# Presentazione del Corso Codifica di Testi aa 2018-2019

Angelo Mario Del Grosso

angelo.delgrosso@ilc.cnr.it

*CNR-ILC-LicoLab*

Istituto di Linguistica Computazionale "A. Zampolli",
17th September 2018

# Piano della presentazione

# Progress status

# Di cosa mi occupo



## Filologia Computazionale

Analisi, progettazione e sviluppo di componenti software per sistemi Web di linguistica e filologia computazionale volti al trattamento di testi di tradizione medievale, a stampa e di autori moderni e contemporanei.

## Modelli Object Oriented per il Textual Scholarlship

Impiego delle nuove tecnologie nell'ambito delle Digital Humanities (DH) per la progettazione object–oriented di strumenti digitali Web–based rispondenti alle esigenze degli utenti accademici, studenti e sviluppatori.

# Presentazione del Corso



# Progress status

## Obiettivo

Illustrare i principi di modellazione e le prassi di codifica del testo per una adeguata rappresentazione ed elaborazione digitale di risorse testuali.

## Rationale

Fornire gli strumenti e le conoscenze necessarie per progettare e realizzare criticamente una codifica digitale di testi complessi, in particolare testi letterari e di interesse storico-culturale, usando le linee guida della Text Encoding Initiative (TEI).

## Competenze attese

Al termine del corso sarete in grado di valutare il metodo di codifica più appropriato allo scenario d'interesse, di creare uno schema di codifica TEI e di usare gli strumenti più idonei per la codifica e la (semplice) elaborazione e visualizzazione di un testo.

# Principali Argomenti
## Obiettivi, competenze e conoscenze



- Codifica dei caratteri e di testi
- I linguaggi di markup e introduzione a XML
- Creazione di schemi di codifica
- Le norme TEI (Text Encoding Initiative)
- Alcuni specifici Moduli TEI
- Definizione di schemi di codifica personalizzati
- introduzione ai fogli di stile XSLT
- elaborazione documenti XML-TEI (XSLT2.0, DOM)
- esempi, esercitazioni e seminari

# Bozza piano del Corso: Calendario 2018/2019

Lunedì dalle 14.00 alle 18.00



- Lezione 1 (24/9): Introduzione al corso e strumenti software utilizzati
- Lezione 2 (1/10): Elementi teorici relativi alla rappresentazione del testo
- Lezione 3 (8/10) Elementi di XML e introduzione alla definizione degli Schemi XML
- Lezione 4 (15/10): Elementi di Codifica TEI
- Lezione 5 (22/10): Modulo Specifico TEI

# Bozza piano del Corso: Calendario 2018/2019

Lunedì dalle 14.00 alle 18.00



- Lezione 6 (29/10): Moduli Editoriali TEI
- Lezione 7 (5/11): Codifica dei caratteri e Unicode
- Lezione 8 (12/11): Elaborazione documenti XML-TEI (XSLT e DOM)
- Lezione 9 (19/11): Esercitazione e discussione sui progetti
- Lezione 10 (26/11): Esercitazione e Recupero Argomenti
- (03/12-17/12): Eventuali esercitazioni, attività laboratoriali, recupero argomenti

## Dove troviamo i testi

Nella nostra cultura tradizionale la quasi totalità dei testi è "registrata", "trasmessa" e quindi "conservata" attraverso supporti e materiali fisici di varia natura e forma (iscrizioni su pietra, manoscritti di pergamena, papiri, carta, libri a stampa, incunabula, cinquecentine, etc).

## Perché codificare i testi

Per rendere disponibile questo patrimonio attraverso i sistemi per la gestione dell'informazione digitali e computazionali è necessario effettuare una trasposizione/transcodifica* dei testi dal loro supporto originario verso il nuovo supporto elettronico.

* *procedimento di conversione dei dati codificati secondo un sistema verso un sistema diverso*

## Scienze umane vs Scienze informatiche

Il rapporto tra sapere umanistico e informatica non è solo una questione meramente strumentale:
*l'informatica non è solo un utensile dalle notevoli capacità.*

Salto di paradigma sia da un punto di vista teorico sia metodologico.

## La codifica come metodologia

L'attività di codifica diviene funzione metodologica nell'ambito delle discipline che si occupano del testo.

## La codifica come analisi teorica

Il linguaggio di codifica adottato può essere considerato come un linguaggio teorico:

*Esplicitare e formalizzare le ipotesi interpretative su un certo oggetto di studio*

## Rappresentare il testo

Il focus del corso sarà incentrato sulla rappresentazione digitale del testo.

## Esistono dibattiti e controversie

Per ottenere una rappresentazione digitale del testo ci sono diversi formati e formalismi:
la nostra scelta ricade sulle norme suggerite dal consorzio TEI.

Molte questioni ancora non sono state risolte altre sono molto controverse, sia teorico-metodologico, sia pratico-tecnologico.

## Perché codificare

*Le differenze di formato sono più che altro estetiche e non sostanziali*

## Perché codificare

**Ma anche l'occhio umano vuole la sua parte**

# Progress status

## Rappresentare il testo in formato digitale

L'adozione di metodologie informatiche per il trattamento dei testi richiede in primo luogo la disponibilità di un'adeguata rappresentazione dei dati testuali in formato digitale.

## Perché è importante la codifica dei caratteri

La codifica dei caratteri costituisce il grado zero (basso livello) della rappresentazione di testi su supporto digitale.

*Le codifiche dei caratteri sono la base di qualsiasi schema di codifica testuale.*

## Rappresentazione digitale dei caratteri

I caratteri vengono rappresentati all'interno di un elaboratore mediante una sequenza di codici binari formati da opportune disposizioni di cifre composte da 0 e 1: 01100001 *lettera a*

**Character set** Per le discipline che studiano i sistemi di scrittura e l'analisi del linguaggio naturale, un insieme di caratteri astratti è detto Character set (unità alfabetiche). Astratto perché non riguarda la rappresentazione materiale della forma sul supporto, ma è relativo alla forma mentale, fatta di simboli di codifica (referenti).

**Coded Char Set** Per poter trattare un insieme di unità alfabetiche in formato digitale bisogna assegnare a ciascun carattere un numero intero non negativo detto code point.

**Character encoding** Il fine ultimo della codifica è quello di rappresentare una sequenza di caratteri in una sequenza di byte. La codifica di un carattere utilizza uno "encoding schema" che a sua volta mappa o trasforma ciascun code point in una sequenza di byte e quindi in ultima istanza in una sequanza di bit.

**Tabella del code page** Generalmente i code points sono espressi attraverso un sistema numerico esadecimale e disposti in una tabella di associazione.

## Codifica dei caratteri

Quindi trasformare una sequanza di caratteri appartenenti ad un char set in una sequenza di byte (bit) significa prima di tutto trasformare/mappare ciascun carattere nel proprio corrispettivo code point e successivamente codificare/serializzare questo code point nella relativa sequenza di byte (bit).

## Tabella Code Page ASCII 7 bit

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS | HT | LF | VT | FF | CR | SO | SI |
| 10 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US |
| 20 | SP | ! | " | # | $ | % | & | ' | ( | ) | * | + | , | − | . | / |
| 30 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 40 | @ | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| 50 | P | Q | R | S | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 60 | ` | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o |
| 70 | p | q | r | s | t | u | v | w | x | y | z | { | \| | } | ~ | DEL |

7 bit = 128 possibili caratteri; 32 caratteri di controllo; 96 caratteri effettivi

## codifica *ciao mondo!* 7 bit ASCII

6369 616F 206D 6F6E 646F 210A

## codifica *ciao è mondo!* 8 bit ASCII

6369 616f 20**e8** 206d 6f6e 646f 210a

## codifica **ciao è mondo!** UNICODE UTF-8

6369 616f 20**c3** **a8**20 6d6f 6e64 6f21 0a

## Complessità di rappresentazione universale dei caratteri

Se si considerano tutti i possibili alfabeti del mondo e le molteplici esigenze poste dalla scrittura delle fonti manoscritte antiche e medievali, ci si accorge che la realizzazione di un sistema universale per la codifica dei caratteri è un progetto molto complesso con svariate sfide da affrontare.

## Complessità di rappresentazione universale

Ad oggi, lo standard de facto per la codifica dei caratteri è lo UNICODE. Esso è in grado di codificare più di un milione di differenti unità alfabetiche, segni di interpunzione e diacritici, appartenenti a centinaia di diverse lingue.

## Complessità di rappresentazione universale

Unicode assegna i propri code point in un range che va da $0x0$ a $0x10FFFF$. In Unicode il code point viene indicato con una "U" seguita da un segno "+" seguito a sua volta dall'esadecimale con padding del codice (es: U+0041 lettera a).

## Unicode Transformation Format

Lo Unicode è un Coded Char Set e per essere concretamente serializzato su un supporto elettronico deve essere trasformato attraverso qualche tipo di schema di codifica. L'UTF (Unicode Transformation Format) mappa i code point Unicode in sequenze di byte (bit).

## UTF standards

Esistono tre tipi di schemi di codifica che vanno sotto il nome di UTF, ciascuno è identificato dal minimo numero di bit necessario a codificare ciascun code point: UTF-8; UTF-16; UTF-32.

# Progress status

## Codificare un testo

La codifica dei caratteri evidentemente non esaurisce i problemi per una opportuna rappresentazione delle caratteristiche interne ed esterne di un testo.

## Codificare un testo

Di fatti la codifica del testo è una questione molto più complessa di una semplice riproduzione meccanica di un dato.

## Rappresentare un testo

La rappresentazione digitale di un testo è una operazione intrinsecamente assai difficile perché coinvolge una pletora di aspetti a vari livelli di astrazione, a varie dimensioni e a varie granularità sia teorici, sia metodologici, sia tecnologici e sia pratici.

## Rappresentare un testo

**Prima di poter fare qualsiasi ipotesi su come compiere una codifica di un testo e su come rappresentarlo digitalmente, bisogna stabilire cosa si intende per testo.**

## Un testo non ha una struttura rigida, predefinita:

- Non è rappresentabile solo come un insieme di record di un archivio elettronico.
- Non è rappresentabile solo come un insieme di tabelle di una banca dati.
- Non è rappresentabile solo come un albero o un insieme di sotto-alberi
- Non è rappresentabile solo come un grafo o come un insieme di sotto grafi

## La rappresentazione di un testo

- modello lineare: sequenza di dati non strutturati
- modello a record: enumerazione delle proprietà
- modello tabulare: insieme di dati omogenei
- modello ad albero: gerarchie di dati e insiemi di dati
- modello grafo: rete di strutture informative interconnesse tra loro

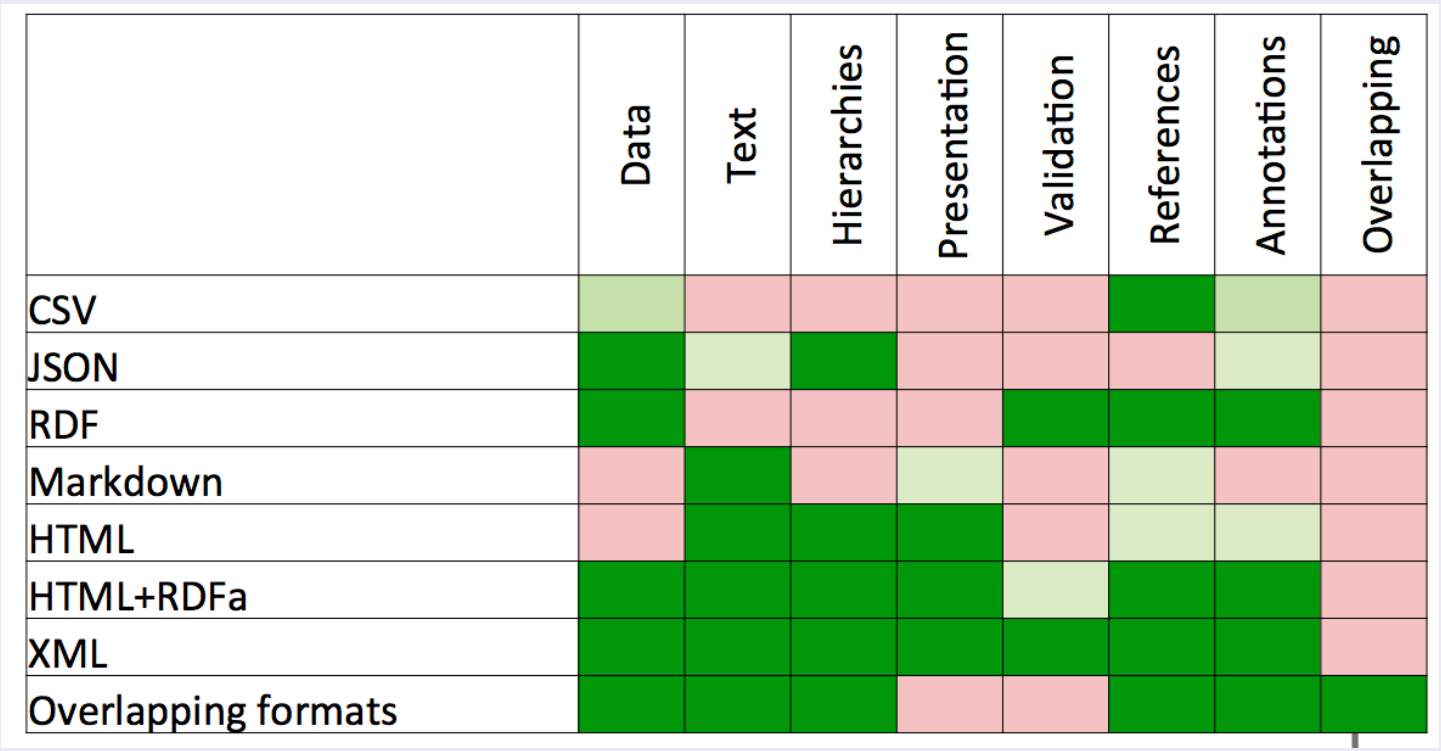
## Formati di rappresentazione

Un formato è un insieme di regole e convenzioni formali per rappresentare un insieme di dati, nel nostro caso un testo.

## Importanza dei formati

Seppur isomorfi la scelta di un formato condiziona molto l'efficienza delle operazioni e l'efficacia delle dichiarazioni.

## Formalismi



| | Data | Text | Hierarchies | Presentation | Validation | References | Annotations | Overlapping |
|---|---|---|---|---|---|---|---|---|
| CSV | | | | | | | | |
| JSON | | | | | | | | |
| RDF | | | | | | | | |
| Markdown | | | | | | | | |
| HTML | | | | | | | | |
| HTML+RDFa | | | | | | | | |
| XML | | | | | | | | |
| Overlapping formats | | | | | | | | |

courtesy of *Fabio Vitali*

## In literature, there are **four main models** to represent a **text**:

1. Text as **linear data** model (typical in Natural Language Processing)
   - ✓ Plain Text and Regular Expression
     - ➢ *String Manipulation*
   - ✗ Limit: nor physical nor logical structure among textual elements
2. Text as **tabular data** model (typical in document processing)
   - ✓ Comma separated values (CSV)
     - ➢ *Indexing, retrieval*
   - ✗ Limit: no hierarchies inherently described
3. Text as **hierarchical data** model (typical in scholarly editing - OHCO)
   - ✓ XML data model
     - ➢ *TEI/XML Encoding and Serialization*
   - ✗ Limit: no overlapping hierarchies allowed
4. Text as **graph data** model (typical in knowledge representation)
   - ✓ Multi-dimensional Texts
   - ✓ Multi-versioned Texts
     - ➢ *RDF as Data Model*
     - ➢ *OWL as Language for Knowledge Description*
   - ✗ Limit: No type systems for textual scholarship

## Formati come formalismi

Data l'importanza metodologica il formato del dato diviene un vero e proprio formalismo, si parla cioè di linguaggi di codifica in quanto questi sistemi si basano su un insieme di istruzioni rigorose di codifica.

## Formati e formalismi di codifica

Quindi ogni pezzo di informazione aggiunta ad un testo grezzo attraverso l'inserimento di dati metatestuali (markup, annotazione, codifica), constituisce il risultato di una analisi e di una interpretazione che è stata condotta (da un umano o da una macchina) al fine di esplicitare e rappresentare nel modo più accurato e completo possibile le informazioni da veicolare attraverso il formato digitale prescelto (anche in modo incrementale).

# Progress status

## Metodi e tecniche per la codifica di testi

La riflessione sui metodi e le pratiche migliori per la codifica elettronica dei testi è stato uno dei temi fondamentali della ricerca e della sperimentazione nel dominio dell'Informatica umanistica per molti anni.

## XML per la descrizione e la codifica

Ad oggi la soluzione considerata ottimale per una corretta rappresentazione del testo è l'adozione dei markup language descrittivi basati su XML.

## TEI-XML

Standard de facto per la codifica dei testi è considerato lo schema XML messo a punto dalla Text Encoding Initiative (TEI-XML).

## Perché XML

- separazione dei dati dall'applicativo di authoring/editing
- separazione della rappresentazione dei dati dalla presentazione dei dati
- possibilità di trasformare i dati in qualsiasi altro formato compatibile
- leggibilità dei documenti XML da parte di esseri umani

## Perché XML

- standard w3c testuale, aperto, personalizzabile e liberamente utilizzabile
- semplicità di condivisione e scambio dati (interoperabilità e portabilità)
- adatto per codificare dati semistrutturati oltre che a dati strutturati
- validazione del documento attraverso spcificazioni formali

# XML titolo
## XML sottotitolo



- XSD: XML Schema Definition Language
- XPath: XML Path Language
- XSL: eXtensible Stylesheet Language
- XSL-T: XSL – Transformations
- XSL-FO: XSL – Formatting Objects
- XQuery: XML Query Language for XML Databases
- XInclude: XML inclusion Language
- DTD: Document Type Definition Language
- RelaxNG: Regular Expression Language for XML (New Generation)

## Perché XML

Adottando la tecnologia e il linguaggio XML abbiamo la possibilità di creare linguaggi di marcatura personalizzati e specifici per ogni esigenza e dominio.

## Vantaggi

Attraverso XML è possibile strutturare i dati, gestire in modo nativo strutture gerarchiche, elaborare e presentare i dati con strumenti xml nativi, validare i tipi di strutture e i tipi di dati consentiti, gestire riferimenti incrociati tramite opportuni meccanismi di dereferenziazione, aggiungere e gestire annotazioni a vari livelli di granularità.

# Progress status

# Prerequisiti
## Cosa è consigliato sapere



- Buona conoscenza dell'inglese
- Nozioni di HTML / CSS / JS
- Computer personale o a disposizione per svolgere gli esercizi assegnati
- Uso di editor di testo
- Uso di editor per programmatori o editor XML
- Eseguire commandi da shell

- Editor di testo professionali (syntax highlighting per XML, workspace)
- Editor XML + processore XSLT (normalmente è integrato nell'editor XML)
- Navigatore web
- Manuali di codifica (Guidelines TEI P5)
- Materiale didattico (slide, esempi, esercizi)

## Visual Studio Code

## XMLlint

# Editor XML
## Cosa è consigliato usare



- un buon editor open source: XML Copy Editor (http://xmlcopy-editor.sourceforge.net/)
- per Mac: XMLSpear, Textmate, Eclipse, IntelliJIDEA
- un ottimo editor: Oxygen (http://www.oxygenxml.com/) - multi piattaforma, ma non gratuito (in prova gratuita per un mese)
- altri editor: funzioni fondamentali sono la validazione, l'autocompletamento, l'esecuzione di fogli di stile

## EditiX

# Programma d'esame
## Cosa bisogna fare per superare l'esame



- Studiare i lucidi delle lezioni
- Padronanza degli esercizi svolti durante il corso
- Studiare i testi indicati dal docente
- Studiare i capitoli delle Guidelines TEI che riguardano il corso e il progetto
- Realizzare il progetto di codifica concordato con il docente

# Modalità d'esame
## In cosa consiste l'esame



- Invio del progetto (consigliato tramite github)
- Colloquio
  - Discussione del progetto
  - Verifica delle conoscenze di base XML, XSD, XSL
  - Verifica delle basi teoriche
  - Conoscenza di TEI P5 (moduli principali, parti spiegate a lezione, moduli particolari utilizzati nel progetto)

## Slide

- Slide delle lezioni corso aa 2018-2019
- Materiale integrativo fornito dal docente
- Repo github del corso
  `https://github.com/angelodel80/corsoCodifica`

## Libri


- Burnard, L. (2014). What is the Text Encoding Initiative? How to add intelligent markup to digital resources. Marseille: OpenEdition Press.
- Ciotti, F. (2007). Il testo e l'automa: saggi di teoria e critica computazionale dei testi letterari. Aracne.
- Carey, P., and Vodnik, S. (2014). New Perspectives on XML, Comprehensive. Cengage Learning.
- Goldberg, K. H. (2010). XML: Visual QuickStart Guide. Pearson Education.

## Siti Web

- http://www.tei-c.org/
- http://teibyexample.org/
- https://www.w3.org/standards/xml/

# Progress status

# References




Ciotti 2011


Orlandi 2010


Pierazzo 2015


Pierazzo 2016


Williams, I. (2009). Beginning XSLT and XPath: Transforming XML Documents and Data. Wiley.


Kay, M. (2011). XSLT 2.0 and XPath 2.0 Programmer's Reference. Wiley.


CC Wiki *Best practices for attribution*, CC Wiki 2014, `https://wiki.creativecommons.org/wiki/Best_practices_for_attribution`


IBM XML *XML*, `http://ibm.com`