemi diversi, dalla rappresentazione di immagini alla definizione di formati di dati. (da Wikipedia)

XML

ecco un esempio tipico di file XML. Lo si può visualizzare (così com'è e basta...) con un browser qualsiasi semplicemente salvandolo in un **file di testo** con estensione **.xml**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<utenti>
  <utente>
    <nome>Luca</nome>
    <cognome>Ruggiero</cognome>
  </utente>
  <utente>
    <nome>Max</nome>
    <cognome>Rossi</cognome>
  </utente>
</utenti>
</xml>
```

XML

La struttura vera e propria è composta dai **tag** (**elementi, marcatori**) **creati dallo sviluppatore**, che hanno essenzialmente due caratteristiche:

- devono essere comprensibili in funzione dello scopo (onde evitare di non capirci nulla ad una seconda lettura e per facilitare la comprensione agli altri utenti);
- devono rispettare delle regole, come la differenza tra maiuscolo e minuscolo, non possono iniziare con numeri o caratteri speciali e non possono contenere spazi.

Ogni **record**, se si vuol pensare ad XML come ad un **database**, viene chiamato **nodo** ed ogni tag può essere corredato da **attributi**, come per HTML.

Ad esempio, la struttura precedente può essere modificata come segue:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<utenti>
  <utente nome="Luca" cognome="Ruggiero" />
  <utente nome="Max" cognome="Rossi" />
</utenti>
</xml>
```

I tag che non hanno l'omonimo tag di chiusura vanno chiusi con uno spazio più uno **slash (/)** finale, prima della fine del tag stesso.

XML

Un file XML dunque **non ha interfaccia propria** ma è accessibile da qualsiasi linguaggio di programmazione Web (sia lato server che lato client).

Il rispetto delle sue *semplici regole* ed il fatto che *i dati sono accessibili semplicemente insieme alla loro struttura* a qualsiasi tipo di applicazione, locale o remota, fanno dell'XML **uno standard** che si caratterizza per la sua **portabilità**.

L'accesso ad una struttura XML può avvenire attraverso diversi sistemi, il più comune ma rudimentale dei quali è lo sfruttamento del *filesystem* (la gestione dei file in memoria, il gestione risorse di Windows) del server su cui gira il motore del linguaggio in uso.

La semplicità, adattabilità e piena compatibilità di XML con gli altri linguaggi di marcatura, la quasi totale mancanza di caratteristiche opzionali, la leggibilità umana e la facilità per le macchine di elaborarlo ne hanno decretato il successo.

Lo sviluppo del linguaggio DTD prima, di XSD poi, l'adozione dei *NameSpace* (per la definizione delle strutture dati e il richiamo di diverse regole in uno steso file) e l'uso degli *stylesheet* (per la visualizzazione) hanno sostenuto questo percorso. L'unico difetto – volendo - è la poca economicità in termini di spazio di occupazione, dovuta alla ridondanza di elementi oltre ai dati veri e propri.

Sintassi XML

Le regole sintattiche di Xml sono poche:

- Ogni documento deve presentare **un prologo** per la dichiarazione della specifica Xml che si usa, le eventuali informazioni di codifica e il richiamo esplicito alla DTD in uso

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

Questa prima riga, racchiusa tra i simboli `<?` e `?>` si dice *processing instruction* e dalla versione 1.1 di XML è divenuta obbligatoria. Se omessa, il documento diventerà automaticamente 1.0

Sintassi XML

Il contenuto di un documento XML deve corrispondere a una gerarchia di **tag** (o **elementi**) **non sovrapposti**, quindi i tag che si aprono per primi debbono chiudersi per primi.

Per elemento/tag (marcatore) si intende una coppia
<parola>(apertura) e </parola> (chiusura) con parola=nome del tag

<DOCUMENTO>

<TITOLO>Titolo del documento</TITOLO>

<TESTO>

<PARAGRAFO>testo del primo paragrafo</PARAGRAFO>

<PARAGRAFO>testo del secondo paragrafo</PARAGRAFO>

</TESTO>

</DOCUMENTO>

Sintassi XML

- Deve esistere un tag che contenga tutti gli altri, che si dice **tag radice** o **root**.
- Tutti i tag devono **sempre essere aperti e chiusi** e se non contengono testo, per economia, possono essere presentati usando la cd. *chiusura implicita*:

<TAGVUOTO/> equivale a <TAGVUOTO></TAGVUOTO>

- I nomi dei tag sono **a nostro piacere**, rispettando però alcune regole:
 - Devono iniziare con un carattere o con un _ (*underscore*), mai con numeri, possono contenere un numero qualsiasi di lettere, numeri, trattini, punti e underscore
 - Non possono contenere spazi e sono case-sensitive: <TAG> e <tag> sono diversi, i tag di apertura e di chiusura devono essere uguali

Sintassi XML

- Possono essere associati **attributi** agli elementi (tag), rappresentandoli nel tag di apertura, racchiudendone **il valore tra apici o virgolette** e non ripetendoli più di una volta:

```
<PREZZO valuta='euro'>15</PREZZO>
```

```
<PREZZO valuta="euro">15</PREZZO>
```
- Per i nomi degli attributi valgono le stesse regole degli elementi
- Esistono ***attributi riservati*** che possono essere usati solo rispettandone il significato previsto, ad es.

```
xml:lang="it | en | fr ..."
```

 indica la lingua del tag

```
xml:space="preserve | default"
```

 indica se gli spazi presenti nel contenuto devono essere mantenuti o considerati superflui (quindi ev. eliminati)

Sintassi XML

- Nel caso sia necessario usare **all'interno del contenuto caratteri speciali** (come <, >, &, " etc.) si dovrà usare il cd. **Entity-name** corrispondente, composto da un &, da un nome e da un punto-e-virgola finale, ad es.

& corrisponde a &

< corrisponde a <

> corrisponde a >

" corrisponde a "

- Se si deve inserire nel contenuto una porzione di testo **scritta in un codice diverso** (ad es. HTML, JavaScript), si dovrà usare la tipologia di contenuto **CDATA section**

```
<contenuto>
  <![CDATA[
    <html><body> .... </body></html>
  ]]>
</contenuto>
```

XML e *namespace*

Il rispetto della sintassi XML garantisce **documenti *well-formed*** (ben costruiti), leggibili da qualunque software su qualunque piattaforma hardware. I browser visualizzano i file XML ed evidenziano se ci sono errori.

I software che leggono i documenti XML si dicono ***parser*** perché scandiscono riga per riga il codice e lo interpretano.

A volte è necessario poter usare dentro al proprio documento XML alcuni elementi che hanno **una particolare provenienza**, vale a dire che sono **già stati definiti in un qualche schema** di codifica. Lo strumento per andare incontro a questa esigenza è chiamato ***namespace***.

XML e *namespace*

Un *namespace* è indicato da un prefisso distintivo.

Per usarlo è necessario **dichiararlo** all'interno di un qualsiasi elemento di un documento XML, usando un attributo standard, **xmlns** (abbreviazione di XML NameSpace) seguito da un due punti, dal prefisso distintivo del namespace da adottare e dalla URL dove si trova lo schema cui il namespace fa riferimento.

Se, per esempio, vogliamo usare alcuni elementi definiti dallo schema *Dublin Core* per codificare una citazione bibliografica, basterà usare il **namespace dc** nel modo che segue:

```
<citazione xmlns:dc="http://purl.org/dc/elements/1.1/">  
  <dc:creator>Carlo Dionisotti</dc:creator>  
  <dc:title>Geografia e storia della letteratura italiana  
  </dc:title>  
  <dc:publisher>Einaudi</dc:publisher>  
</citazione>
```

XML e *namespace*

..che è più funzionale, internazionale e *interoperabile* rispetto a questo codice:

<citazione>

<autore>Carlo Dionisotti</autore>

<titolo>Geografia e storia della letteratura italiana</titolo>

<editore>Einaudi</editore>

</citazione>

perché si adotta **uno standard usato da molti** e che quindi molti (intendendo soprattutto le macchine) possono decodificare facilmente.

XML: definizione degli schemi

Ci sono diversi modi per **dichiarare uno schema di codifica**, vale a dire la grammatica che stabilisce quali elementi sarà possibile usare all'interno di un documento XML e in che modo tali elementi si potranno relazionare fra di loro.

Il modo più conosciuto per dichiarare uno schema di codifica è attraverso una **Document Type Definition** (o **DTD**), una sintassi ereditata direttamente dal linguaggio SGML, da cui l'XML deriva. La DTD è costituita da una sintassi estremamente semplice, ma con varie limitazioni.

Recentemente a questo metodo si sono aggiunti alcuni linguaggi per scrivere **XML Schema** (o **XSD**). Lo schema più usato è appunto quello del W3C

<http://www.w3.org/TR/xmlschema-0/>.

Dublin Core in XML

- Fin dal 2003 la *Dublin Core Metadata Initiative* ha prodotto una raccomandazione per l'implementazione di dichiarazioni DC (cioè di **metadati descrittivi**) usando il linguaggio XML
<http://www.dublincore.org/documents/dc-xml-guidelines/>
- Le raccomandazioni sono queste:
 1. Le applicazioni XML devono essere basate su [XMLSCHEMA] e non su XML DTD
 2. L'uso dei XML Namespaces [XMLNS] deve servire a identificare univocamente gli elementi DC, gli eventuali *refinements* e gli schemi di codifica

Dublin Core in XML

- **Raccomandazione 3:** gli sviluppatori dovrebbero codificare le proprietà di una risorsa come elementi XML e i valori come il contenuto di tali elementi. Il nome dell'elemento XML dovrebbe essere un nome XML completo che associa il nome dell'elemento con il nome appropriato DCMI namespace. Ad esempio, si deve usare

`<dc:title> Dublin Core in XML </ dc: title>`

piuttosto che

`<dc:title value="Dublin Core in XML" />`

Dublin Core in XML

- **Raccomandazione 4:** I nomi delle proprietà per i 15 elementi DC dovrebbero essere tutti espressi in minuscolo.

Ad esempio, utilizzare

```
<dc:title> Dublin Core in XML </ dc: title>
```

piuttosto che

```
<dc:Title> Dublin Core in XML </ dc: title>
```

Dublin Core in XML

- **Raccomandazione 5:** I valori delle proprietà multiple dovrebbero essere codificati ripetendo l'elemento XML che serve a esprimere quella proprietà.
- Per esempio:

<dc:title> Primo titolo </ dc: title>

<dc:title> secondo titolo </ dc: title>

Dublin core in XML: esempio

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<metadata xmlns="http://example.org/myapp/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://example.org/myapp/
http://example.org/myapp/schema.xsd"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <!-- questa parte è servita a dichiarare quale schema si adotta e che si
userà il namespace di Dublin Core →
  <dc:title> UKOLN </dc:title>
  <dc:description> UKOLN is a national focus of expertise in digital information
management. It provides policy, research and awareness services
to the UK library, information and cultural heritage communities. UKOLN is
based at the University of Bath. </dc:description>
  <dc:publisher> UKOLN, University of Bath </dc:publisher>
  <dc:identifier> http://www.ukoln.ac.uk/ </dc:identifier>
  .....
</metadata>
```

Elaborare testi: il progetto TEI

La ***Text Encoding Initiative*** (TEI) è uno standard internazionale nato per permettere alle biblioteche, ai musei, alle case editrici ed a tutti gli studiosi di rappresentare fedelmente qualsiasi testo in formato digitale. Lo standard è stato creato soprattutto per i testi letterari del passato, al fine di poterli conservare per il futuro. Ma non solo: i testi codificati in questo modo sono:

- portabili;
- facilmente archiviabili;
- facilmente gestibili con strumenti informatici.

Lo standard TEI (espresso tramite uno schema XML) è gestito da un consorzio di istituzioni internazionali di ambito linguistico e letterario (l'Association for Computers and the Humanities l'Association for Computational Linguistics e l'Association for Literary and Linguistic Computing).

Elaborare testi: il progetto TEI

Lo scopo della TEI è dunque quello di definire **uno standard di codifica specifico per i testi letterari**, e di **normalizzare i formati di memorizzazione dell'informazione testuale**.

Per realizzare questo progetto è stato assunto come linguaggio di base l'SGML e successivamente la sua evoluzione, l'XML; come insieme di caratteri è stato adottato l'ISO 646.

Dal 1989, dopo un'accurata analisi condotta su molti testi letterari, è stata definita una vasta e complessa DTD di TEI, le cui caratteristiche sono state pubblicate nel 1994 (*Guidelines for Electronic Text Encoding and Interchange* o TEI P3).

Elaborare testi: il progetto TEI

La versione corrente delle TEI *Guidelines* (P5 Guidelines) è stata ufficialmente pubblicata il 1 novembre 2007.

Data la complessità del linguaggio di codifica elaborato, la TEI ne ha reso disponibile una versione "ristretta" (*Lite*) con un manuale introduttivo, ***TEI Lite. An Introduction to Text Encoding for Interchange***

<http://www.tei-c.org/Guidelines/Customization/Lite/index.xml>, di cui esiste la traduzione italiana *TEI Lite: Introduzione alla codifica dei testi*,

http://www.tei-c.org/Guidelines/Customization/Lite/teiu5_it.xml su cui ci baseremo per questa breve presentazione.

Il manuale è anche stato pubblicato a stampa: *Il manuale TEI Lite. Introduzione alla codifica elettronica dei testi letterari*, a cura di Fabio Ciotti, Edizioni Sylvestre Bonnard, Milano 2005.

Elaborare testi: il progetto TEI

Una **codifica rende esplicite solo quelle caratteristiche testuali importanti per il codificatore**. Oltre alla marcatura delle frasi e dei paragrafi, ad esempio:

- potrebbero essere aggiunte note a piè di pagina, commenti o glosse su qualsiasi brano;
- potrebbero essere aggiunti puntatori che colleghino le parti di questo testo ad altri testi;
- i nomi propri di vario tipo potrebbero essere distinti dal testo circostante;
- dettagli bibliografici sulla provenienza e sul contesto del testo potrebbero precederlo;
- potrebbe essere fornita un'analisi linguistica del brano in frasi, proposizioni, parole etc., associando ogni unità con appropriati codici di classificazione;
- il testo potrebbe essere segmentato in unità narrative o discorsive;
- potrebbe essere inclusa nella codifica un'analisi sistematica o un'interpretazione del testo mediante un complesso allineamento o collegamento tra il testo e l'analisi, o tra il testo e una o più sue traduzioni;
- brani del testo potrebbero essere collegati a immagini e suoni gestiti da altri media.

Elaborare testi: lo schema TEI *lite*

Struttura di un testo TEI

Tutti i testi conformi alla TEI contengono:

- (a) una **intestazione** (codificata come elemento <teiHeader>)
- (b) la **trascrizione del testo** vero e proprio (marcata con l'elemento <text>).

L'intestazione TEI contiene informazioni analoghe a quelle contenute nel frontespizio di un testo a stampa. Essa può avere fino a quattro parti: una **descrizione bibliografica** del testo memorizzato su supporto digitale, una descrizione del modo nel quale è stato **codificato**, una descrizione non bibliografica del testo (un **profilo del testo**), e un **elenco delle revisioni**.

L'intestazione è insomma un *frontespizio elettronico*, ben più ricco di informazioni di un frontespizio a stampa.

Elaborare testi: lo schema TEI *lite*

Struttura di un testo TEI – la codifica della trascrizione

Un testo TEI può essere **unitario** (un'opera singola) o **composito** (una collezione di opere, come un'antologia).

In entrambi i casi, il testo può avere un peritesto iniziale o un peritesto finale facoltativi. In mezzo c'è il corpo del testo, che, nel caso di un testo composito, può essere costituito da gruppi, ciascuno dei quali può a sua volta contenere più gruppi o testi.

Un testo unitario sarà codificato usando una struttura generale come questa:

```
<TEI.2>  
<teiHeader> [informazioni dell'intestazione TEI] </teiHeader>  
<text> <front> [materiali del peritesto iniziale] </front>  
<body> [testo unitario] </body>  
<back> [materiali del peritesto finale] </back> </text>  
</TEI.2>
```

Elaborare testi: lo schema TEI *lite*

Struttura di un testo TEI – la codifica della trascrizione

Un **testo composito** sarà codificato usando una struttura globale come questa:

```
<TEI.2>
<teiHeader> [intestazione del testo composito] </teiHeader>
<text> <front> [peritesto iniziale del testo composito] </front>
<group>
  <text> <front>[peritesto iniziale del primo testo] </front>
  <body> [primo testo unitario] </body> <back> [peritesto finale del primo
    testo] </back> </text>
  <text> <front>[peritesto iniziale del secondo testo] </front>
  <body> [secondo testo unitario] </body> <back> [peritesto finale del
    secondo testo] </back> </text>
  [altri testi o gruppi di testi]
</group>
  <back> [peritesto finale del testo composito] </back> </text>
</TEI.2>
```

Elaborare testi: lo schema TEI *lite*

un semplice documento TEI è costituito, al livello testuale, dai seguenti elementi:

- <front>** Contiene qualsiasi materiale peritestuale (intestazioni, frontespizio, prefazioni, dediche, etc.) che si trovi prima dell'inizio del testo vero e proprio.
- <group>** Raggruppa un insieme di testi unitari o di gruppi di testi.
- <body>** Contiene l'intero corpo di un singolo testo unitario, con l'esclusione di ogni elemento peritestuale.
- <back>** Contiene qualsiasi appendice che segua la parte principale di un testo.

Elaborare testi: lo schema TEI *lite*

Gli elementi per le partizioni testuali

Il corpo di un testo in prosa può essere costituito solamente da *una serie di paragrafi*, oppure tali paragrafi possono essere raggruppati insieme in *capitoli*, *sezioni*, *sottosezioni*, etc.

Nel primo caso, ogni paragrafo viene codificato usando il marcatore `<p>`. Nel secondo caso, l'elemento `<body>` può essere diviso sia in una serie di elementi `<div1>`, sia in una serie di elementi `<div>`, i quali possono essere ulteriormente suddivisi:

`<p>` Codifica i paragrafi in prosa.

`<div>` Contiene una sezione del peritesto o del corpo di un testo.

`<div1>` Contiene una sezione di primo livello del peritesto o del corpo di un testo.

Elaborare testi: lo schema TEI *lite*

Qualora siano necessarie partizioni strutturali inferiori a una `<div1>`, quest'ultima può essere divisa in elementi `<div2>`, una `<div2>` in elementi inferiori `<div3>`, etc., fino al livello `<div7>`.

Tutti questi elementi di segmentazione accettano 3 attributi:

- type** Indica il nome convenzionale per questa categoria di partizione del testo (Libro, Capitolo, Poesia, etc.) Altri possibili valori includono Gruppo per gruppi di poesie, etc., trattati come un'unità singola, Sonetto, Discorso o Canzone.
- id** Specifica un identificatore unico per la partizione, che può essere usato per creare riferimenti incrociati o altri collegamenti (ad esempio a un commento).
- n** Specifica un breve nome mnemonico o un numero per la divisione, che può essere usato per identificarla, invece di **id**. Se già esiste una forma convenzionale di riferimento o un'abbreviazione per le parti di un lavoro (come il sistema libro/capitolo/verso delle citazioni bibliche), l'attributo **n** è la scelta da fare per registrarlo.

Elaborare testi per la ricerca storica

Da quanto accennato qui, lo schema TEI è stato elaborato per le opere letterarie, in prosa o in versi, ma può essere usato anche per codificare saggi e/o fonti storiche. In quest'ultimo caso, però, raramente le fonti si presentano (e si utilizzano) solo in forma di contenuti testuali unitari.

Va distinto a questo punto il **trattamento quantitativo delle fonti storiche seriali** (catasti, estimi, registri di dare/avere, registri di stato civile, elenchi, etc.), per le quali si adottano di norma database con schemi elaborati ad-hoc e si eseguono poi delle query in base alla propria ipotesi analitica.

Per il trattamento di **documenti differenziati o di serie difficili da modulare**, è sempre opportuno evidenziare, al fine di trattarli semparatamente, alcuni elementi/parti del documento (si è detto: **una codifica rende esplicite solo quelle caratteristiche importanti per il codificatore**).

digitalizzare

l'attività di **digitalizzazione** (ovvero di **riproduzione digitale di risorse analogiche**) è spesso centrale nel processo di **trattamento delle risorse** digitali, condividendo quasi tutte le problematiche del trattamento delle risorse digital-born e garantendo di norma un buon risultato rispetto agli utenti finali.

digitalizzare

la riproduzione digitale deve garantire quei **parametri di qualità delle risorse** già trattati parlando di analisi qualità tecnica e qualità dei contenuti).

in particolare, si devono rispettare standard tecnici e “pre-catalogare” meglio possibile le risorse così che possano agevolmente essere aggregate/associate e gestite in sistemi.

digitalizzare

la riproduzione digitale, semplificando, può essere schematizzata in quattro distinte attività:

- la “lettura” dell’originale analogico
- il suo trattamento tecnico per garantire la qualità del risultato
- il suo pre-trattamento informativo per garantirne la facile gestione
- il suo “collegamento” con le informazioni necessarie per l'accesso (sistema informativo)

digitalizzare

1. la “lettura” dell’originale analogico

questo, a seconda delle caratteristiche dell’originale analogico da trattare, può avvenire tramite *scanner* piani, basculanti, a rullo, dorsi fotografici digitali, etc.

La tecnologia ha elaborato soluzioni molto avanzate per il trattamento di originali di formato o supporto particolare (grandi mappe, volumi antichi con rilegature originali, pergamene arrotolate, etc.)

digitalizzare

1. la “lettura” dell’originale analogico

in questa fase si impongono alcune scelte tecniche generali per il progetto di digitalizzazione:

- qualità tecnica dell’immagine
- procedimenti speciali (frequenze di luce particolari per recuperare palinsesti, rasature ...)
- formato e denominazione automatica dei file prodotti
- Scelte per la conservazione dei *master*
- garanzia del rapporto tra originale e file

2. il trattamento tecnico dei files per garantire la qualità del risultato

in questa fase si possono/devono:

- trattare con OCR le immagini di testi
- ottimizzare alcuni elementi della qualità dell'immagine (risoluzione, saturazione, gamma, contrasto...)
- modificare formato e denominazione dei file
- creare copie dei file di formato/qualità diversa
- eseguire trattamenti speciali

3. il suo pre-trattamento informativo per garantirne la facile gestione

in questa fase si possono/devono:

- associare ai singoli files-immagine elementi informativi (metadati amministrativo-gestionali) che testimonino le scelte tecniche e amministrative compiute
- associare ai singoli files-immagine elementi informativi (metadati descrittivi e strutturali) che identifichino univocamente il rapporto tra risorse digitali e risorsa analogica

digitalizzare

i metadati amministrativi-gestionali (MAG)

la categoria funzionale dei metadati *gestionali* ha lo scopo di supportare la gestione della risorsa all'interno di una collezione/*repository* digitale.

L'importanza dei *metadati* gestionali-amministrativi e strutturali diventa sempre più netta via via che emergono le problematiche poste

sia dalla gestione “quotidiana” degli oggetti digitali (acquisire, gestire, archivi di file digitali),

sia dalla necessità di garantire la **conservazione a lungo termine** (cioè **l'accesso** a lungo termine agli oggetti digitali).

digitalizzare

i metadati amministrativi-gestionali (MAG)

può essere insito nei **metadati gestionali** anche un aspetto di **organizzazione degli oggetti digitali** parte di una collezione (si pensi ad esempio alla sequenza delle immagini delle pagine di un libro digitalizzato), che diventa indispensabile per una corretta fruizione degli oggetti digitali stessi.

In questo caso si parla di **metadati strutturali**: XML è funzionale all'organizzazione interna a un documento, per un sistema di dati è preferibile adottare uno schema apposito.

Lo schema MAG

In Italia lo standard più usato per gestire repository digitali risultato di digitalizzazione di documenti storici è MAG, prodotto nell'ambito del progetto Biblioteca Digitale Italiana e giunto alla versione 2.1. Comprende metadati sul progetto, descrittivi, strutturali e gestionali (per tutti i tipi di oggetti digitali) e qualcosa sui diritti.

Sul sito dell'ICCU si trova lo schema e un manuale in italiano con una premessa che spiega bene le regole di XML e degli XSD

<http://www.iccu.sbn.it/genera.jsp?id=267>

II MAG application profile

Lo schema MAG è un ***application profile***, vale a dire che integra elementi provenienti da **più namespace**. Nel dettaglio, lo schema è composto di quattro file (metadigit.xsd, metatype.xsd, audio.xsd. e video.xsd), collegati gli uni agli altri mediante un meccanismo di inclusione, e di quattro diversi namespace:

- **mag**: il target namespace
- **niso**: schema che traduce le linee guida del *Data Dictionary* NISO (LoC);
- **dc**: schema Dublin Core;
- **xlink**: schema W3C che consente di creare collegamenti fra diverse risorse

Il MAG application profile

L'elemento root di MAG (*metadigit*) può contenere i **seguenti elementi** corrispondenti ciascuno ad altrettante sezioni:

<gen>: contiene informazioni generali sul progetto e sul tipo di digitalizzazione; obbligatorio e non ripetibile

<bib>: contiene metadati descrittivi dell'oggetto analogico; obbligatorio e non ripetibile

<stru>: contiene metadati strutturali dell'oggetto digitale; opzionale e ripetibile

: contiene metadati specifici per le immagini fisse; opzionale e ripetibile

<audio>: contiene metadati specifici per file audio; opzionale e ripetibile

<video>: contiene metadati specifici per file video; opzionale e ripetibile

<ocr>: contiene metadati specifici relativi al riconoscimento ottico del testo; opzionale e ripetibile

<doc>: metadati specifici per oggetti digitali in formato testo che possono essere derivati o born digital; opzionale e ripetibile

<dis>: contiene metadati specifici per la distribuzione di oggetti digitali; opzionale e non ripetibile.