UNIVERSITÀ DEGLI STUDI DI MESSINA

FACOLTÀ DI LETTERE E FILOSOFIA

CORSO DI LAUREA IN LETTERE MODERNE

CATTEDRA DI LETTERATURA E FILOLOGIA SICILIANA

CODIFICA ELETTRONICA DEI TESTI LETTERARI ED E-BOOK: LA MARCATURA XML – TEI ED IL TRATTAMENTO INFORMATICO DEL ROMANZO *BALTICO* DI MATTEO COLLURA

DigiSic Archivio Digitale della Letteratura Siciliana

2004

Premessa

Oggetto di questa relazione è la codifica elettronica del testo letterario ed i suoi rapporti con il cosiddetto libro elettronico o e-book e con la nuova realtà delle biblioteche digitali.

Il lavoro si sviluppa in tre parti strettamente correlate fra di loro: la prima affronta in modo generale le tematiche della codifica elettronica del testo, la seconda focalizza l'indagine sul tema dell'e-book, nella terza ed ultima parte invece si espongono dettagliatamente le attività compiute ed i risultati ottenuti nella codifica del romanzo Baltico di Matteo Collura.

Prima di entrare nel vivo della trattazione è opportuna una precisazione relativamente al sistema adottato per le note. Vista la natura dell'argomento affrontato è ovvio che numerose delle fonti da noi consultate fossero costituite da pagine Web o da altri testi in formato digitale, si è presentato dunque il problema della citazione bibliografica non essendo ancora invalsa una pratica comune per la citazione di questo tipo di fonti. La metodologia da noi adottata in questo testo si rifà, con alcuni piccoli aggiustamenti necessari per rimanere coerenti con il metodo usato nella citazione di fonti tradizionali, al sistema illustrato nella guida bibliografica della Columbia University, che al momento gode di una buona diffusione. Nel dettaglio si è deciso di fare così: nome e cognome dell'autore in maiuscoletto; titolo dell'articolo in corsivo; titolo del sito o nome dell'organizzazione che lo gestisce in corsivo; URL completo di http tra < > se citato in una pagina web; data di pubblicazione; data dell'ultimo accesso da parte del compilatore della citazione, tra parentesi.[1]

[1] La guida bibliografica della Columbia University suggerisce di inserire cognome e nome dell'autore, separati da una virgola, noi abbiamo invece preferito adottare lo stesso sistema adoperato nella citazione di fonti tradizionali con nome e cognome dell'autore in maiuscoletto e titolo della fonte in corsivo.

PARTE PRIMA

I.1. BIT E ATOMI

Oggetto del nostro studio è il libro elettronico, ossia un oggetto digitale; appare, dunque, opportuno soffermarsi a riflettere sulla natura del "mondo digitale" che, analogamente al mondo materiale, ha proprie regole e caratteristiche che sovrintendono e governano l'esistenza digitale.

Quella dei computer e dei dispositivi digitali in genere è una realtà fatta di 0 e di 1, queste due cifre non sono altro che bit, così come si legge nel dizionario De Mauro: "l'unità minima di informazione che il calcolatore può riconoscere, rappresentata dalla presenza o assenza di un impulso elettrico", nonché "ciascuna delle due cifre del sistema di numerazione binario".

La parola bit deriva dall'inglese ed è contrazione di *binary digit* "numero binario", in sostanza il mondo dei computers è fatto di bit.

"Un bit non ha colore, dimensioni o peso, e può viaggiare alla velocità della luce. È il più piccolo elemento atomico del DNA dell'informazione"[2] questa è la celebre e provocatoria definizione di bit data da Nicolas Negroponte nel suo saggio Being Digital del 1995 , che bene ci introduce nella riflessione sulle caratteristiche dell'oggetto digitale.

Come il mondo reale al suo livello più basso è un mondo fatto di atomi, così quello digitale è fatto di bit.

Gli oggetti digitali hanno peculiarità e caratteristiche uniche, limiti e vantaggi:

I.1.1. Immaterialità

Una delle caratteristiche più importanti degli oggetti digitali è rappresentata dalla loro immaterialità. Un oggetto digitale, al suo livello più basso, è sempre

[2] Cfr. NICHOLAS NEGROPONTE, *Essere Digitali*, trad. it. di Franco e Giuliana Filippazzi, Milano, Sperling & Kupfer Editori S.p.A. 1999[2], p. 3 (ed. orig. *Being Digital*, Alfred A. Knopf, Inc., 1995).

rappresentato da una sequenza di zero ed uno, ciò comporta una grande flessibilità sia per quanto riguarda la trasmissibilità , sia per quanto riguarda la conservazione di materiali digitali.

I.1.2. Trasmissibilità

A differenza degli atomi, è estremamente semplice spostare bit da un capo all'altro del globo.

"Un bit non ha colore, dimensioni o peso, e può viaggiare alla velocità della luce", qualsiasi canale di comunicazione capace di esprimere almeno due valori,cioè gli zero ed uno del sistema binario, è un canale idoneo alla trasmissione dell'informazione digitale.

I bit possono viaggiare sotto forma di impulsi elettrici (ed è cosi che viaggiano all'interno dei dispositivi elettronici), onde radio (il linguaggio Morse, ad esempio, è sostanzialmente un codice binario), impulsi luminosi, suoni (i modem analogici dei nostri computers non fanno altro che trasformare i bit provenienti dal computer in dei suoni che vengono inoltrati nella linea telefonica ed una volta recepiti da un altro modem trasformati nuovamente in bit; queste operazioni hanno il nome di modulazione e demodulazione, da ciò deriva il termine modem).

I.1.3. Conservazione

La grande astrattezza dei bit permette loro di essere archiviati sui dispositivi più diversi. Come il lettore avrà capito, unica indispensabile esigenza del supporto è quella di poter presentare almeno due stadi, stadi che possono essere rappresentati ad esempio dalla carica positiva o negativa delle memorie magnetiche come nei floppy, negli Hard Disk etc. oppure dai microsolchi incisi sulle superfici dei cd, dvd e poi letti da un raggio laser.

I.1.4. Riproducibilità

Uno dei vantaggi, e per certi versi anche un limite, degli oggetti digitali è la perfetta riproducibilità.

Qualsiasi copia di un oggetto digitale sarà perfettamente identica all'oggetto di cui è copia, in sostanza un clone perfetto. La copia è indistinguibile dalla matrice e non vi è alcun decadimento qualitativo nel passaggio dall'originale alla copia, cosa che invece avveniva con i sistemi di copia analogici; in realtà non avremo altro che la medesima sequenza di zero e di uno.

I.1.5. Duttilità

I bit sono duttili. È estremamente facile modificarli e rielaborarli, inoltre i bit sono il pane quotidiano dei computers, è naturale quindi eseguire su di loro tutte quelle operazioni di calcolo, elaborazione, valutazione che questi strumenti ci consentono.

A conclusione di questa breve, e necessariamente lacunosa, dissertazione su bit ed atomi desidero citare nuovamente Nicolas Negroponte al quale sono profondamente debitore per le idee di fondo di questo paragrafo, che ritengo ben si presti a concludere queste pagine.

"Tutti concordano sul fatto che una biblioteca pubblica sia una buona cosa. È una cosa buona per la cultura, per la società. Una biblioteca pubblica funziona perché essa si basa su atomi: dovete portare i vostri atomi alla biblioteca. Alcuni di noi hanno un po' troppi atomi. Allora prendete il libro in prestito. Non è solo un altro atomo, ma - e questo è così ovvio che non ci pensiamo mai - il guaio è che quando prendete in prestito un atomo non ci sono atomi rimanenti. Resta uno spazio vuoto. Voi portate il libro a casa, lo leggete, diciamo in una settimana, lo riportate alla biblioteca. Magicamente qualcuno lo prende in prestito di nuovo, e lo riporta indietro dopo una settimana. Così 52 persone avranno letto il libro in un anno. Ora invece renderò la biblioteca pubblica digitale. Cambierò solo questo: muterò gli atomi in bit. Non dovrò trasportare i miei atomi alla biblioteca. È una cosa così ovvia, ma non viene mai detta a scuola: è che quando prendete in prestito un bit, c'è sempre un altro bit che rimane. Così ora 20 milioni di persone possono

prendere in prestito questo libro simultaneamente, senza muoversi di casa, giusto battendo alcuni tasti….

È davvero molto, ma molto interessante considerare alcuni eventi in termini di bit e di atomi: questo cambierà il vostro modo di vedere quel tipo di mondo che è il mondo digitale."[3]

[3] NICHOLAS NEGROPONTE, durante un'intervista realizzata dallo staff di Mediamente *La rivoluzione digitale*, in *Mediamente Biblioteca digitale*
<http://www.mediamente.rai.it/home/bibliote/intervis/n/negrop02.htm>
Venezia - Ca' Foscari, 3 giugno 1995, (23 settembre 2003).

I.2. CODIFICA INFORMATICA DEI TESTI

Dobbiamo a Fabio Ciotti una delle più complete ed esaustive definizioni di codifica informatica dei testi. Secondo lo studioso con codifica informatica dei testi *intendiamo la rappresentazione formale di un testo ad un qualche livello descrittivo, su di un supporto digitale, in un formato utilizzabile da un elaboratore (Machine Readable Form) mediante un opportuno linguaggio informatico*[4].

Come giustamente nota Ciotti ogni testo può esser considerato da vari punti di vista, da una parte vi è la componente materiale: il supporto, le tracce d'inchiostro; dall'altra vi è una componente astratta, la sequenza verbale, la quale a sua volta genera una serie di livelli semantici e strutturali.

Con la codifica elettronica si fanno passare i testi dal mondo degli atomi a quello dei bit, con tutto ciò che questo comporta , soprattutto in termini di fruizione ed elaborazione; tuttavia si ha anche il passaggio da un codice ad un altro: "Si tratta dunque di un processo rappresentazionale che implica una serie di operazioni di selezione e classificazione degli elementi rilevanti in funzione di un determinato punto di vista"[5] . A proposito di questo, ben si è espresso uno dei pionieri dell'informatica umanistica in Italia, Giuseppe Gigliozzi: "Codificare un testo significa… esplorare il codice di partenza. Studiarne regolarità e funzioni. Confrontarle con le costrizioni proprie del codice prescelto e riproporre le relazioni (che sono già in un codice) del testo originale nel nuovo codice che, nel nostro caso, dovrà essere utilizzabile per un'elaborazione elettronica,… Lavoreremo, quindi con due codici: il codice del documento da cui muoviamo e il codice che renderà questo documento disponibile per un'elaborazione elettronica"[6].

---

[4] Cfr. FABIO CIOTTI, *Cosa è la codifica informatica dei testi*, in AA.VV., *Atti del Convegno Umanesimo & Informatica* (Trento 24-25 maggio 1996), a cura di Daniela Gruber e Patrick Pauletto. <http://circe.lett.unitn.it/circe/html/attivita/uman/ciotti.pdf>
(24 settembre 2003)

[5] Vedi nota precedente.

[6] Cfr. GIUSEPPE GIGLIOZZI , *Il testo e il computer. Manuale di informatica per studi letterari*, Milano, Bruno Mondadori, 1997, p. 56.

Sempre secondo Gigliozzi col processo di codifica si realizza anche un modello dell'oggetto testo, modello che può essere qualcosa di "più piccolo del testo" e, vorrei aggiunger io, qualcosa di "più grande", ossia contenere, o meglio esplicitare, informazioni in più rispetto al testo originario; in ogni caso, come afferma Gigliozzi, è bene tener presente la necessità che il modello rispetti "le leggi dell'*isomorfismo*. Dove per *isomorfismo* potremmo intendere una trasformazione che conservi l'informazione"[7].

Alla luce di tutto ciò più chiara dovrebbe apparire la definizione di codifica come *rappresentazione formale di un testo ad un qualche livello descrittivo* ed in virtù di quanto detto si può ben comprendere come la codifica possa esser vista anche come un'attività interpretativa , che comporta scelte e valutazioni da parte del codificatore, che non è un mero trascrittore del testo, ma può, per certi versi, esser visto come un traduttore che traspone il testo in una forma che ne consenta l'analisi con mezzi informatici, e possa eventualmente permettere nuove forme di fruizione.

[7] Cfr. GIGLIOZZI *Il testo e il computer. Manuale di informatica per studi letterari*, (vedi n.3) p.59.

I.3. LA CODIFICA DEI CARATTERI

Se come detto *il più piccolo elemento atomico del DNA dell'informazione* è il bit, potremmo considerare il carattere l'unità atomica dell'informazione.

Il grado più basso della codifica informatica dei testi è quindi la codifica dei caratteri.

Come abbiamo visto i computers sono in grado soltanto di trattare numeri in formato binario, affinché l'elaboratore sia in grado di operare con lettere o altri segni è necessario creare delle tabelle per la conversione dei numeri (bit) in caratteri (lettere, numeri, etc.), semplificando al massimo, possiamo dire che con queste tavole si assegna ad ogni carattere un numero.

I.3.1. Le tavole dei caratteri

Si suole parlare di *character repertoire* per l'insieme dei simboli che si vuol rappresentare, di *code set* per i codici numerici che fanno riferimento ai simboli del *caracter repertoire*, l'insieme risultante di *character repertoire* e *code set* è detto *coded character set* o *charset;* abitualmente i *charset* vengono rappresentati sotto forma di tabelle contenenti i simboli ed i rispettivi codici numerici, per questo motivo si suole parlare di tavole dei caratteri o *code table*.

Un piccolo esempio potrà aiutare a comprendere meglio quanto appena detto.

Allorché digitiamo sulla tastiera un tasto, per esempio la A maiuscola, mandiamo al computer un impulso elettrico, dei bit rappresentanti un numero binario, per la precisione nel caso della A maiuscola 1000001, il computer va a verificare nella propria tavola caratteri che il numero binario ricevuto corrisponde alla posizione 65 che a sua volta corrisponde al simbolo A, allora l'elaboratore preleva dal proprio repertorio di simboli grafici, "font", il glifo, ossia l'immagine del simbolo scelto, e lo manda a video. Lo stesso procedimento di corrispondenze fra numeri e simboli avviene allorquando apriamo un file contenente testo, tutti i codici numerici dei

caratteri del file vengono tradotti nelle rispettive immagini grafiche e mandati in output.

È opportuna una precisazione, i caratteri contenuti nei charset non sono solo caratteri grafici, ma poiché un computer deve gestire non solo i cosiddetti caratteri "stampabili", ossia tutti quelli associati ad un simbolo grafico, ma anche istruzioni per la visualizzazione e le operazioni sui files, un carattere può anche, per esempio, rappresentare il comando di "a capo", il comando d'emissione di un segnale sonoro , o il segnale di inizio o di fine di un file; tali caratteri per lo più vengono usati nella trasmissione dati tra computers, o tra computer e stampante, e prendono il nome di caratteri di controllo; in genere ai caratteri di controllo vengono assegnate le prime 32 posizioni delle tavole dei caratteri.

I.3.2. Un po' di storia

Come detto, tutti i computer memorizzano il testo sotto forma di numeri. È tuttavia possibile che sistemi diversi memorizzino lo stesso testo utilizzando formati numerici differenti, e quindi possibile che il valore numerico che corrisponde ad un determinato carattere in un sistema , corrisponda a tutt'altro carattere in un altro; ad esempio in un computer che esegue Microsoft Windows utilizzando la tabella codici predefinita 1252 e in un computer Apple Macintosh che utilizza la tabella codici Macintosh Roman, la posizione 232 di queste tabelle dei caratteri corrisponde rispettivamente alla lettera **è** nel sistema Windows ed alla lettera **Ë** in quello Macintosh[8]. Quindi transitando da un sistema all'altro un documento contenente testo vedrebbe sostituite tutte le **è** con **Ë**.

Agli albori dell'era informatica il problema dell'incomunicabilità fra sistemi era assai diffuso. Ogni apparato informatico usava la propria tavola dei caratteri,

---

[8] CHRIS LOVETT, *Come codificare i dati XML*, in *MSDN Library* <http://www.microsoft.com/italy/msdn/library/default.asp?url=/italy/msdn/library/xmlsoap/xmlencodings.asp?frame=true> marzo 2000, (26 settembre 2003).

spesso totalmente incompatibile con quella degli altri sistemi; si è parlato perciò di "babele informatica". Per cercare di ovviare a tutto questo, a partire dal 1963, un gruppo di studiosi tra cui Robert W. Bemer (considerato il padre dell'ASCII), si impegnarono nello sviluppo di un sistema di codifica standard per l'interoperabilità e l'interscambio di informazioni tra computers. Il lavoro di sviluppo culminò nel 1968 con l'approvazione da parte dell'ente nazionale americano per la standardizzazione (**ANSI**), che registrò come standard nazionale la tavola che prese il nome di *American standard code for information interchange,* meglio conosciuta come ASCII. (vedi tavola 1)

La tavola ASCII fa uso di sette bit per determinare i caratteri, cosicché sono solo 128 le posizioni possibili nella stessa (trattandosi ovviamente di tutte le combinazioni possibili di 0 e 1: $2^7 = 128$) .

Nel set ASCII, i primi 32 codici sono assegnati a caratteri di controllo, i restanti 96 codici sono assegnati ai segni d'interpunzione ed altri simboli, alle cifre da 0 a 9 e alle lettere dell'alfabeto latino, maiuscole e minuscole; mancano totalmente le lettere accentate di cui la lingua inglese non fa uso.

Nonostante tutto ciò, fu grande il successo di questa tavola, tanto che tutt'oggi è usata soprattutto per quanto riguarda le comunicazioni telematiche e la posta elettronica, e resta tuttora, praticamente, la sola universalmente condivisa da tutte le macchine.

Nel 1991 l'ISO[9] ha registrato, rendendola standard internazionale, la tavola ASCII col nome di ISO 646-IRV (International Reference Version).

Ben presto ci si rese conto, soprattutto da parte dei paesi non anglofoni, dell'inadeguatezza del charset ASCII per i testi scritti nella propria lingua madre. Si usavano vari espedienti per rappresentare le lettere accentate ed i vari simboli diacritici delle scritture occidentali, finché non si pensò di portare da 7 a 8 i bit

[9] *L'International Organization for Standardization* (**ISO**) è un organizzazione non governativa internazionale, fondata nel 1947 con sede a Ginevra, che riunisce le organizzazioni nazionali per la standardizzazione di 147 diversi paesi, con lo scopo di promuovere lo sviluppo di standard internazionali, al fine di facilitare lo scambio di beni e servizi e di sviluppare la cooperazione nelle sfere dell'attività intellettuale, scientifica, economica e tecnologica.

della tavola ASCII, così il numero di caratteri disponibile divenne 256 (2^8 = 256), si ebbe in tal modo il cosiddetto ASCII esteso, che nelle varie versioni internazionali ospitava tutti quei caratteri non presenti nell'ASCII a 7 bit.

A partire dagli anni ottanta, soprattutto per opera della *European Computer Manufacturer's Association* (ECMA), sono stati rilasciati numerosi charset a 8 bit affinché fossero usati come standard per determinati gruppi di lingue, i quali furono successivamente recepiti dalla ISO e rilasciati come standard internazionali col nome di ISO 8859.

Particolarmente diffuso è il code set ISO 8859-1 , meglio conosciuto come ISO Latin 1, dal nome della precedente tavola ECMA . Esso contiene i caratteri principali delle lingue occidentali e anglosassoni con alfabeti latini , ed è usato da molte applicazioni Internet, e da molti sistemi operativi. La dicitura "-1" sta ad indicare che si tratta della prima di una serie di tavole , ad esempio la ISO 8859-2 è invece usata per le scritture latine del centro e dell'est europeo; la più recente è la ISO 8859-15, la quale è uguale alla ISO 8859-1 con in più l'aggiunta del simbolo dell'euro .

Queste tavole sono costruite tutte in modo che per la prima metà garantiscono l'interoperabilità fra computers, così i primi 128 caratteri sono esattamente quelli di Iso 646 Irv; la seconda metà cerca di soddisfare, con altri 96 caratteri stampabili, le necessità di specifiche lingue o gruppi di lingue.

I.3.3. Prospettive future

Se le tavole ISO 8859 hanno rappresentato un importante passo avanti per quanto riguarda l'interscambio di comunicazioni soprattutto all'interno di uno stesso gruppo linguistico, tuttavia continuano a persistere problemi nella visualizzazione di testi rappresentati con differenti tavole dei caratteri. Per cercare di risolvere il problema della "Babele informatica", a partire dagli inizi degli anni ottanta, sono sorti due progetti con l'obiettivo di creare una tavola di codifica dei caratteri universale.

Uno è stato il progetto ISO 10646, l'altro il progetto Unicode[10], portato avanti da un consorzio di aziende produttrici di software. A partire dal 1991 le due organizzazioni, comprendendo l'inopportunità dell'esistenza di due progetti distinti per una tavola universale dei caratteri, hanno unito i loro sforzi ed hanno lavorato insieme per creare un'unica tavola dei caratteri. Entrambi i progetti esistono ancora e pubblicano i loro rispettivi Standard indipendentemente, comunque il consorzio Unicode e la ISO si sono accordati per tenere le tavole dei codici UCS[11] ed ISO 10646 standard e compatibili, inoltre si sono impegnati per coordinare ogni futura estensione.

Unicode 1.1 corrisponde ad ISO 10646-1:1993, Unicode 3.0 corrisponde ad ISO 10646-1:2000, e Unicode 3.2 corrisponde ad ISO 10646-2:2001.

Tutte le versioni di Unicode a partire dalla due , e quindi anche le rispettive tavole ISO 10646, sono compatibili, solo i nuovi caratteri vengono aggiunti, i caratteri esistenti non vengono rimossi o rinominati.

È da notare che le prime 256 posizioni dell'Unicode sono uguali a quelle della tavola Iso 8859-1 questo agevola la migrazione verso il nuovo sistema.

In origine nel progetto Unicode per aggirare i limiti delle tavole ad 8 bit si era pensato di creare una nuova tavola da 16 bit che avrebbe permesso ben 65.536 combinazioni, numero più che sufficiente per rappresentare tutti i caratteri delle lingue occidentali antiche e moderne, tuttavia non abbastanza capiente per accogliere, ad esempio, gli 80.000 simboli ideografici Han oppure le 11.000 sillabe coreane Hangul.

Oggi lo standard ISO10646/Unicode si basa su una codifica a 32 bit (in realtà per motivi tecnici sono solo 31 i bit utilizzati per definire le combinazioni) che consente oltre due miliardi di possibili caratteri, numero che dovrebbe essere decisamente sufficiente per realizzare una tavola di codifica dei caratteri veramente universale.

[10] L'Unicode Consortium è un consorzio senza fini di lucro, istituito per sviluppare, estendere e promuovere l'uso dello standard Unicode. I membri del consorzio rappresentano un'ampia gamma di aziende e di organizzazioni attive nell'information technology, <www.unicode.org>.

[11] UCS ossia *Universal Character Set*, questo è il nome delle tavole di codifica dei caratteri rilasciate dal consorzio Unicode.

La versione 3.2 di Unicode definisce ben 95.221 caratteri comprendenti tutte le principali lingue occidentali con alfabeto latino, quelle slave ad alfabeto cirillico, l'ebraico, l'arabo nonché buona parte delle scritture africane, asiatiche ed indiane, tra le scritture comprese in Unicode vi sono anche il greco classico, oltre ovviamente quello moderno, ed il copto.

L'implementazione informatica di Unicode tuttavia comporta dei problemi, in particolar modo per quanto riguarda la crescita della dimensione dei files, per tal motivo sono stati creati dei charset detti UTF, *Universal Character Set Transformation Format*.

UTF usa una tecnica di bit-shifting (spostamento dei bit) per codificare i caratteri Unicode. In sostanza l'UTF usa 7 bit per carattere per codificare i primi 127 caratteri corrispondenti all'ASCII standard, e attiva l'ottavo bit solo quando serve la codifica Unicode. Ogni carattere Unicode viene quindi codificato con un numero variabile di byte[12] che va da 1 a 3. Il vantaggio è che il testo è visualizzabile con un normale editor di testi, anche se naturalmente i caratteri diversi dall'ASCII standard potrebbero apparire non correttamente. Per i testi comuni utilizzati in occidente consente di mantenere più compatte le dimensioni dei files rispetto ad UCS in quanto la maggior parte dei caratteri sono ASCII. Inoltre, l'UTF-8 attraversa indenne, come se fosse un normale testo, anche i sistemi e i programmi che non supportano Unicode. Per questi motivi l'UTF è molto usato nella posta Internet ed in alcuni sistemi operativi[13].

Tra le codifiche UTF la più comune è la UTF-8, che tra l'altro è la codifica di default di XML.

[12] Il byte è un'unita di misura delle grandezze informatiche corrispondente ad un gruppo di 8 bit

[13] Cfr. FABIO UTILI, *I caratteri di testo e internet*, in *Home page of Fabio Utili* <http://digilander.libero.it/fabioutili/tavolASCII.html> 11 luglio 2003, (29 settembre 2003).

Tavola dei caratteri ASCII

DEC.	BINARIO	VALORE	DEC.	BINARIO	VALORE
000	00000000	NUL (Null char.)	064	01000000	@ (chiocciola)
001	00000001	SOH (Start of Header)	065	01000001	A
002	00000010	STX (Start of Text)	066	01000010	B
003	00000011	ETX (End of Text)	067	01000011	C
004	00000100	EOT (End of Transmission)	068	01000100	D
005	00000101	ENQ (Enquiry)	069	01000101	E
006	00000110	ACK (Acknowledgment)	070	01000110	F
007	00000111	BEL (Bell)	071	01000111	G
008	00001000	BS (Backspace)	072	01001000	H
009	00001001	HT (Horizontal Tab)	073	01001001	I
010	00001010	LF (Line Feed)	074	01001010	J
011	00001011	VT (Vertical Tab)	075	01001011	K
012	00001100	FF (Form Feed)	076	01001100	L
013	00001101	CR (Carriage Return)	077	01001101	M
014	00001110	SO (Shift Out)	078	01001110	N
015	00001111	SI (Shift In)	079	01001111	O
016	00010000	DLE (Data Link Escape)	080	01010000	P
017	00010001	DC1 (XON) (Device Control 1)	081	01010001	Q
018	00010010	DC2 (Device Control 2)	082	01010010	R
019	00010011	DC3 (XOFF)(Device Control 3)	083	01010011	S
020	00010100	DC4 (Device Control 4)	084	01010100	T
021	00010101	NAK (Negative Acknowledgement)	085	01010101	U
022	00010110	SYN (Synchronous Idle)	086	01010110	V
023	00010111	ETB (End of Trans. Block)	087	01010111	W
024	00011000	CAN (Cancel)	088	01011000	X
025	00011001	EM (End of Medium)	089	01011001	Y
026	00011010	SUB (Substitute)	090	01011010	Z
027	00011011	ESC (Escape)	091	01011011	[(par. quadra a sinistra)
028	00011100	FS (File Separator)	092	01011100	\ (back slash)
029	00011101	GS (Group Separator)	093	01011101	] (par. quadra a destra)
030	00011110	RS (Request to Send)	094	01011110	^ (caret)
031	00011111	US (Unit Separator)	095	01011111	_ (underscore)
032	00100000	SP (Space)	096	01100000	`
033	00100001	! (punto esclamativo)	097	01100001	A
034	00100010	" (virgolette doppie)	098	01100010	B
035	00100011	# (cancelletto)	099	01100011	C
036	00100100	$ (dollaro)	100	01100100	D
037	00100101	% (percento)	101	01100101	E
038	00100110	& (e commerciale)	102	01100110	F
039	00100111	' (virgolette singole)	103	01100111	G
040	00101000	((parentesi tonda a sinistra)	104	01101000	H
041	00101001	) (parentesi tonda a destra)	105	01101001	I
042	00101010	* (asterisco)	106	01101010	J
043	00101011	+ (più)	107	01101011	K
044	00101100	, (virgola)	108	01101100	L
045	00101101	- (meno)	109	01101101	M
046	00101110	. (punto)	110	01101110	N
047	00101111	/ (forward slash)	111	01101111	O
048	00110000	0	112	01110000	P
049	00110001	1	113	01110001	Q
050	00110010	2	114	01110010	R
051	00110011	3	115	01110011	S
052	00110100	4	116	01110100	T
053	00110101	5	117	01110101	U
054	00110110	6	118	01110110	V
055	00110111	7	119	01110111	W
056	00111000	8	120	01111000	X
057	00111001	9	121	01111001	Y
058	00111010	: (due punti)	122	01111010	Z
059	00111011	; (punto e virgola)	123	01111011	{ (par. graffa a sinistra)
060	00111100	< (minore)	124	01111100	| (barra verticale)
061	00111101	= (uguale)	125	01111101	} (par. Graffa a destra)
062	00111110	> (maggiore)	126	01111110	~ (tilde)
063	00111111	? (punto interrogativo)	127	01111111	DEL (delete)

I.4. I LINGUAGGI DI CODIFICA

La codifica dei caratteri, come sopra detto, rappresenta il livello più basso della codifica informatica di un testo[14]. Tuttavia a livelli rappresentazionali più accurati è possibile prendere in considerazione elementi relativi sia all'aspetto formale del testo, focalizzando l'attenzione sulle caratteristiche grafiche dello stesso quali l'allineamento, il tipo di carattere, l'interlinea e via di seguito tutti quei particolari tipici della rappresentazione tipografica di un testo; sia, d'altra parte, considerando il testo da un punto di vista più astratto, tener presente le strutture funzionali, la segmentazione logica, le partizioni interne del testo.

Per aggiungere al flusso dei caratteri costituenti il testo tutte le ulteriori informazioni relative a caratteristiche tipografiche o strutturali si ricorre a dei linguaggi di marcatura del testo, i cosiddetti markup language[15].

Grazie ad un linguaggio di markup vengono inserite insieme alla successione ordinata dei caratteri una serie di istruzioni, dette tag, ovvero delle etichette che definiscono determinate caratteristiche per la parola o il blocco di testo cui fanno riferimento.

Riporto un frammento di codice HTML, il linguaggio di marcatura per la creazione delle pagine web, per spiegare con un esempio pratico quanto detto:

---

[14] Infatti alcune delle prime esperienze di digitalizzazione di testi presenti su internet si basavano su formati testuali, in pratica il testo è poco più che una sequenza ordinata di caratteri e segni ortografici; a tal proposito merita di esser ricordato il progetto Gutenberg <www.gutenberg.net> nato nel 1971 dall'iniziativa individuale di Michael Hart. Il Progetto Gutenberg nel corso degli anni ha costituito, grazie all'apporto di numerosi volontari, un archivio che supera ormai i 6.000 testi. I titoli sono relativi ad opere non solo in inglese ma anche in francese, tedesco, spagnolo e italiano. I testi sono in formato ASCII a sette bit al fine di garantire l'universalità dell'accesso.
Per quanto riguarda l'Italia merita di essere menzionato il Progetto Manuzio <www.liberliber.it>. Gestito dall'associazione Liber Liber, attiva sin dal 1993, il Progetto Manuzio rende disponibili in rete numerosi testi della letteratura italiana ormai liberi dai diritti d'autore. Le opere in origine erano accessibili prevalentemente in formato txt, oggi si è dato spazio anche ad altri formati aperti quali: html, rtf ed alcuni formati e-book quali oeb e lit.

[15] "L'espressione markup deriva dalla analogia tra questi linguaggi e le annotazioni inserite da autori, curatori editoriali e correttori nei manoscritti e nella bozze di stampa di un testo al fine di indicare correzioni e trattamenti editoriali, chiamate in inglese mark up".
MARCO CALVO, FABIO CIOTTI, GINO RONCAGLIA, MARCO ZELA, *Come funziona World Wide Web*, in *Internet 2000* (versione online)
<http://www.laterza.it/internet/leggi/internet2000/online/testo/30_testo.htm>

```
<div align="center"><strong>questo e del testo in grassetto allineato al centro</strong></div>
```

intuitivamente tutti i comandi HTML (tag) sono facilmente riconoscibili: sono racchiusi tra i segni di minore e maggiore (**<** e **>**) e presentano un comando di apertura (ad esempio `<div align="center">`) ed uno di chiusura identificato dalla presenza dello slash / (ad esempio `</div>`), il testo contenuto fra i tag di apertura e chiusura è quello cui si applicano le istruzioni definite dal tag.

I.4.1. I programmi di text processing WYSIWYG

Un caso particolare di marcatura del testo è quella realizzata dai cosiddetti programmi di text processing WYSIWYG (acronimo inglese che sta per *What You See Is What You Get*, cioè "quello che vedi è quello che ottieni"), si tratta di programmi, primi fra tutti i word processor quali il celebre Microsoft Word, utilizzati per la stesura ed elaborazione di documenti a prevalente contenuto di testo nel quale le caratteristiche grafiche del documento appaiono sullo schermo così come saranno sulla pagina stampata.

Nel caso dei word processor la marcatura è invisibile all'utente, il quale si limita ad interagire con l'interfaccia del programma per definire le varie caratteristiche del testo; le etichette di marcatura vengono automaticamente inserite dal programma in relazione alle scelte dell'utilizzatore, in un formato spesso non intellegibile dall'utente, ma solo dal programma stesso; si suole per questo parlare di formati proprietari o chiusi, in quanto, un documento in un formato proprietario può esser visualizzato solo dalla stesso tipo di applicazione in cui è stato creato oppure da un apposito visualizzatore (quando disponibile). Come si potrà comprendere questo comporta grossi problemi per quanto riguarda la distribuzione del documento che potrà esser aperto solo su determinati sistemi e solo con specifici programmi.

I formati proprietari o chiusi vengo anche detti formati binari, per distinguerli dai cosiddetti formati aperti, in cui sia il contenuto testuale sia le informazioni di codifica sono memorizzate sotto forma di testo.[16]

I.4.2. Codifiche procedurali e codifiche dichiarative

Convenzionalmente si distinguono due tipologie di linguaggi di markup: i linguaggi procedurali e quelli dichiarativi.

I linguaggi procedurali, conosciuti anche come specific markup language, sono costituiti da istruzioni operative che , analogamente ai programmi, eseguono delle procedure computazionali sulle porzioni di testo cui fanno riferimento.

I linguaggi dichiarativi, detti anche generic markup language, invece, non attivano con i loro tag una procedura informatica, ma piuttosto specificano, dichiarano, la funzione astratta assolta dal blocco di testo cui si riferiscono.

I.4.3. Linguaggi di codifica procedurali

I linguaggi procedurali vengono abitualmente usati per codifiche di tipo presentazionale, finalizzate alla definizione precisa della resa grafica del documento.

Uno dei più comuni linguaggi di codifica procedurali è l'RTF (Rich text format, "formato di testo ricco"), realizzato dalla Microsoft, che ne è proprietaria, al fine di agevolare lo scambio di documenti fra diverse applicazioni.

L'RTF è un tipo di codifica del testo "tagged", in cui il documento viene codificato come puro testo, e degli "identificatori" (tag) indicano quale tipo di strutture di formattazione (allineamento, impaginazione, tipo di carattere) applicare al semplice testo.

[16] Propriamente si parla di markup language solo per quei linguaggi che si basano sul testo, sia per il markup sia per il contenuto. Si tenga presente che per "formato sotto forma di testo" si intende un file in cui le informazioni sono memorizzate esclusivamente come una sequenza di caratteri ASCII (o altra tavola di codifica dei caratteri) e visualizzabile per mezzo di qualsiasi editor di testi, quale può essere l'Edit del sistema operativo Microsoft DOS, oppure il Notepad (Blocco Note) del sistema operativo Microsoft Windows.

Per mezzo della marcatura il testo del documento RTF (definito sfruttando soltanto set di caratteri molto semplici e diffusi, tipo l'ASCII) viene arricchito di nuove informazioni riguardanti la formattazione, di cui i formati di puro testo sono privi: da ciò la spiegazione del nome di "formato di testo ricco".

I tag RTF sono in genere definiti dal carattere \ (back slash) seguito da alcune lettere che specificano il tag e da uno o più numeri nel caso il tag richieda un parametro numerico (ad esempio i tag che specificano la dimensione del carattere richiedono che la dimensione sia indicata in numeri); il tag deve essere separato dal testo a cui si applica da uno spazio, questo spazio però non apparirà nel testo stampato, il tag ed il testo a cui questo deve essere applicato sono racchiusi tra parentesi graffe { }.[17]

Un esempio di testo marcato con tag rtf è:

{\b Questo testo è in grassetto, {\i e questo è anche in corsivo.}}

Sebbene teoricamente possibile, i tag RTF non sono stati pensati perché il file possa essere letto o codificato da un operatore umano, bensì da un programma; nato come formato per lo scambio di documenti, oggi tutti i word processor più diffusi hanno la possibilità di importare e di esportare dei file in RTF.

Tra gli altri linguaggi di codifica procedurale si ricordano lo Script, il TROFF, il TEX. Particolarmente efficaci nel definire la resa grafica di un testo, i linguaggi di codifica procedurale, palesano un grave limite allorché si sposti la nostra analisi sul messaggio di cui il testo è veicolo. In una codifica procedurale informazioni indispensabili sulla natura funzionale dei blocchi testuali quali, ad esempio, le divisioni in sezioni o capitoli, la presenza di discorso diretto, citazioni etc. non sono rappresentate.

---

[17] DAVIDE BIANCHI, *Breve introduzione al formato* RTF, in *Articoli, documenti, faq e tutto quello che fa programmazione*
<http://www.soft-land.org/documenti/rtf.html>
21 Dicembre 2000, (1 Ottobre 2003).

I.4.4. Linguaggi di codifica dichiarativi

Un approccio completamente diverso rispetto alle codifiche procedurali è quello dei cosiddetti linguaggi dichiarativi.

Mentre i linguaggi di codifica procedurale dicono al computer come fare qualcosa, ossia come rendere graficamente un testo, i linguaggi di markup generico dicono al computer cosa è qualcosa, identificano la funzione strutturale dei blocchi testuali.

Allo stesso modo della codifica procedurale anche nella codifica dichiarativa si ricorre a tag per la marcatura del testo.

Pure le istruzioni (tag) dei generic markup language sono delle etichette, dei marcatori inseriti nel flusso del testo, costituiti da sequenze di caratteri delimitati da segni (caratteri) speciali (**<**…**>**) che hanno la funzione di permettere al computer di distinguere i tag dal testo.

Ecco un esempio di testo codificato con un linguaggio di codifica dichiarativo:

```
<capitolo n="1" id="I.1">
<titolo tipo="convenzionale">I</titolo>
<titolo tipo="tematico">L'ubriaco che sapeva auscultare la
terra</titolo>
<paragrafo> Dalle parti di <nome tipo="luogo.reale">Montedoro</nome>
dicono che fu un pastore, per caso.[…] </paragrafo>
[…]
</capitolo>
```

Un linguaggio basato su di una codifica di tipo procedurale si concentra sull'aspetto del documento, invece un linguaggio di codifica dichiarativa, rappresentando la struttura astratta del testo, si concentra sul messaggio.

Questo comporta degli indubbi vantaggi per quanto riguarda il trattamento informatico dei testi; ad esempio, un'applicazione per l'estrazione delle informazioni (information retrival) ci potrebbe permettere di sapere se e quante

volte determinate parole compaiono dentro un titolo, oppure pronunciate da un personaggio specificato.

Inoltre il markup generico presenta vantaggi anche per quanto riguarda la rappresentazione grafica dei testi, infatti grazie all'uso di strumenti noti come fogli di stile, è possibile associare, sul modello dello specific markup, delle istruzioni di resa grafica ai tag. Tutto questo si rivela particolarmente vantaggioso in virtù del fatto che si ha così la possibilità di ottenere numerosi output con caratteristiche tipografiche e di impaginazione anche molto varie tra di loro, che possono così soddisfare esigenze diverse, il tutto partendo da uno stesso documento sorgente.

Sono molteplici le possibilità che i linguaggi di marcatura offrono agli umanisti: "Se il linguaggio scelto è abbastanza potente, è possibile rappresentare il testo (i suoi elementi e i rapporti che intercorrono tra loro) a vari livelli. Si possono costruire vari modelli del testo: accanto alla struttura editoriale possiamo immaginare di codificare la sua struttura grammaticale (per esempio i tempi verbali), la struttura retorica (le figure, ma anche la distribuzione del discorso diretto e indiretto), i luoghi; e poi è solo questione di voglia e fantasia."[18]

I.4.5. SGML

Il padre dei moderni linguaggi di codifica dichiarativi è l'SGML *Standard Generalized Markup Language*.

Le origini dell'SGML risalgono alla fine degli anni sessanta. All'epoca il markup utilizzato nei documenti era di tipo procedurale, ciò comportava numerosi problemi all'interscambio di documenti tra macchine ed applicazioni diverse, in particolare i sistemi di tipocomposizione[19], macchine molto costose, utilizzavano ognuno il proprio metodo proprietario per descrivere l'impostazione tipografica di un documento. Di conseguenza, l'impostazione di un testo creato in un sistema non poteva esser trasferita in un altro. Ciò rendeva assai difficoltoso e costoso

[18] Cfr. GIGLIOZZI *Il testo e il computer. Manuale di informatica per studi letterari*, (vedi n.3) p.109.
[19] Si tenga presente che il riferimento è per un periodo che può esser considerato la preistoria dell'era informatica: basti pensare che all'epoca non esistevano i personal computers.

l'interscambio di documenti ed il riutilizzo di parte degli stessi.[20] Per queste ragioni si cominciò a pensare ad un nuovo tipo di linguaggio in cui le istruzioni sulla rappresentazione fossero mantenute separate dal contenuto del documento, che invece sarebbe stato marcato per mezzo di etichette che indicavano la struttura astratta delle varie parti del documento.

Una prima elaborazione di un markup di tipo generico si ebbe alla fine degli anni sessanta presso i laboratori della GCA *Graphic Communications Association* dove fu definito il "GenCode concept", che stabiliva che codici generici differenti erano necessari per documenti di tipo diverso. Inoltre sottolineava che i documenti più piccoli potevano essere incorporati come elementi di documenti più grandi. Si gettavano così le basi teoriche dei moderni linguaggi dichiarativi.

Il progetto si trasformò successivamente nel comitato GenCode che rivestì un ruolo strumentale nello sviluppo dello standard SGML.

All'incirca nello stesso periodo in cui la GCA elaborava GenCode, nasceva GML *Generalized Markup Language* , si trattava di un linguaggio realizzato nell'ambito di un progetto di ricerca della IBM sui sistemi integrati per la gestione dei documenti legali. A capo del progetto c'era l'ingegnere Charles Goldfarb , il quale insieme ai propri collaboratori Edward Mosher e Raymond Lorie, diede vita ad un sistema per consentire ai programmi di modifica di testi, di formattazione e di *information retrieval* la condivisione degli stessi documenti. Con GML veniva introdotto il concetto di un tipo di documento formalmente definito con una struttura annidata esplicita. GML era implementato su mainframe dell'IBM che lo adottò come standard di codifica per i propri documenti. Al completamento di GML Goldfarb continuò la sua ricerca sulle strutture nei documenti, creando nuovi concetti che non facevano parte di GML ma che successivamente furono introdotti in SGML.

[20] Cfr. ED TITTIEL, NORBERT MIKULA, RAMESH CHANDAK, *XML for Dummies*, Milano,Apogeo, 1998, pp. 43-46 (ed. orig. *XML for Dummies*, IDG Books Worldwide, Inc., 1998).

Nel 1978 l'*American National Standards Institute* (ANSI) istituì un comitato per la definizione di uno standard per la trasmissione e l'archiviazione di documenti. La direzione del comitato fu affidata a Goldfarb. Questo comitato fu supportato anche dal comitato GenCode e partendo da GML fu avviata la standardizzazione di un linguaggio per la descrizione di documenti che divenne SGML. La prima versione delle specifiche dello standard fu pubblicata nel 1980. Negli anni seguenti lo sviluppo, sempre sotto la direzione di Goldfarb, fu proseguito in seno all'ISO.

Nel 1986 SGML divenne uno standard ISO col nome di "ISO 8879: 1986 Information processing - Text and Office systems - Standard Generalized Markup Language (SGML)". [21]

Fu chiamato "Linguaggio di Marcatura Generalizzato standard", in quanto si trattava del primo sistema di codifica dichiarativa altamente astratto e generalizzato.

SGML è stato progettato con l'intento di favorire lo scambio di documenti fra diverse piattaforme informatiche , per tal motivo indipendentemente dall'hardware o dal sistema operativo utilizzato, è in grado di funzionare sui più vari dispositivi.

SGML è stato per così dire il capostipite dei linguaggi di marcatura in cui i tag sono sempre delimitati dalle cosiddette parentesi angolari, ossia tra '**<**' e '**>**', e sono caratterizzati da un nome (come ad esempio *capitolo*, *titolo*, ...) ed eventualmente accompagnati da attributi espressi come tante coppie NomeAttributo="valore".

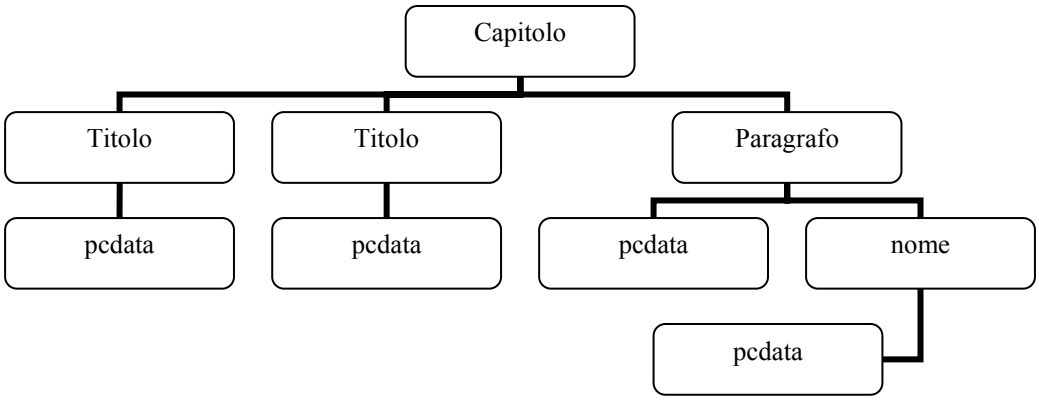
SGML non definisce direttamente i tag per la marcatura della struttura logica del testo, infatti più che di un vero e proprio linguaggio, si tratta di un metalinguaggio, esso infatti fornisce una serie di norme sintattiche astratte con cui definire le regole da applicare nella marcatura di un determinato tipo di documenti. SGML è una sorta di supergrammatica con cui definire la grammatica. SGML è un linguaggio per creare altri linguaggi, ognuno di questi linguaggi

[21] Per maggiori informazioni riguardo la storia di SGML si veda: SGML USERS' GROUP,*A Brief History of the Development of SGML*, in *The CoverPages*
<http://www.oasis-open.org/cover/sgmlhist0.html>
11 giugno 1990, (02 ottobre 2003).

costituisce un'applicazione SGML, una delle più note è HTML (Hypertext markup language) il linguaggio con cui si costruiscono le pagine Web.

Uno dei concetti teorici su cui si basa l'impostazione di tutto il linguaggio è quello di *tipo di documento.* Il tipo di documento descrive le caratteristiche di una classe di documenti strutturalmente omogenei, identificandone la struttura, gli elementi che compongono la struttura, le relazioni di dipendenza tra gli elementi, le relazioni di ricorrenza degli elementi. Visivamente un tipo di documento può essere rappresentato graficamente con un grafico ad albero che rappresenta la struttura del documento.

Ad esempio il codice di pag. 22 può esser così rappresentato:



La DTD (Document Type Definition) è lo strumento con cui si definisce il tipo di documento e, in sostanza, si crea un'applicazione SGML.

"Una DTD… mette a disposizione un modello per la marcatura di documenti che indica la presenza, l'ordine e la posizione degli elementi e dei loro attributi in un documento."[22]

Più precisamente nella DTD sono elencati e definiti tutti gli oggetti necessari all'elaborazione di un determinato tipo di documento:

- **Elementi**: i nomi dei marcatori adottati per identificare i blocchi strutturali costituenti l'albero dei dati.

[22] Cfr. DEVAN SHEPHERD, *XML. Guida completa*, trad. it. staff Apogeo, Milano, Apogeo, 2002, p. 54 (ed. orig. *Teach yourself XML in 21 Days second edition*, Sams, 2001).

- ***Content model***: il modello di contenuto per ciascun elemento indica quali altri elementi, quando ed in che numero, possono comparire dentro un elemento. Indica inoltre quali attributi possono essere associati ad un elemento. Gli attributi permettono l'aggiunta all'elemento di informazioni addizionali, che possono sia essere utili per descrivere più accuratamente la struttura dei dati rappresentata, sia esser veicolo di metainformazione, ovvero specificare caratteristiche e funzioni non propriamente strutturali. Gli attributi possono essere paragonati a degli aggettivi.

- **Entità**: le entità permettono di creare dei riferimenti a dati esterni, ad altri documenti, o a file grafici. I riferimenti a entità rappresentano segnaposti per altri valori o tipi di contenuto; molto usate sono le entità per rappresentare caratteri che non sono presenti nella tabella codici del documento o che si teme potrebbero essere rappresentati non correttamente nel passaggio da un sistema all'altro; spesso le entità sono utilizzate per rappresentare tutti i caratteri di un documento che non fanno parte della tavola ISO 646 IRV (l'ASCII a sette bit). Così come per essere distinti dal testo i tag degli elementi sono racchiusi tra le parentesi angolari, i nomi delle entità sono delimitati dalla e commerciale (&) e dal punto e virgola (;). Ad esempio È oppure È .

Riportiamo a titolo di esempio una ipotetica DTD per il codice di pag.22

```
<!ELEMENT capitolo
      (titolo+, paragrafo+) >

<!ATTLIST capitolo
      n CDATA #IMPLIED
      id ID #REQUIRED>


<!ELEMENT titolo
      (#PCDATA) >

<!ATTLIST titolo
      tipo (convenzionale|tematico|indefinito) "indefinito">
```

```
<!ELEMENT paragrafo
      (#PCDATA|nome)* >

<!ELEMENT nome
      (#PCDATA) >

<!ATTLIST nome
      tipo CDATA #IMPLIED>
```

Come si può vedere abbiamo una serie di dichiarazioni denominate *markup declaration.*

Ogni dichiarazione è aperta dal simbolo di minore e dal punto esclamativo (<!) cui segue una parola chiave che identifica il tipo di dichiarazione: ELEMENT per gli elementi, ATTLIST per gli attributi. Di seguito troviamo l'identificatore dell'elemento, in pratica il nome del tag. In fine racchiuso tra parentesi tonde si trova il modello di contenuto dell'elemento, costituito dall'elenco degli elementi figli che possono esser contenuti dall'elemento padre.

Gli elementi figli nel modello di contenuto possono essere separati da una virgola oppure da una barra (|), nel primo caso questo significa che gli elementi si devono necessariamente presentare nell'ordine definito dalla dichiarazione di markup, ad esempio nel nostro caso l'elemento paragrafo deve sempre seguire l'elemento titolo.

La barra invece indica che gli elementi possono presentarsi in qualsiasi ordine, per esempio l'elemento paragrafo può contenere caratteri[23] ed elementi nome in qualsiasi ordine.

Nelle dichiarazioni di elemento notiamo la presenza di simboli quali + oppure *, si tratta degli *indicatori di ricorrenza*, i quali specificano quante volte un elemento può esser presente dentro un altro. Il più '+' indica che un elemento ricorre una o più volte, l'asterisco '*' che l'elemento ricorre zero o più volte, il punto interrogativo '?' che l'elemento ricorre una volta o non ricorre affatto. Nel nostro esempio l'elemento *capitolo* deve contenere almeno un elemento *titolo* ed uno

[23] #PCDATA è la parola chiave utilizzata nelle DTD per descrivere elementi di base contenenti dati di tipo carattere, costituiti da testo non elaborato che potrebbe contenere riferimenti a entità, ma non altri elementi figlio o markup.

paragrafo, ma può averne anche altri, mentre l'elemento *paragrafo* può contenere un qualsiasi numero di elementi PCDATA e *nome*, ma può anche non contenere nessun elemento.

Leggermente diversa è la dichiarazione degli attributi. Dopo il delimitatore d'apertura ed il nome dell'elemento con cui è associato, si incontra il nome dell'attributo seguito dai valori possibili dello stesso, nel caso in cui si presenti un elenco di valori prestabiliti questi sono racchiusi tra parentesi tonda e separati dalla barra (|), è il caso dell'attributo *tipo* dell'elemento *titolo*; altrimenti si può incontrare una speciale parola chiave quale CDATA, che indica che il valore dell'attributo sarà costituito da dati di tipo carattere. Infine viene indicato il valore di default dell'attributo, valore che può consistere anche in una parola chiave tipo #IMPLIED che indica che l'attributo è facoltativo oppure #REQUIRED che specifica l'obbligatorietà dell'attributo.

I.4.6. HTML

Una delle più fortunate applicazioni SGML è HTML, acronimo che sta per Hyper-Text Markup Language e che può essere tradotto come linguaggio di marcatura per ipertesti.

L'inventore di HTML è stato Tim Berners-Lee, un ricercatore del CERN (Conseil Européenne pour la Recherche Nucléaire) di Ginevra, egli nel marzo del 1989 presentò ai dirigenti dei laboratori una relazione dal titolo "Information Management: a Proposal", in cui illustrava la sua idea di "un sistema universale di informazioni fra loro collegate, in cui la generalità e la portabilità sono molto più importanti della bella grafica"[24].

L'obiettivo di Berners-Lee era quello di sviluppare un sistema di pubblicazione e reperimento dell'informazione distribuito su rete geografica che tenesse in contatto

[24] TIM BERNERS-LEE, *Information Management: A Proposal*, in *W3 archive* <http://www.w3.org/History/1989/proposal.html> (06 ottobre 2003)

la comunità internazionale dei fisici, alla base dell'idea dello studioso stava il potente concetto di "ipertesto".

Secondo la definizione data nel 1965 dall'ideatore del termine, il filosofo e sociologo Theodor Holm Nelson, l'ipertesto è "scrittura non sequenziale, testo che si dirama e consente al lettore di scegliere: qualcosa che si fruisce al meglio davanti a uno schermo interattivo. Così come è comunemente inteso, un ipertesto è una serie di brani di testo tra cui sono definiti legami che consentono al lettore differenti cammini. [...]

L'ipertesto include come caso particolare la scrittura sequenziale, ed è quindi la forma più generale di scrittura. Non più limitati alla sola sequenza, con un ipertesto possiamo creare nuove forme di scrittura che riflettano la struttura di ciò di cui scriviamo, e i lettori possono scegliere percorsi diversi a seconda delle loro attitudini, o del corso dei loro pensieri, in un modo finora ritenuto impossibile."[25]

Per materializzare questa sua idea di un sistema di documenti collegati reticolarmente, Berners-Lee nel 1990 ideò un linguaggio di formattazione per creare ipertesti consultabili via rete, l'HTML, ed un protocollo[26] basato su TCP/IP[27], l'http (Hypertext Transer Protocolol), definì inoltre un nuovo metodo universale di indirizzamento di documenti in internet, l'URL (Universal Resource Locator). Nasceva così il *World Wide Web*, la più nota applicazione ospitata dalla rete internet, quasi un'incarnazione della rete stessa, tanto che spesso i termini web ed internet vengono usati come sinonimi[28]. Nell'idea originale di Berners-Lee obiettivo del nuovo sistema doveva essere il reperimento e la navigazione di informazioni tra loro interrelate in un tessuto complesso e non sequenziale di associazioni, non concentrandosi sull'aspetto grafico di queste, ma piuttosto sul loro contenuto; il

[25] Cfr. THEODOR HOLM NELSON, *Literary Machines 90.1*, Padova, Franco Muzzio Editore, 1992 ,pp. 0/2-0/3.

[26] In informatica, insieme di regole formali semantiche e sintattiche che regolano la comunicazione tra computer connessi in una rete.

[27] Il TCP/IP o Transmission Control Protocol / Internet Protocol è il protocollo su cui si basa l'intera rete Internet.

[28] In realtà la rete internet nasce nel 1966 nell'ambito del progetto Arpanet del Dipartimento della Difesa americano. Obiettivo del progetto era quello di garantire le comunicazioni anche in caso di attacco nucleare.

primo *browser* realizzato dallo scienziato aveva infatti una interfaccia a caratteri, tuttavia lo sviluppo del linguaggio seguì una strada ben diversa.

Un avvenimento epocale per lo sviluppo del Web come noi lo conosciamo oggi fu, nel 1993, la realizzazione da parte di Marc Andressen ed Eric Bina, dottorandi presso il National Center for Supercomputing Applications (NCSA) dell'Università dell'Illinois, della prima interfaccia grafica multipiattaforma per l'accesso ai documenti presenti su World Wide Web, il browser Mosaic. Esso divenne così famoso ed utilizzato, che venne creata da Marc Andreesen e Jim Clark la Mosaic Communications (successivamente ribattezzata come Netscape Communications Corp.), che nel 1994 rilasciò la prima versione del browser Netscape.

Il passaggio ai browser grafici rappresentò una importantissima svolta per quanto riguarda la storia di internet, programmi come Netscape rendevano molto semplice la navigazione e favorirono un vero e proprio boom del WWW, che in quegli anni iniziava a non esser più soltanto prerogativa del mondo accademico, ma si diffondeva rapidamente nella società intera.

Si passava allora dall'ipertesto puro del Web concepito nei laboratori del CERN, verso una nuova impostazione di tipo ipermediale, immagini e suoni cominciavano a rappresentare un nuovo fulcro nell'esperienza della navigazione in internet.

HTML in origine non doveva essere un linguaggio per il "*rendering*" nei browser, ma per la marcatura dei dati in modo da consentirne la condivisione su reti e su piattaforme diverse. Come conseguenza della nuova linea di sviluppo della rete occorsero revisioni del linguaggio maggiormente incentrate sull'aspetto del documento.

La prima versione di HTML non ebbe mai un documento di riferimento ufficiale fu costruita e migliorata tra i vari ricercatori impegnati nello sviluppo del Web. La prima versione ufficiale di HTML fu la 2.0 pubblicata nel novembre 1995 a cura di Tim Berners-Lee e Dan Connolly per conto dell'Internet Engineering Task Force (IETF).

Sin dalle prime revisioni, sono stati inseriti marcatori speciali con un senso tipografico, come il corsivo <*I*>, il grassetto <*B*>, le interruzioni di riga <*BR*>, in seguito si è aggiunta la possibilità di inserire nel documento immagini e tabelle. La versione 3.2 di HTML includeva l'utilizzo di *Javascript*, un linguaggio di scripting atto a favorire l'interattività con le pagine web, la versione 4.0 comprendeva le specifiche per il DHTML (*Dynamic HTML*), nel 1996 fu rilasciata le prima versione delle specifiche per i CSS (Cascading Style Sheets), i fogli di stile a cascata, si tratta di strumenti di formattazione per le pagine HTML attraverso i quali è possibile definire numerosi aspetti riguardanti la resa grafica dei documenti sul web.

In seguito al grande successo avuto dal World Wide Web, nell'ottobre del 1994 Tim Berners-Lee fondò il World Wide Web Consortium (W3C) presso il Laboratory for Computer Science del Massachusetts Institute of Technology, in collaborazione con il CERN, dove il web è nato, e con il supporto del Dipartimento della Difesa americano (Defense Advanced Research Project Agency, DARPA) e della Commissione Europea.

Il W3C è costituito da un consorzio di aziende del settore informatico che si occupa di stabilire standard di riferimento per il Web. Obiettivo del W3C è quello di favorire l'interoperabilità e la standardizzazione di tutte le tecnologie riguardanti il web e per mezzo di documenti pubblici detti "raccomandazioni" ufficializzarne l'utilizzo.

I.5. XML

Il linguaggio di markup HTML, insieme al protocollo http, è stato la colonna portante su cui ha basato il proprio sviluppo il World Wide Web, tuttavia ha ben presto mostrato alcuni importanti limiti in relazione all'evoluzione della rete.

Nato come linguaggio per l'interscambio di documenti di testo reticolarmente collegati fra di loro per mezzo di link ipertestuali, HTML è stato negli ultimi anni piegato ed adattato come strumento per la visualizzazione di dati. Tuttavia la potenza rappresentazionale di HTML non è elevata, all'atto della sua creazione non c'era l'intenzione di includervi aspetti stilistici, basti pensare che anche marcatori per stili assai comuni come il grassetto (<*B*>) o il corsivo (<*I*>) sono stati aggiunti solo in seguito. L'impossibilità di avere un controllo puntuale della visualizzazione è un importante limite di HTML, è alquanto problematico progettare pagine web per monitor con risoluzioni diverse, inoltre nel caso si desideri fornire una versione stampabile della stessa pagina occorre crearla ex novo.

HTML è un linguaggio di rappresentazione chiuso che permette di scegliere soltanto tra un insieme prefissato di elementi. Il linguaggio manca della flessibilità sufficiente a descrivere tipi differenti e specifici di informazioni e non permette la pubblicazione di un singolo insieme di informazioni su supporti diversi oppure la trasmissione delle informazioni in differenti formati.

Come ha giustamente notato Bill Gates in una intervista nel 2001 "Le pagine web sono ancora un'immagine, una rappresentazione dei dati e non sono i dati sottostanti"[29].

Ci si è presto accorti della necessità di una separazione del contenuto dallo stile. Questa separazione è la principale caratteristica di SGML, il progenitore di HTML, tuttavia SGML, sebbene flessibile e potente, è eccessivamente complesso, inoltre

[29] LAURA MASSACRA, *Come sarà il cybermondo del 2001. Bill Gates. Bilanci e previsioni sul futuro di Internet*, in *Mediamente Biblioteca Digitale*
<http://www.mediamente.rai.it/biblioteca/prov/001229gates.asp>
(13 ottobre 2003).

per via di certe caratteristiche del linguaggio, SGML può essere assai esigente in termini di risorse computazionali e tempo di elaborazione. Su Internet la velocità di elaborazione è molto importante, un formato di documento deve essere progettato in modo che il navigatore possa acquisire ed elaborare velocemente i documenti.

Per tutti questi motivi nacque XML (Extensible Markup Language), XML è una revisione/semplificazione del linguaggio SGML. Il gruppo di lavoro originale si era prefisso un obiettivo di "80/20", ossia l'ottanta percento delle funzionalità di SGML con solo il venti percento della sua complessità.

Il progetto XML prese l'avvio nel 1996 in seno al World Wide Web Consortium (W3C), lo sviluppo fu curato dal XML Working Group (originariamente noto come SGML Editorial Review Board), esso era presieduto da Jon Bosak della Sun Microsystems con la partecipazione attiva dell'XML Special Interest Group (precedentemente noto come SGML Working Group) anch'esso organizzato dal W3C.

Nel febbraio del 1998, dopo oltre un anno di lavoro, le specifiche sono state rilasciate come raccomandazione ufficiale, con il titolo *Extensible Markup Language (XML) 1.0*.

Gli obiettivi progettuali di XML, leggiamo nelle specifiche, sono:

1. *XML deve essere utilizzabile in modo semplice su Internet*.
2. *XML deve supportare un gran numero di applicazioni.*
3. *XML deve essere compatibile con SGML.*
4. *Deve essere facile lo sviluppo di programmi che elaborino documenti XML.*
5. *Il numero di caratteristiche opzionali deve essere mantenuto al minimo possibile, idealmente a zero.*
6. *I documenti XML dovrebbero essere leggibili da un uomo e ragionevolmente chiari.*
7. *La progettazione XML dovrebbe essere rapida.*
8. *La progettazione XML deve essere formale e concisa.*

9. *I documenti XML devono essere facili da creare.*

10. *Non è di nessuna importanza l'economicità nel markup XML.*[30]

Come SGML, da cui deriva, XML è un metalinguaggio che fa dell'astrattezza e della generalità la sua forza, cito un documento del W3C: "XML è un metodo per mettere dati strutturati in un file di testo. Per "dati strutturati" pensate a cose come fogli elettronici, agende, parametri di configurazione, transazioni finanziarie, disegni tecnici, ecc. I programmi che producono questi dati li salvano anche su disco, per cui possono usare sia il formato binario o sia quello testuale. Quest'ultimo permette, se necessario, di guardare i dati senza l'ausilio del programma che li ha creati. XML è un insieme di regole, di linee guida, convenzioni, in qualsiasi modo li vogliate chiamare, per progettare file di testo per questi dati, in un modo che produca file che siano facili da generare e leggere (da parte di un computer), che siano non ambigui, e che tenga lontano da pericoli comuni, come mancanza di estensibilità, mancanza del supporto per l'internazionalizzazione/localizzazione, e dipendenza dalla piattaforma [...] Come l'HTML, XML fa uso di *tag* (parole racchiuse tra '**<**' e '**>**') e *attributi* (della forma name="value"), ma mentre l'HTML specifica cosa ogni tag & attributo significhi (e spesso come un testo tra essi apparirà in un browser), XML usa i tag solo per delimitare pezzi di dati, e lascia l'interpretazione dei dati completamente all'applicazione che li legge. In altri termini, se vedi "<*p*>" in un file XML, non è detto che sia un paragrafo. A seconda del contesto, può essere un prezzo, un parametro, una persona, una p…"[31]

XML eredita da SGML il concetto di tipo di documento e utilizza ancora le Document Type Definition DTD per definire le regole sintattiche con cui

[30] WORLD WIDE WEB CONSORTIUM, *Extensible Markup Language (XML) 1.0*, traduzione ufficiale delle specifiche XML 1.0 effettuata da Andrea Marchetti <http://www.w3c.it/traduzioni/xml-19980210/REC-xml-19980210-it.html>

[31] BERT BOS, *XML in 10 punti*, trad. it. Emiliano Tellina, in *Latoserver.it* <http://www.latoserver.it/XML/in10punti/> (ed. orig. *XML in 10 points*, <http://www.w3.org/XML/1999/XML-in-10-points>)

rappresentare la struttura dati.[32] XML è retrocompatibile con SGML , un documento XML valido è anche un documento SGML valido, d'altra parte è spesso abbastanza semplice la conversione di un documento SGML in XML.

Un concetto nuovo di XML è quello di "ben formato". Un documento SGML per poter essere utilizzato deve essere di norma associato ad una DTD che ne specifica i vincoli sintattici, se il documento rispetta le norme stabilite nella DTD è detto "valido". A differenza di SGML, XML permette anche di distribuire documenti non associati ad una DTD, tali documenti sono detti "ben formati".

Per essere ben formata una istanza XML deve rispettare alcuni vincoli formali:

- Deve contenere un unico elemento, detto radice, che contiene tutti gli altri elementi.
- Gli elementi contenuti nell'elemento radice possono avere figli, nipoti, pronipoti, etc. ma questi devono essere annidati correttamente, in sostanza i tag vanno chiusi sempre nell'ordine dall'ultimo aperto fino ad arrivare al primo, per esempio questa porzione di codice <*tag1*><*tag2*>testo</*tag1*></*tag2*> non è corretta, il giusto annidamento è <*tag1*><*tag2*>testo</*tag2*></*tag1*>, occorre chiudere prima l'elemento tag2 e poi tag1.
- I nomi di elementi, attributi ed entità sono *case sensitive*, ossia sensibili alla differenza tra maiuscolo e minuscolo, in XML <*Tag*> e <*tag*> sono due elementi diversi.
- Tutti gli elementi devono necessariamente presentare sia il tag di apertura sia quello di chiusura, non sono consentite minimizzazioni come in SGML, inoltre i tag vuoti, ovvero quei tag che non vengo utilizzati per marcare del testo, devono essere chiusi con "/>", ad esempio <*tag3*/>.
- I valori di attributo vanno racchiusi fra virgolette.
- I nomi degli elementi e degli attributi posso contenere solo caratteri alfanumerici (lettere da "a" a "z" e da "A" a "Z" e numeri da 0 a 9), possono

[32] Le DTD XML mancano di alcuni elementi ed opzioni delle DTD SGML

inoltre contenere tre caratteri di punteggiatura:il carattere di sottolineatura o underscore "_" , il trattino "-" ed il punto "."; in realtà nei nomi delle strutture XML sono consentiti anche i due punti ":" tuttavia sono riservati per i Namespace XML. I nomi di tag XML non possono contenere spazi, né iniziare con un trattino, un punto o un numero. Possono invece iniziare con le lettere a-z e A-Z e il carattere di sottolineatura.

- Il mark-up è separato dal contenuto testuale mediante caratteri speciali: le parentesi angolari "< >" per i tag e la e commerciale "&" per le entità, quando questi segni vanno inseriti come elementi del testo, allora vanno codificati per mezzo delle entità predefinite XML: "&" per la e commerciale "&", "<" per il simbolo di minore "<", ">" per il simbolo di maggiore ">". Xml ha altre due entità predefinite ' per le virgolette singole (') e """ per le virgolette doppie (").

Una istanza XML per essere valida oltre a rispettare i vincoli imposti dalla sua DTD deve anche essere ben formata, l'analisi della validità e della "corretta formazione" del documento è affidata a software detti parser.

Tutti questi vincoli sono stati posti ai documenti XML al fine di ridurre al minimo la difficoltà di implementazione di questa tecnologia nelle applicazioni, e facilitare l'apprendimento del linguaggio.

Le istanze XML sono costituite da documenti di testo codificati secondo le tabelle caratteri Unicode/ISO10646. Di default la codifica dei documenti XML è UTF-8

UTF-16. In XML è possibile definire qualsiasi carattere come entità numerica facendo riferimento alla posizione dei caratteri nelle tavole di codifica Unicode/ISO10646, le entità numeriche sono identificate dalla combinazione &#numero_nella_tavola_Unicode; ad esempio "È" sta per il carattere È.

XML è una tecnologia di pubblico dominio, non occorre pagare licenze ad alcuno per il suo utilizzo, disponibile per tutte le implementazioni, la sua adozione da parte di chiunque è completamente gratuita ed anche le tecnologie componenti sono di pubblico dominio.

Grazie alla sua indipendenza dalla piattaforma informatica, alla sua astrattezza e generalità nella rappresentazione dei dati, XML si è ormai avviato a diventare il formato universale per l'interscambio di dati, basti pensare che la prossima suite Office della Microsoft adotterà XML come formato per i propri file[33]. Ben si esprimeva Tim Bray,uno dei collaboratori alla stesura delle specifiche XML 1.0, quando definiva XML come l'ASCII del futuro.

I.5.1. XSL

Una delle caratteristiche principali di XML è la separazione dei dati dalla loro rappresentazione. Per formattare e visualizzare i dati contenuti in un file XML si ricorre ad XSL.

Con l'acronimo XSL (Extensible Stylesheet Language) si è soliti indicare una famiglia di tecnologie sviluppate, e formalizzate da raccomandazioni ufficiali del W3C, finalizzate alla rappresentazione e trasformazione di dati strutturati per mezzo di XML.

Lo sviluppo di XSL prese l'avvio nel 1997 in seno al W3C, in origine gli autori[34] usavano l'acronimo XSL per *Extensible Style Language*, ma ben presto la denominazione è divenuta *Extensible Stylesheet Language*, definizione che può essere tradotta come "linguaggio espandibile per fogli di stile". Predecessore diretto di XSL è il *Document Style Semantics and Specification Language* (DSSSL), un linguaggio utilizzato soprattutto in ambito SGML per manipolare documenti contenenti dati strutturati e convertirli in un formato adatto alla visualizzazione o alla stampa. Quella dei DSSSL è una tecnologia molto onerosa dal punto di vista della risorse computazionali richieste e dei costi di licenza; DSSSL, anche a causa della difficoltà di implementazione, non ha avuto una grande diffusione, è stato usato soltanto in alcune applicazioni associate a sistemi per la composizione tipografica di alta qualità. XSL ricava soltanto alcune sue caratteristiche da DSSSL,

[33] Nel momento in cui scriviamo non è stata ancora rilasciata alla vendita la versione di Microsoft Office con supporto XML.
[34] Gli Autori della proposta iniziale di XSL sottoposta al W3C nell'agosto del 1997 sono: Microsoft Corporation, Inso Corporation, ArborText, University of Edinburg e James Clark.

a differenza di XML con SGML. XSL non può essere considerato un sottoinsieme di DSSSL in quanto certe caratteristiche di XSL non esistono in DSSSL.

Durante il lavoro di sviluppo di XSL, col crescere delle specifiche, è divenuto evidente che "l'attribuzione di stili a documenti XML comporta due processi distinti, anche se strettamente correlati. Il primo comporta una trasformazione strutturale, in cui gli elementi vengono selezionati, raggruppati e riordinati; il secondo genera spesso un processo di formattazione che produce la semantica di rendering per specifici agenti utente che presentano i documenti su schermo, su carta, in voce o con altri mezzi"[35]. In sostanza ci si è resi conto che con i fogli di stile sono possibili due tipi di approccio per la rappresentazione dei dati, da una parte si possono associare delle regole di formattazione agli elementi XML, dall'altra si è visto come sia anche possibile creare, a partire dai dati del sorgente XML, un documento completamente nuovo che raffigura i dati in una nuova struttura. Per queste ragioni nelle prime bozze delle specifiche XSL si parlava, a proposito delle istruzioni che avrebbero dovuto esser presenti in un foglio di stile, di *norme di stile* e *norme di costruzione*. Per norma di stile si intende la specifica di un modello e di un'azione che la norma applica quando viene individuato il modello specificato, in pratica si stabilisce che ogni qual volta nel documento XML è presente un determinato elemento occorre intraprendere un'azione specificata (per lo più l'applicazione di regole di formattazione). La norma di costruzione invece comporta la specifica di un modello, che fa riferimento agli elementi del sorgente XML, e che attiva la creazione di un nuovo elemento quando viene individuato il modello specificato.

Nel corso della definizione delle specifiche tale distinzione è stata abbandonata a favore della divisione dello sviluppo di XSL in tre filoni separati, che ha poi prodotto tre distinte raccomandazioni: una per un linguaggio per trasformare documenti XML (XSLT), una per un vocabolario per formattare documenti XML (XSL-FO) ed una per un linguaggio per definire parti di un documento XML

[35] DEVAN SHEPHERD, *XML Guida completa* (vedi n.22) p.294.

(XPath), perciò oggi XSL consta di tre specifiche distinte: XSLT, XSL-FO[36] e XPath[37].

Le raccomandazioni ufficiali di XSLT e XPath sono state rilasciate nel novembre del 1999, mentre quelle di XSL-FO nell'ottobre del 2001.

Come detto XSL è composto da XSL Formatting Objects (XSL-FO), XML Path (XPath) e XSL Transformations (XSLT), grazie a questi tre strumenti XSL definisce regole per specificare come estrarre informazioni da un documento XML e come formattare queste informazioni in modo da poterle visualizzare.

I.5.2. XSL-FO

XSL Formatting Objects (XSL-FO) è un vocabolario XML[38] che definisce dei tag, detti Formatting Objects, che sono adoperati per specificare come visualizzare un documento XML. I documenti XSL-FO vengono utilizzati in accoppiata con il Formatting Object Processor (FOP), una applicazione software la quale si occupa di effettuare il rendering (la formattazione grafica) del documento e produrre un nuovo documento di output adatto alla visualizzazione. I file XSL-FO possono essere usati in associazione ad un processore FOP per creare, ad esempio, documenti Adobe PDF, oppure in numerosi formati grafici o ancora per inviare l'output direttamente ad una stampante.

XSL-FO è assai potente per la formattazione di alta qualità tuttavia è una tecnologia ancora giovane e per questo ad oggi non molto diffusa.

I.5.3. XPath

XML Path (XPath) è un linguaggio di espressione utilizzato per accedere a parti di un documento XML e farvi riferimento. Abbiamo rilevato che un documento XML può essere rappresentato come una struttura ad albero con un nodo radice ed una

[36] In realtà il nome della specifica è XSL 1.0 tuttavia è invalso l'uso di definire questa tecnologia XSL-FO.

[37] XPath è un linguaggio per identificare parti di un documento XML, sviluppato come parte integrante di XSLT, dove è ampiamente utilizzato, è stato adottato anche da altri linguaggi correlati ad XML, per tale motivo il W3C ha separato lo sviluppo di XPath da quello di XSLT.

[38] Un vocabolario XML è un insieme di tag XML definiti per un determinato scopo. Ad esempio XHTML (una riscrittura in XML dei tag HTML) è un altro caso di vocabolario XML.

serie di nodi figli. XPath offre gli strumenti per identificare nodi specifici in un albero di documento XML. La sintassi di XPath usa una notazione a percorso simile a quella usata per la risoluzione degli URL degli indirizzi Internet. Una espressione XPath può essere usata nella manipolazione di stringhe, nei calcoli numerici e nella logica boleana.

I.5.4. XSLT

Il cuore della tecnologia XSL è rappresentato da XSL Transformations (XSLT). XSLT è un linguaggio[39] utilizzato per manipolare documenti XML. XSLT estrae i dati da un documento XML per produrre un nuovo documento. Il processo di creazione di un nuovo documento a partire da un sorgente XML è detto *trasformazione*, questo processo non cambia il documento originale. XSLT è un linguaggio di programmazione dichiarativo. Nella programmazione di tipo tradizionale si stabiliscono accuratamente i passi che il computer deve eseguire; nella programmazione dichiarativa si dice al computer cosa fare, invece di come fare qualcosa. La programmazione dichiarativa è un tipo di programmazione guidata dai dati, in cui il computer è istruito su quali operazioni intraprendere allorché incontra determinati dati. La sintassi di XSLT è basata su modelli, detti *template*, che fanno riferimento per mezzo di espressioni XPath agli elementi del documento sorgente. I template attivano determinate azioni, quando incontrano nel sorgente gli elementi individuati dall'espressione XPath. Per operare XSLT si appoggia ad un processore; un processore è un'applicazione software che applica l'XSLT all'XML. Il processore lavora assieme al parser che in primo luogo analizza la sintassi dell'XML e dell'XSLT e se questa è corretta, ossia i documenti sono ben formati ed eventualmente, se associati ad una DTD, validi, permette la trasformazione. Quando viene attivato, il processore XSLT inizia a leggere il sorgente XML ed il documento XSLT, quest'ultimo contiene le regole per trasformare il documento

[39] Come XSL-FO, XSLT è formalmente un vocabolario XML, si presenta infatti come un insieme di tag XML, tuttavia a differenza di XSL-FO, i suoi tag non dicono ad un programma come visualizzare qualcosa, ma piuttosto cosa fare quando incontra un certo tag.

originale in un nuovo documento. Le regole sono espresse come insieme di modelli messi in relazione con gli elementi del documento XML originale. Ogni modello contiene espressioni XPath che indicano al processore quali siano il nodo o il gruppo di nodi obiettivo nel documento sorgente. Quando si verifica una corrispondenza, le regole del modello vengono applicate al contenuto dell'elemento corrispondente trovato. Il processo prosegue fino a quando tutti i modelli e tutte le corrispondenze sono state elaborate correttamente.

Il codice di pag. 22 può anche esser considerato una istanza XML ben formata, grazie ad XSLT può essere trasformato in un documento HTML adatto alla visualizzazione. Di seguito riportiamo, a titolo di esempio, il codice XSLT che permette ciò:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet                                          version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
      <xsl:output method="html"/>

<xsl:template match="/">
      <html>
      <head>
            <title>
Matteo Collura BALTICO capitolo <xsl:value-of select="/capitolo/@id"/>
            </title>
      </head>
            <body bgcolor="#FFFFFF">
                  <xsl:apply-templates/>
            </body>
            </html>
</xsl:template>

<xsl:template match="titolo[@tipo='convenzionale']">
            <div align="center">
                  <h1>
                        <xsl:apply-templates/>
                  </h1>
            </div>
</xsl:template>

<xsl:template match="titolo[@tipo='tematico']">
            <div align="center">
                  <h2>
                        <xsl:apply-templates/>
                  </h2>
                  <hr/>
            </div>
            <br/>
</xsl:template>

<xsl:template match="paragrafo">
            <p>
```

```
                <xsl:apply-templates/>
            </p>
        </xsl:template>

<xsl:template match="nome">
            <i>
                <xsl:apply-templates/>
            </i>
</xsl:template>

</xsl:stylesheet>
```

Fulcro di XSLT sono i modelli identificati dal tag xsl:template, ogni tag template è accompagnato da un attributo match="*espressione_XPath*" il quale contiene l'espressione XPath che mette in corrispondenza le regole del modello con gli elementi del sorgente XML. All'interno di ogni template sono contenute le regole del modello, nel nostro caso i tag HTML necessari per produrre l'output voluto. L'elemento `<xsl:apply-templates>` dice al processore di invocare i modelli per gli elementi figli dell'elemento selezionato e solo allora di concludere l'elaborazione del modello.[40]

Ecco l'output prodotto dal foglio XSLT dell'esempio applicato al sorgente XML di pag.22 utilizzando il processore Istant Saxon 6.22:

```
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-
8">

      <title>Matteo Collura BALTICO capitolo I.1</title>
   </head>
   <body bgcolor="#FFFFFF">

      <div align="center">
         <h1>I</h1>
      </div>

      <div align="center">
         <h2>L'ubriaco che sapeva auscultare la terra</h2>
         <hr>
      </div><br>

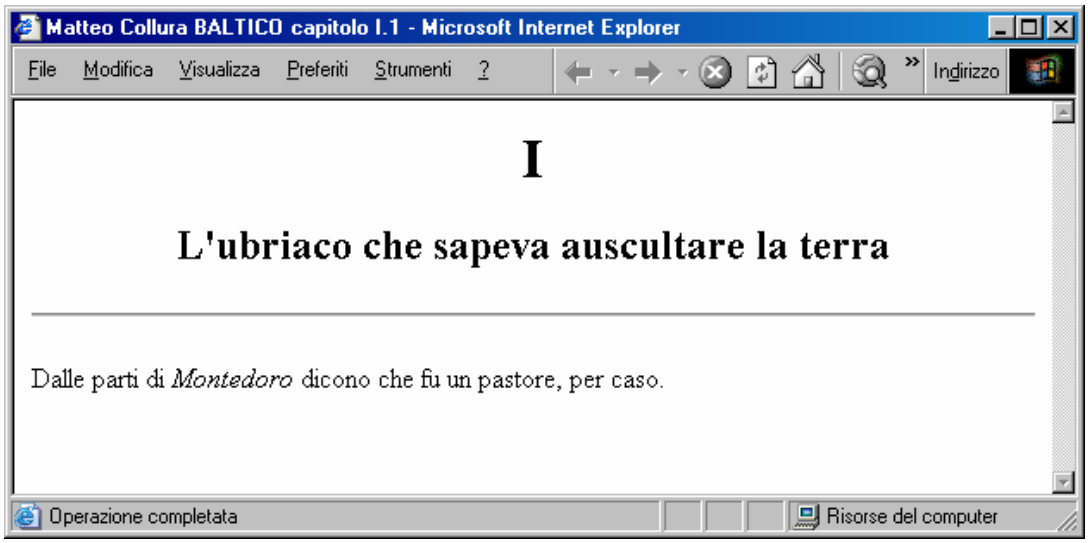
      <p> Dalle parti di <i>Montedoro</i> dicono che fu un pastore,
per  caso.
      </p>

   </body>
```

[40] Nell'esempio ho dato soltanto una panoramica minima delle funzioni XSLT, non essendo obiettivo di questo lavoro una illustrazione completa del linguaggio.

```
</html>
```

Come si può vedere i dati del file XML sono stati inseriti in una struttura completamente nuova. Il precedente documento è un sorgente HTML, ecco di seguito come questo viene visualizzato da un browser quale Internet Explorer 6.0:



Nella trasformazione dei dati dei documenti XML, XSLT consente una serie di attività di manipolazione dei dati stessi.

- **Ordinamento**: consente di modificare l'ordine degli elementi in base a vari criteri. Per esempio avendo un documento XML contenete un elenco di nomi è possibile mandarli in output ordinati alfabeticamente.
- **Filtro**: il linguaggio permette di rimuovere gli elementi non applicabili ad un contesto specifico. Ad esempio si potrebbe creare l'indice di un libro mandando in esecuzione solo il titolo di ogni capitolo.
- **Calcolo**: XSLT consente di eseguire funzioni aritmetiche con i dati del sorgente XML.
- **Unione**: permette di combinare in un singolo documento più documenti.

Il grosso vantaggio di XSLT è che consente di creare a partire da un unico sorgente XML innumerevoli output atti a soddisfare le esigenze più diverse, ad esempio possiamo ottenere un file contenente tag XSL-FO per formattare i dati del

documento oppure un file HTML o XHTML per permettere la visualizzazione dei dati con un browser web, ma gli stessi dati possono anche essere utilizzati per inviare le informazioni a un programma di sintesi del discorso che legge il testo a una persona con problemi visivi. Oppure, è anche possibile creare l'output per un dispositivo di lettura del Braille. Sono innumerevoli le possibilità offerte dai fogli di stile al trattamento dei dati. Grazie ad XML ed XSLT si è finalmente attuata la separazione dei dati dalla loro rappresentazione, ciò produce effetti positivi in numerosi campi applicativi, per tal motivo queste tecnologie si vanno sempre più affermando, candidandosi seriamente a diventare il formato universale per l'interscambio dei dati, "l'ASCII del ventunesimo secolo".

Nel momento in cui scriviamo è incorso di elaborazione una evoluzione-revisione del linguaggio XSLT, la versione 2.0.

I.5.5. CSS

Un altro linguaggio per la visualizzazione di documenti XML è quello dei CSS (Cascading Style Sheets - Fogli di stile a cascata), nato per l'utilizzo con HTML, viene in taluni casi usato anche per associare regole di formattazione a documenti XML. A differenza di XSLT, i CSS si limitano ad associare norme di visualizzazione ai tag XML senza produrre un nuovo documento. La prima specifica ufficiale di CSS (CSS1) risale al dicembre del 1996. Nel maggio 1998 è stata rilasciata la seconda versione CSS2 in cui sono state effettuate numerose aggiunte alla prima versione del linguaggio. È attualmente in corso di sviluppo una nuova specifica CSS3.

I.5.6. Tecnologie correlate a XML

Attorno ad XML è fiorita tutta una famiglia di tecnologie ad esso correlate. *XML Schema Definition (XSD)* è un linguaggio che, analogamente alle definizioni dei tipi di documento (DTD), consente di descrivere la struttura dei documenti XML in termini di elementi attributi e dati in essi contenuti. A differenza del linguaggio

delle DTD, XSD è espresso usando una sintassi XML[41], inoltre un ulteriore vantaggio di XSD è che permette di specificare il tipo di dati da utilizzare negli elementi. In realtà, in ambito prettamente umanistico non sono particolarmente rilevanti i vantaggi portati da XML Schema Definition rispetto alle DTD. *XML Link Language (Xlink)* è un linguaggio che descrive un modo standard per inserire collegamenti ipertestuali in un file XML, mentre HTML consente soltanto collegamenti unidirezionali nei quali l'unico modo per tornare alla pagina di partenza è quello di utilizzare l'apposito pulsante del browser, Xlink permette link multi direzionali grazie ai quali un utente può tornare indietro utilizzando lo stesso link che gli ha permesso di arrivare a destinazione. Consente inoltre Link a destinazione multipla per mezzo dei quali, un utente, partendo da un singolo link può scegliere fra differenti destinazioni. Sono molte altre le opzioni che Xlink permette di applicare ai collegamenti ipertestuali, tuttavia ad oggi il linguaggio non è ampiamente supportato dai browser, pertanto per il momento il suo uso e la sua diffusione sono estremamente limitati. *XML Pointer Language* (*Xpointer)* offre gli strumenti per indirizzare e localizzare informazioni all'interno di un documento XML. *XML Data Island* viene utilizzato per integrare dati XML all'interno di pagine HTML. Meritano anche una breve menzione anche alcuni dei numerosi vocabolari XML sviluppati in questi ultimi anni. *MML (Matematical Markup Language)* utilizzato per la definizione di formule matematiche complesse. *CML (Chemical Markup Language)* un linguaggio per la rappresentazione della struttura delle molecole. *SVG (Scalable Vector Graphics)* con cui è possibile disegnare grafici a due dimensioni usando il markup. *XBRL (eXtensible Business Reporting Language)* è un linguaggio che usa XML per la descrizione degli estratti conto.

Vi sono numerose altre tecnologie figlie di XML ma darne un elenco completo sarebbe tedioso e non opportuno in questa sede. Quanto detto basti per rendersi conto dell'attività di sviluppo che ferve attorno ad XML e per comprendere come

[41] XSD si presenta come un vocabolario XML.

ormai questa tecnologia si sia avviata a divenire un caposaldo in numerosi ambiti applicativi, non ultimo quello del trattamento informatico dei testi letterari.

PARTE SECONDA

II.1. E-BOOK

Il Termine e-book, libro elettronico, deriva dalla contrazione delle parole inglesi *electronic* e *book*.

Nei documenti il termine può essere incontrato indifferentemente col prefisso "e" di electronic unito direttamente al resto della parola, "ebook", oppure separato dal termine book da un trattino, "e-book". In Italia si preferisce la notazione "e-book"[42] con trattino, mentre in ambiente anglosassone prevale quella di "ebook" senza trattino.

Il termine e-book viene spesso utilizzato indistintamente per indicare dei concetti, sebbene tra di loro strettamente collegati, nettamente differenti:

- **E-book**: il libro elettronico vero e proprio, ne daremo nel prosieguo del discorso una definizione precisa.
- **E-book reader device**: il dispositivo materiale grazie al quale si rende possibile la fruizione del libro elettronico, in sostanza l'hardware che supporta i lettori software. Si può trattare di un dispositivo dedicato, progettato e realizzato esclusivamente per la lettura di e-book[43], oppure di un qualsiasi altro apparato elettronico, quale un tablet-PC, un palmare, un notebook etc., grazie al quale accedere ai libri elettronici.
- **E-book reader**: il software per la lettura degli e-book. Al momento i software supportano soltanto uno specifico formato e-book; esempi di e-book readers sono Adobe Reader e Microsoft Reader.

[42] I dizionari da noi consultati (versione 2002) non presentano ancora il termine e-book, tuttavia sia il Devoto - Oli, sia il De Mauro accolgono il termine "e-mail", neologismo che analogamente ad e-book è formato dalla contrazione di "electronic" e "mail", con la "e" di "electronic" separata da un trattino; per questo motivo nel corso della trattazione abbiamo ritenuto preferibile adottare tale grafia.
[43] Questo è il caso in cui più corretto appare l'utilizzo del termine E-book reader device.

- **E-book format**: il formato elettronico, accessibile mediante un apposito software (e-book reader), nel quale viene salvato il contenuto degli e-book; tali formati hanno caratteristiche di visualizzazione, formattazione, navigazione e protezione[44] più avanzate rispetto ai più comuni formati digitali per il salvataggio dei testi *(txt, doc, rtf, etc.)*.

Una delle migliori definizioni di e-book in Italia è stata elaborata da Gino Roncaglia e Fabio Ciotti, studiosi che tra i primi nel nostro paese si sono occupati in modo approfondito delle tematiche inerenti l'e-book. Secondo Ciotti "con il termine *e-book* (libro elettronico) si intende un'opera letteraria monografica pubblicata in *forma digitale* e consultabile mediante appositi *dispositivi informatici"*[45]*. Q*uesta definizione è stata riformulata più accuratamente da Gino Roncaglia in occasione di un suo intervento pubblicato nel Bollettino dell'*Associazione italiana biblioteche*[46]:"potremmo parlare di libro elettronico, o e-book, davanti a un testo elettronico ragionevolmente esteso, compiuto e unitario ("monografia"), opportunamente codificato ed eventualmente accompagnato da metainformazioni descrittive, accessibile attraverso un dispositivo hardware e un'interfaccia software". Come si evince dalle precedenti definizioni, col termine e-book si intende la versione elettronica di un libro e si fa riferimento ad un oggetto digitale caratterizzato da una emulazione avanzata delle modalità di fruizione dei libri cartacei. Non qualsiasi testo digitale può essere considerato e-book, ma soltanto un materiale che sia riconducibile all'idea condivisa di libro. Secondo la "Raccomandazione circa la normalizzazione internazionale delle statistiche relative all'edizione dei libri e alle pubblicazioni periodiche", che la conferenza generale dell'Unesco approvò il 19 novembre del 1964, un libro è "una pubblicazione stampata non periodica che consta come minimo di 49 pagine, senza contare

[44] A tali formati sono spesso associate delle tecnologie di Digital Rights Management per la protezione della proprietà intellettuale.
[45] MARCO CALVO, FABIO CIOTTI, GINO RONCAGLIA, MARCO A. ZELA, *Frontiere di rete*, Roma-Bari, Laterza 2000, pp. 105-106.
[46] GINO RONCAGLIA, *Libri elettronici: problemi e prospettive*, a cura di Anna Galluzzi, in *Associazione italiana biblioteche. BollettinoAIB*
<http://www.aib.it/aib/boll/2001/01-4-409.htm>
25 febbraio 2002, (23 ottobre 2003).

quelle di copertina". Si considera un opuscolo la "pubblicazione stampata non periodica che consta di non meno di cinque pagine e non più di 48, senza contare quelle di copertina". In ambiente digitale una distinzione di questo genere non è proponibile, in quanto potendo variare dinamicamente nel passaggio da un dispositivo ad un altro, il numero delle pagine non rappresenta un discrimine sufficiente*, pertanto parliamo di testo elettronico ragionevolmente esteso, compiuto e unitario ("monografia")*. In questa direzione si muove anche la definizione che offre di e-book l'enciclopedia Microsoft Encarta: "e-book testo strutturato e articolato, di estensione analoga a quella di un libro (superiore dunque all'articolo o al trafiletto, e ipoteticamente senza limiti quanto al contenuto), disponibile in formato digitale in Internet".

Caratteristiche distintive dell'e-book sono quindi:

- **Oggetto digitale**: l'e-book è un oggetto digitale che possiede tutte le caratteristiche degli oggetti digitali, che può essere utilizzato soltanto per mezzo di un dispositivo informatico.
- **Emulazione libro**: l'e-book, seppur ampliandola, cerca di riprodurre quanto più fedelmente possibile sui dispositivi informatici l'esperienza della lettura di un libro cartaceo.
- **Opera monografica**: "La determinazione di 'opera monografica' differenzia un e-book vero e proprio dalla versione elettronica di una pubblicazione periodica, per indicare la quale si tende ad adottare il termine *e-journal* (periodico elettronico)"[47]. L'estensione di un e-book deve essere per analogia riconducibile a quella di un libro su supporto cartaceo.

II.1.1. E-book e codifica elettronica

È opportuna una precisazione a proposito dei rapporti, che intercorrono fra codifica elettronica dei testi letterari e l'e-book. Sebbene l'e-book rappresenti un caso di testo codificato elettronicamente, e spesso la codifica di un testo rappresenti il

[47] Vedi nota n°45.

primo stadio per la realizzazione di un e-book, scopo dell'e-book è quello di produrre un oggetto digitale la cui esperienza di fruizione richiami da vicino quella di un libro cartaceo, mentre finalità della codifica elettronica dei testi letterari è la rappresentazione formale di un testo in un formato utilizzabile dall'elaboratore. La codifica può sì sfociare nella creazione di un e-book, ma anche nella produzione di un testo digitale le cui informazioni possano essere trattate con l'ausilio di dispositivi informatici per le finalità più varie.

II.1.2. E-text

Un termine, cui è talvolta associato quello di e-book e che in taluni casi è adoperato come suo sinonimo, è quello di e-text, testo elettronico. In un'accezione di tipo estensivo con e-text si indica qualsiasi testo fruibile per mezzo di strumenti informatici, ad esempio pagine web, e-mail, documenti di word processor[48]. Secondo un'accezione più ristretta, l'e-text è un testo codificato in formato ASCII a 7 bit. La differenza fra e-text ed e-book ritengo risieda nel fatto che l'e-text manca delle caratteristiche di emulazione del libro tradizionale dell'e-book. A conferma di ciò vi è il fatto che il Project Gutenberg[49], che raccoglie e mette a disposizione in rete testi in ASCII puro, definisce le opere ospitate dai propri server come e-text e fornisce nelle pagine del sito accurate istruzioni per convertirle in e-book. Sebbene si tratti di testi digitalizzati di opere letterarie complete che quindi hanno le caratteristiche di *testo elettronico ragionevolmente esteso, compiuto e unitario ("monografia")* gli e-text del Project Gutenberg mancano degli aspetti di emulazione del libro tradizionale che contraddistinguono gli e-book, da ciò nasce l'esigenza di una trasformazione degli e-text in e-book.

[48] Cfr. BRUCE STERLING, "E-text" *Wired*, 3.05 maggio 1995, in WILLIAM GIBSON, BRUCE STERLING, *Parco giochi con pena di morte,* ("Piccola Biblioteca Oscar Mondadori"), Milano, Arnoldo Mondadori S.p.A. 2001 pp. 317-332.
[49] <www.gutenberg.net>

II.2. STORIA DELL'E-BOOK

La storia dell'e-book ha origine intorno alla fine degli anni novanta, in seguito all'affermazione dei siti commerciali per la vendita di libri (cartacei) on-line, i quali iniziarono ad offrire ai propri clienti contemporaneamente alle librerie, oltre alla versione cartacea, anche una trasposizione digitalizzata dei libri in uscita.

II.2.1. Alan Kay

Secondo alcuni[50] l'idea del libro elettronico è nata insieme a quella del personal computer, partorita dalla fervida mente di Alan Kay. Nel 1968 Kay concepì l'idea di un dispositivo, da lui chiamato *Dynabook*, che avrebbe dovuto essere "un personal computer interattivo e portatile, accessibile come un libro"[51]. L'idea del Dynabook fu alla base del lavoro svolto dallo studioso presso lo Xerox Palo Alto Research Center (PARC). Qui videro la luce le *graphical user interfaces*; concetti come quello di "finestra", "icona", "doppio click" furono sviluppati presso il Parc anche grazie al decisivo contributo di Alan Kay, il quale era ispirato e guidato dalla sua originale visione di "libro dinamico".[52]

II.2.2. Project Gutenberg

Un importante passo avanti nella storia dell'e-book è rappresentato dall'interesse per i testi letterari in formato elettronico suscitato dal Project Gutenberg.

---

[50] Tale idea presente in BRUNELLA LONGO, *La nuova editoria*, Milano, Editrice Bibliografica, 2001 p. 106 e probabilmente ripresa da alcune discussioni presenti in forum americani dedicati all'e-book, è stata riproposta da molti e si può trovare in diversi articoli e pagine web; tra queste merita di esser ricordata <http://www.ebookit.org/cronologia.htm> che offre una cronistoria assai accurata del fenomeno e-book ed a cui siamo debitori per alcune delle informazioni presenti in questo paragrafo.

[51] "a portable interactive personal computer, as accessible as a book" in BRIAN RAMPERSAD, *Alan Kay's Dynabook*, in *The Design of Everyday Things. Computer Software: Mapping our minds to create a brain prosthesis*. <http://www.sheridanc.on.ca/~randy/design.dir/webctsoftware/alan_kay.htm> aprile 1997, (27 ottobre 2003).

[52] Alan Kay è stato uno dei pionieri dell'informatica; infatti oltre ad aver inventato le interfacce grafiche moderne e tra i primi a concepire i *laptop*, è stato col suo linguaggio *Smalltalk* uno dei padri della programmazione orientata agli oggetti. Prima di collaborare come ricercatore con la Xerox al PARC, Kay ha collaborato al gruppo di ricerca dell'ARPA presso l'Università dello Utah, lavorando sia a progetti di grafica tridimensionale, sia al progetto ARPANet, dal quale derivò Internet, contribuendo a creare le *ethernet* e il modello client-server. Lasciata la Xerox, Kay lavorò per tre anni alla Atari. Nel 1984 ha fatto parte della Apple ed attualmente collabora con la Walt Disney Imagineering.

Probabilmente, il primo ad utilizzare il computer non soltanto per l'esecuzione di complesse operazioni matematiche, ma anche per la diffusione di opere letterarie su supporto digitale fu Michael Hart[53]. Nel lontano 1971 (siamo agli albori dell'era informatica) egli, studente dell'Università dell'Illinois, ebbe in modo del tutto casuale la possibilità di accedere al mainframe Xerox Sigma V., un costosissimo computer del Materials Research Lab dell'Università. Hart ottenne un account con 100.000.000 di dollari in tempo computer[54]; egli decise che il modo migliore per sfruttare la costosa potenza del computer consisteva nell'archiviazione, nel recupero e nella ricerca delle informazioni contenute nel patrimonio librario mondiale, al fine di diffondere, sfruttando le possibilità offerte dalla tecnologia digitale, il patrimonio culturale dell'umanità al maggior numero di persone possibile. Animato da questi nobili propositi Hart cominciò a digitare manualmente sul suo terminale il testo della Dichiarazione di indipendenza degli Stati Uniti: prendeva così l'avvio il Project Gutenberg. Nel giro di pochi anni al capostipite Hart si aggiunse una folta serie di volontari "amanuensi elettronici", i quali contribuirono a creare il più noto e ricco archivio testuale presente su Internet. Il Progetto Gutenberg ospita opere ormai libere dai diritti d'autore codificate in formato "Plain Vanilla ASCII" (ASCII puro a sette bit), il minimo comune denominatore per quanto riguarda la trasmissione di dati nella stragrande maggioranza dei computer esistenti[55]. Gli oltre seimila testi ospitati dal progetto sono direttamente accessibili da chiunque tramite internet[56]; per la sua notorietà il progetto Gutenberg è replicato su moltissimi server FTP ed è anche distribuito su CDROM dalla Walnut Creek. Al progetto Gutenberg sono anche dedicati una mail list e un newsgroup (bit.listserv.gutenberg), grazie ai quali è possibile avere

[53] Per le informazioni di queste pagine si veda: PROJECT GUTENBERG, *What is PG? HISTORY AND PHILOSOPHY OF PROJECT GUTENBERG*, in *Project Gutenberg official website* <http://promo.net/pg/history.html>
13 Maggio 2002, (05 settembre 2003).

[54] All'epoca l'accesso agli scarsi computers esistenti era prerogativa di pochi, in taluni casi le capacità computazionali degli elaboratori erano, per così dire, affittate a tempo, a costi assai elevati, ad aziende ed istituzioni.

[55] Questo era particolarmente importante negli anni in cui il progetto iniziò a muovere i primi passi, all'epoca non esisteva una piattaforma hardware e software preponderante come oggi, bensì una serie di piattaforme differenti spesso totalmente incompatibili fra di loro.

[56] L'indirizzo del sito del progetto è: <www.gutenberg.net>.

informazioni sugli ultimi titoli inseriti nella biblioteca ed essere aggiornati sulle nuove iniziative.

II.2.3. The Random House Electronic Thesaurus

Dobbiamo a Dick Brass, oggi vicepresidente per lo sviluppo tecnologico di Microsoft, il primo dizionario elettronico, nonché il primo software di correzione ortografica. Nel 1981 per la Dictronics Publishing, Inc. egli contribuì alla realizzazione del Random House Electronic Thesaurus, che può a buon diritto esser considerata la prima enciclopedia elettronica disponibile sul mercato.

II.2.4. Franklin Electronic Publishers

Il primo dispositivo che per certi aspetti può esser ricondotto ad un e-book reader device, fu realizzato nel 1986 dalla Franklin Electronic Publishers, Inc., società con base a Burlington, New Jersey; si trattava di un'evoluta agenda elettronica, la quale aveva la possibilità di contenere un dizionario digitale. Successivamente la Franklin realizzò una serie di altri titoli, opere di consultazione per professionisti ed uomini d'affari, integrabili nell'agenda; si dava così agli utilizzatori dell'apparecchio la possibilità di avere sempre con sé in uno spazio assai ridotto manuali, dizionari, enciclopedie che altrimenti non sarebbero stati agevolmente trasportabili per via del loro ingombro.

II.2.5. Il Sony Data Discman

Nel luglio del 1990 la Sony introdusse sul mercato giapponese e nel 1991 in quello americano il *Sony Data Discman*, un innovativo lettore di cd audio e cd-rom da 8 cm dotato di uno schermo a cristalli liquidi da 3,4 pollici che associava alle capacità di lettura di compact disc audio e video anche quella di cd contenenti testi. In associazione al dispositivo venivano forniti tre titoli in cd-rom Compton's Concise Encyclopedia, Wellness Encyclopedia e Passport's World Travel Translator. I libri elettronici per il Sony Data Discman occorreva fossero preparati usando il software *Sony Electronic Book Authoring System* (SEBAS), un sistema di authoring

basato su SGML. I titoli che furono realizzati per il Data Discman furono soprattutto opere di consultazione, quali dizionari, guide cinematografiche e di viaggio, vocabolari; tra i pochi titoli di argomento letterario (forse l'unico) si ricorda "Library of the Future" un cd-rom realizzato dalla World Library, Inc. contenente nella sua prima edizione ben 450 opere complete di narrativa, filosofia, teologia, politica, teatro e scienza di circa 50 autori; la seconda edizione conteneva un numero di testi doppio. Caratteristica rilevante del titolo era la possibilità di effettuare ricerche all'interno dell'intero parco dei libri contenuti nel cd adoperando oltre agli operatori boleani anche filtri per luogo (nazione, regione, continente), tempo (secolo, anno, era) e categoria (disciplina, genere) dell'opera ricercata. Le schermate potevano essere salvate, le configurazioni archiviate, e le opere confrontate sinotticamente.

Tra il 1990 e il 1994 furono prodotti circa 300.000 titoli di libri per il formato Data Discman. Le unità vendute di questo apparecchio nel 1994 erano state complessivamente 800.000; nonostante il formato Sony fosse stato adottato anche da altri grandi produttori di elettronica di consumo (Panasonic, Sanyo, Sharp), non fu grande il successo di questo innovativo prodotto e la sua diffusione restò circoscritta al mercato giapponese.

II.2.6. Rocket e Softbook

Il primo lettore dedicato per e-book è stato il Rocket e-book prodotto dalla Nuvomedia, una società fondata nel 1998 con il supporto dei capitali di Bertelsmann Ventures e della Barnes & Noble. Il dispositivo, una tavoletta di non più di un chilo di peso dotata di uno schermo monocromatico a cristalli liquidi sensibile al tocco, è stato presentato nel corso della Fiera del Libro di Francoforte nel 1998. Nello stesso anno sono iniziate le vendite dei titoli per il Rocket e-book, le "Rocket Editions", nella libreria on-line di Barnes & Noble, numerosi altri editori hanno in seguito adottato il formato Rocket per le proprie edizioni elettroniche.

Il Rocket e-book ha di poco strappato la palma di primo e-book reader device portatile ad un altro dispositivo, il Softbook, prodotto dalla Softbook Press. Il Softbook presenta caratteristiche analoghe al lettore della Rocket fatta eccezione per la presenza di un modem interno che permetteva di scaricare direttamente da Internet il libri elettronici per l'apparecchio.

II.2.7. Il 2000 anno zero

L'introduzione sul mercato del Rocket e-book e del Softbook suscitò un grande interesse da parte di numerosi soggetti del mondo del libro tradizionale e degli ambienti informatici. La nuova realtà dell'e-book venuta alla ribalta sul finire degli anni novanta accese speranze ed aspettative sia negli ambienti tecnologici, sia in quelli editoriali e culturali per la nascita di un nuovo fiorente (si prevedeva) mercato e per l'affermazione di nuove modalità di scrittura e di fruizione dei testi. Attorno alla innovativa tecnologia del libro elettronico si registrò un fervore di iniziative e di progetti con la scesa in campo di protagonisti sia del mondo dell'informatica, sia di quello editoriale che temevano di rimanere tagliati fuori da un nuovo, potenzialmente assai consistente, business. Intorno all'e-book si focalizzò anche l'interesse dei media e del mondo accademico, si ebbe un fiorire di saggi, servizi giornalistici, dossier dedicati alla nuova nascente realtà, in alcuni casi si annunciava ottimisticamente il sorgere di una nuova era per la lettura ed il libro in genere. Il duemila può essere considerato l'anno zero per l'e-book e per certi versi l'inizio del, si spera temporaneo, declino.

Nel gennaio 2000 la Gemstar, famosa per l'introduzione del sistema *show view* per i videoregistratori, ha acquisito le società pioniere del mercato e-book la Nuvomedia e la Softbook Press acquisendo così anche la tecnologia dei loro rispettivi lettori, che frattanto avevano ispirato una serie di, più o meno validi, epigoni. La Gemstar ha poi stabilito un accordo con il colosso dell'elettronica di consumo Thomson-RCA per sviluppare le nuove versioni dei due prodotti, presentate già nel mese di agosto ed immesse sul mercato a metà ottobre.

II.2.8. Il caso King

Un evento che, come ha giustamente notato Fabio Ciotti, "per la sua carica simbolica ha avuto l'effetto di un macigno lanciato nello stagno del mercato editoriale"[57] è stato, il 14 Marzo 2000, il rilascio da parte di Stephen King, autore tra i più venduti al mondo, del romanzo breve *Riding the Bullett* (Cavalcando il proiettile). Il maestro dell'horror ha reso disponibile il romanzo, in realtà una racconto breve di appena sessantasei pagine, esclusivamente sul web mettendolo in vendita alla cifra di due dollari e mezzo ed affidandone la distribuzione ai principali venditori di libri on-line, tra cui Amazon, SimonSays e NetLibrary, oltre che ad alcuni produttori di dispositivi e titoli per il mercato dell'editoria elettronica (Glassbook Inc., SoftLock.com., netLibrary, Nuvomedia Inc.'s Rocket eBook, Peanutpress.com e SoftBook Press). Quello di *Riding the Bullett* ha rappresentato un vero e proprio caso editoriale; nelle prime quarantotto ore dal rilascio il libro è stato acquistato e scaricato da ben cinquecentomila persone, e gli stessi server delle librerie virtuali spesso non sono riusciti a reggere l'imponente mole di traffico.

L'esperimento di King poté esser considerato un esperimento riuscito oltre che un successo, tuttavia dopo appena due settimane il libro venne piratato: infatti, disponibile in vari formati e per diverse piattaforme, la protezione della versione PDF del libro, basata su di un sistema di criptazione a quaranta bit, venne rimossa da hacker sconosciuti. La Glassbook inc., all'epoca titolare della tecnologia di protezione e responsabile della pubblicazione del libro on-line per il formato PDF, si affrettò a modificare il sistema di protezione alzando il livello di criptazione del testo usando una chiave a 64 bit; tuttavia, il danno era fatto e il libro cominciò a circolare in rete ed essere duplicato illegalmente tra gli appassionati in via gratuita.

[57] MARCO CALVO, FABIO CIOTTI, GINO RONCAGLIA, MARCO A. ZELA, *Frontiere di rete*, Roma-Bari, Laterza 2000, p. 107.

Nonostante l'incidente Glassbook, l'esperienza fu considerata positiva; nel luglio di quello stesso anno King iniziò a pubblicare come e-book i capitoli di un nuovo romanzo, *The Plant*; le puntate del romanzo venivano pubblicate con cadenza mensile in blocchi di circa cinquemila parole e messe in vendita con la formula dello shareware[58] sul sito dello scrittore[59] al prezzo di un dollaro l'una per le prime tre parti, due dollari invece per le altre fino all'ottava; lo scrittore si impegnava a rendere gratuiti tutti i capitoli successivi all'ottavo, si prospettava così per l'opera completa una spesa complessiva di tredici dollari, pressappoco il prezzo di una edizione tascabile. Il patto con i lettori, si leggeva nelle pagine del sito, era: "If you pay, the story rolls. If you don't, the story folds" (se paghi la storia continuerà altrimenti si interromperà). E la storia venne interrotta. Con la sesta parte pubblicata a dicembre la prosecuzione del racconto è stata sospesa, messa "in ibernazione" per usare le parole dell'autore. In termini di download l'avvio per The Plant era stato esaltante: centocinquantaduemila solamente nel corso della prima settimana, tuttavia erano soltanto centoventimila i lettori paganti che avevano scaricato il primo capitolo, solo quarantamila il quinto[60]. Come afferma l'assistente dello scrittore, Marsha DeFilippo: "In ottobre solo il 46 per cento di chi ha scaricato il quarto episodio ha effettivamente sborsato il prezzo richiesto"[61]. In termini puramente commerciali l'esperimento poté considerarsi fallito (secondo i fortunati standard di Stephen King), il profitto netto per l'autore è stato di 463.832 dollari (circa 1 miliardo di lire). Infatti, il profitto lordo dovuto alle vendite è stato di 721.000 dollari, ai quali vanno, però, detratti i 14.000 dollari per realizzare le varie versioni dell'e-book, 141.150 dollari spesi in pubblicità e 103.000 dollari per

[58] Lo shareware è una particolare forma di distribuzione del software che si basa sul principio "prova prima di comprare". I programmatori, soprattutto quelli ancora poco conosciuti, danno talvolta la possibilità all'utente finale di provare i propri prodotti per un tempo limitato (in genere 30 giorni) o per un numero limitato di utilizzi al fine di valutarne la qualità, con l'impegno, spesso soltanto morale, che l'utente paghi quanto dovuto al programmatore nel caso in cui decida di utilizzare il software.
[59] <http://www.stephenking.com/>.
[60] RICCARDO STAGLIANO, *King: "Perché ho smesso di pubblicare online"*, in *Repubblica.it* <http://www.repubblica.it/online/tecnologie_internet/king/spiega/spiega.html> 13 dicembre 2000, (29 ottobre 2003).
[61] ANNALISA CUZZOCREA, *Stephen King ferma il suo e-book*, in *Repubblica.it* <http://www.repubblica.it/online/tecnologie_internet/king/plant/plant.html> 29 novembre 2000, (29 ottobre 2003).

il sito web[62]. Agli inizi di gennaio del 2001 lo scrittore ha affidato alla propria casa editrice, la Philtrum Press, la vendita elettronica vera e propria del racconto, in tutte le sue sei parti, per il prezzo complessivo di sette dollari, e come ha detto l'autore: "Per questo, cari amici avrete bisogno della vostra carta di credito... Mia mamma non ha allevato uno stupido."[63]

II.2.9. La scesa in campo dei giganti dell'editoria

Nel corso del duemila sulla scorta dell'euforia generale per il fenomeno e-book anche alcuni giganti dell'editoria e della distribuzione hanno deciso di entrare nel mercato del libro elettronico. Time-Warner e Random House hanno fondato *iPublish* e *AtRandom* con l'obiettivo di produrre, pubblicare e distribuire titoli in formato elettronico. In seguito anche Simon & Schuster e McGraw-Hill hanno varato iniziative analoghe. Barnes & Noble, uno dei protagonisti della distribuzione editoriale negli Stati Uniti, dopo un accordo con Microsoft, si è lanciata nel mercato e-book con il suo sito (www.barnesandnoble.com), ed ha acquisito *iUniverse.com*, un sito dedicato alla pubblicazione di inediti in formato elettronico; anche Amazon a partire dal duemilauno, grazie ad un accordo con Microsoft, ha iniziato la distribuzione di e-book. Pure in Italia tra il duemila ed il duemilauno si è assistito alla scesa in campo di importanti soggetti quali Mondadori, Rizzoli, Apogeo-Longanesi, Il Sole 24 ore, Fazi Editore e Laterza che hanno avviato iniziative per la distribuzione e commercializzazione di libri elettronici.

II.2.10. Le grandi software house

Non sono rimaste a guardare le grandi software house, multinazionali del calibro di Microsoft e Adobe, che intravedendo la possibilità di congrui guadagni hanno deciso di lanciarsi nell'agone dell'e-book con delle proprie piattaforme tecnologiche.

---

[62] EVOLUTIONBOOK, *La Pianta ha dato ottimi frutti*, in *Evolutionbook* <http://www.evolutionbook.com/Mod_02.php?data_dir=News&data_file=Articolo&data=2001-02-19&numero=04>
19 febbraio 2001, (29 ottobre 2003).

[63] Vedi nota n°60.

Nel corso del duemila Microsoft ha presentato il suo formato per e-book ed il rispettivo software di lettura per le piattaforme Pocket Pc e PC/Windows. La Adobe , già titolare di una tecnologia standard di mercato per la distribuzione di documenti elettronici il *Portable Document Format* (PDF), ha acquistato la Glassbook, società produttrice di una completa piattaforma per la lettura e la protezione di e-book basata appunto su PDF, acquisendone così le tecnologie.

II.2.11. Il declino

Nel corso del duemila vide addirittura la luce un premio letterario internazionale riservato agli e-book, il Frankfurt eBook Awards, organizzato dalla International eBook Award Foundation nell'ambito della Fiera del Libro di Francoforte, a testimonianza del grande entusiasmo che in tutti i settori vi era per l'e-book; tuttavia, già sul finire del duemila e soprattutto nel corso del duemilauno proprio l'entusiasmo fu irrimediabilmente destinato ad affievolirsi. Il mercato non si rivelò maturo per il nuovo oggetto digitale, l'interesse del pubblico non fu ai livelli che entusiasticamente si prevedeva, l'evidenziata debolezza delle soluzioni di protezione del diritto d'autore, l'imperversare della cosiddetta "guerra degli mp3" ed il caso Napster smorzarono decisamente gli iniziali entusiasmi per l'e-book; fallimenti e ripiegamenti su posizioni più prudenti delle iniziative avviate sancirono una vera e propria crisi per un mercato che forse in realtà non si era mai realmente avviato. Già nel novembre duemilauno la Random House cancella il logo *AtRandom* che appariva sui libri stampati, originariamente pubblicati in versione elettronica ed attua un consistente ridimensionamento della sua divisione e-book. La segue a ruota in dicembre la Aol-Time Warnes che chiude la sua divisione *iPublish.com*. Nel duemiladue il Frakfurt eBook Award ha chiuso i battenti per il venir meno del sostegno degli sponsor e per la carenza di titoli da sottoporre al giudizio della giuria. Ha suscitato scalpore nel corso del duemilauno il cosiddetto caso Sklyarov. Dmitry Sklyarov è un giovane programmatore russo che per la Elcomsoft, la software house presso cui lavorava, aveva realizzato un programma,

l'"Advanced eBook Processor" (AEBPR), capace di bypassare le protezioni poste da Adobe sugli e-book realizzati con le proprie tecnologie, originariamente acquisite dalla Glassbook Inc.; AEBPR è perfettamente legale in Russia, dato che la legge locale dà al consumatore il diritto di creare una copia di backup dei prodotti software acquistati. Inoltre il programma funziona soltanto con e-book regolarmente acquistati tuttavia rimuovendo le protezioni rende possibile la copia e la distribuzione illegale degli e-book. Il 19 luglio 2001 a Las Vegas, dopo aver parlato al pubblico in una conferenza sulla sicurezza informatica in cui aveva per l'appunto presentato il software Advanced eBook Processor come rappresentante della Elcomsoft, Dmitry Sklyarov venne arrestato dalle autorità statunitensi proprio per via di quel prodotto, incriminato formalmente per cinque violazioni del copyright in base al Digital Millennium Copyright Act (una recente legge statunitense per la difesa della proprietà intellettuale che offre ampi poteri ai titolari dei diritti). Per il programmatore russo da quel momento iniziò una vera e propria odissea giudiziaria con l'incarcerazione ed in seguito il sequestro del passaporto e l'impossibilità di tornare in Russia. Un coro di voci si levò a favore del giovane contro la stessa Adobe, la quale dopo le numerosissime proteste per l'arresto di Sklyarov aveva ritirato le proprie denunce. Il clamore fu grande, la vicenda si è poi conclusa positivamente con il rilascio ed il rimpatrio del programmatore e l'assoluzione per la Elcolmsoft: tuttavia, ha portato alla ribalta la vulnerabilità dei sistemi di protezione dei contenuti digitali e fatto vacillare la sicurezza degli editori che iniziarono a temere che accadesse con i libri quanto già si verificava nel mondo della musica con gli mp3, una situazione di pirateria selvaggia.

Nel 2002 ed ancora nel 2003 gli annunci di chiusure, fallimenti per le società attive nel settore e-book si susseguono a catena. Nel luglio di quest'anno la Gemstar ha annunciato la chiusura della propria divisione e-book; inoltre, la società ha già interrotto la vendita dei propri e-book reader device ed ha anche cessato la vendita di contenuti nel proprio formato proprietario, una vera e propria ritirata

dal settore. In Ottobre anche Barnes & Noble ha annunciato la decisione di uscire dal mercato della vendita di e-book ed ha iniziato a rimuovere tutti i titoli e-book dai propri cataloghi. A motivare queste decisioni vi sarebbe il volume di vendite non corrispondente alle aspettative e la mancanza di una piattaforma per la lettura dei libri elettronici effettivamente diffusa sul mercato. Tutti questi eventi sembrerebbero segnare la fine del "sogno" e-book; d'altra parte, seppur ancora pallidi, all'orizzonte si delineano numerosi segnali che lasciano ben sperare per il sorgere di una "realtà" del libro elettronico che, non necessariamente legata a doppio filo con le esigenze del mercato, possa essere parallela e complementare al libro tradizionale. Ad esempio i dati di vendita di e-book negli Stati Uniti forniti dall'Open eBook Forum evidenziano una crescita del mercato:

dati forniti dai Retailers:

- nella prima metà del 2003 sono stati venduti 660.991 e-book, con un aumento del 40% rispetto all'analogo periodo 2002 (nel quale furono venduti 471.995 e-book);
- il numero dei titoli disponibili è 280.590, con un incremento del 144% rispetto ai 114.736 precedenti;
- il fatturato complessivo derivato dalla vendita di e-book è pari a quasi 5 milioni di dollari, con un aumento del 30% rispetto alle vendite della prima metà del 2002;

dati forniti dagli Editori:

- nella prima metà del 2003 sono stati venduti 620.277 e-book, con un incremento del 60% rispetto all'analogo periodo del 2002 (con vendite pari a 388.589 e-book);
- il numero dei titoli pubblicati è pari a 3.614, con un aumento del 45% rispetto allo stesso periodo del 2002 (2.485 titoli pubblicati);
- il fatturato complessivo derivato dalla vendita di e-book è pari a poco più di 3,5 milioni di dollari, con un aumento del 29% (2,8 milioni di dollari il precedente).[64]

[64] EVOLUTIONBOOK, *Aumentano ancora le vendite di eBook*, in *Evolutionbook*

Sebbene dai dati emerga ancora la scarsa consistenza del settore traspare chiaramente una tendenza di mercato positiva che autorizza a sperare in una futura crescita del comparto.

Numerose, inoltre, sono le iniziative che in ambito accademico e culturale si riallacciano alle tecnologia del libro elettronico: biblioteche digitali, conferenze virtuali, ristampa digitale di testi out of print, tecnologie di print on demand, prestampa di testi scientifici etc. e quindi verosimilmente la sfida del libro elettronico non vedrà esclusivamente nel mercato il suo principale campo di battaglia.

<http://www.evolutionbook.com/Mod_02.php?data_dir=News&data_file=Articolo&data=2003-09-24&numero=02>
24 settembre 2003, (29 ottobre 2003).

II.3. FORMATI E-BOOK

Sono tre i principali formati, utilizzati per la memorizzazione digitale dei libri elettronici: l'Adobe PDF, Il Microsoft Lit, e L'OEBPS. I primi sono soluzioni proprietarie di due tra le più importanti software house mondiali, l'ultimo è uno standard aperto promosso da un consorzio internazionale di soggetti attivi nel campo dell'editoria elettronica. Ognuno di questi formati ha proprie caratteristiche e peculiarità, limiti e vantaggi. Iniziamo la nostra trattazione con il più vecchio tra questi, l'Adobe PDF.

II.3.1. PDF

È opportuno, affrontando la trattazione del formato Adobe PDF, per meglio comprendere quanto verrà illustrato nel seguito del paragrafo, specificare brevemente la differenza che in ambito informatico intercorre fra grafica vettoriale e grafica bitmap.

Esistono due tipi di computer grafica: grafica a punti (bitmap graphics) e grafica vettoriale.

Nella **grafica bitmap** l'immagine è una griglia rettangolare (raster) di pixel[65] colorati. Tale griglia può essere paragonata ad un mosaico, le cui tessere sono costituite da pixel, quanto più piccole sono queste tessere tanto maggiore sarà la definizione dell'immagine. La grafica a punti è utilizzata soprattutto nella riproduzione di immagini di qualità fotografica. La grafica bitmap si presta meglio alla visualizzazione su video in quanto gli stessi monitor e televisori sono formati da una griglia di pixel. Tuttavia, in questo tipo di grafica l'immagine può essere ingrandita (a video o in stampa) soltanto ingrandendo la dimensione del pixel, che

[65] In informatica, abbreviazione per Picture Element (unità elementare di immagine): ciascuno degli innumerevoli punti che, disposti ordinatamente in righe e colonne, compongono le immagini così come vengono visualizzate da un computer o riprodotte da una stampante. Come il bit è la più piccola quantità d'informazione che un computer può elaborare, così il pixel è il più piccolo elemento manipolabile dall'hardware e dal software per la visualizzazione e la stampa di lettere dell'alfabeto, cifre numeriche o immagini. (fonte: Microsoft Encarta)

può diventare visibile, fino a mostrare la tipica struttura a mosaico (pixelizzazione).

La **grafica vettoriale** crea le immagini, descrivendole per mezzo di linee e curve. In pratica i dati dell'immagine vengono tradotti in equazioni matematiche che contengono tutte le istruzioni necessarie per tracciarla: ad esempio, per un segmento vengono solo memorizzate le coordinate del punto iniziale e di quello finale, per un cerchio solo le coordinate del centro e la lunghezza del raggio, mentre la colorazione avviene attraverso la colorazione delle linee e delle aree chiuse. La grafica vettoriale per essere visualizzata deve passare necessariamente attraverso un processo di rasterizzazione, ossia deve essere trasformata in un'immagine bitmap. Consistendo sostanzialmente in una descrizione matematica dell'immagine sono possibili nella grafica vettoriale, appunto attraverso semplici operazioni matematiche (eseguite dai programmi), operazioni di ridimensionamento dell'immagine senza alcuna perdita di qualità.

Uno dei primi formati multipiattaforma per la distribuzione di documenti elettronici, in seguito adottato anche per la diffusione di e-book, è stato l'Adobe PDF.

PDF (portable document format) è uno dei prodotti di punta della Adobe[66], software house americana fondata nel 1982 da John Warnock e Charles Geschke, già ricercatori dei laboratori Xerox PARC. Introdotto nel giugno 1993, il PDF è nato da una trasformazione di Adobe Postscript. Sviluppato da Warnock e Geschke il Postscript, primo ed a lungo unico prodotto della software house californiana, è un linguaggio di descrizione della pagina, ma anche un linguaggio di programmazione procedurale con comandi che indicano ad un dispositivo di output come rappresentare accuratamente la pagina descritta. Il Postscript si basa sui principi della grafica vettoriale per la descrizione della pagina. Postscript è nato per superare le incompatibilità e favorire l'interoperabilità tra sistemi di

[66] Con sede a San José, California, Adobe Systems è la seconda società americana di software per PC con un fatturato di oltre 1.2 miliardi di dollari.

fotocomposizione, problema rilevante sino all'introduzione del linguaggio negli anni ottanta.

La caratteristica più importante del linguaggio è la sua indipendenza dai vari dispositivi di uscita; un file generato con questa tecnologia può essere stampato correttamente da periferiche di stampa di ogni tipo, con l'unico vincolo che i plotter e gli altri dispositivi devono supportare il linguaggio.

A partire da Postscript fu creato il PDF con l'obiettivo di risolvere il problema di poter condividere e distribuire documenti con testi e grafica formattati su piattaforme differenti, mantenendo il layout originale. I documenti PDF possono essere visualizzati su qualsiasi computer o inviati a qualsiasi dispositivo di stampa, indipendentemente dalla piattaforma hardware e software, in cui il documento originale è stato creato. Per visualizzare i file PDF occorre Acrobat Reader, oggi ribattezzato Adobe Reader, un'applicazione gratuita liberamente distribuibile e disponibile per praticamente tutte le piattaforme informatiche esistenti: si calcola che sino ad oggi siano state distribuite più di cinquecentomilioni di copie di tale software. Per la creazione di file PDF si utilizza il software Acrobat prodotto dalla Adobe; da qualche anno, tuttavia, sono comparse sul mercato soluzioni alternative (sebbene non complete come Acrobat), alcune delle quali gratuite, per la creazione di documenti PDF. I file PDF vengono creati per mezzo di una stampante virtuale chiamata Distiller: in pratica il file viene creato in una qualsiasi applicazione, word processor o altro, e poi inviato alla stampante virtuale che lo "traduce" in PDF. Nel corso degli anni PDF è diventato uno standard *de facto* per la distribuzione di documenti elettronici, nel maggio del duemila è stato ufficialmente approvato dall'ANSI come standard ufficiale. Originariamente il PDF è nato come formato per l'interscambio di documenti; fu la Glassbook Inc. sul finire del novantanove ad immettere sul mercato una soluzione basata su PDF per la distribuzione di e-book. Particolare successo ebbe il Glassbook Reader, un software per la lettura di e-book PDF: la finestra del programma si divideva in un'area principale, alla cui destra era affiancata una barra di comandi. L'area principale consentiva di accedere

all'ambiente biblioteca, in cui erano elencati gli e-book disponibili sotto forma di miniatura dell'immagine della copertina. Era possibile selezionare i titoli visibili in base al soggetto, ordinarli secondo vari criteri e accedere a una finestra di informazioni mediante il tasto destro del mouse.

I pulsanti in alto a destra consentivano di scorrere le pagine avanti e indietro, di ruotare l'orientamento della finestra di 90 gradi, di aumentare o diminuire la dimensione della pagina; inoltre, il lettore poteva essere utilizzato anche affiancando due pagine.

Nell'estate del duemila, anche perché spinta dalla scesa in campo nel settore e-book della Microsoft, Adobe acquisì Glassbook, impadronendosi così delle sue tecnologie che vennero presto integrate nei suoi prodotti. Il Glassbook Reader divenne l'Acrobat Ebook Reader, nel corso del tempo il programma è stato arricchito di funzionalità e perfezionato. Sulla scorta di quanto fatto da Microsoft con il proprio lettore di e-book, anche Adobe integrò nel proprio prodotto una tecnologia di *sub-pixel font rendering*[67] per migliorare la leggibilità del testo denominata CoolType. Con l'acquisizione di Glassbook Adobe affiancò alla propria tecnologia per la tutela della proprietà intellettuale, PDF Marchant, le soluzioni di DRM per la gestione sicura della vendita di e-book nel formato Adobe PDF della società acquisita, tale piattaforma si chiama Content Server. Adottata da più di trecento siti di commercio elettronico la piattaforma Adobe, basata sui principi della moderna crittografia asimmetrica, fornisce una soluzione completa a tutti i soggetti della filiera del libro elettronico, dall'autore all'editore, dal distributore al rivenditore assegnandogli un ruolo preciso nella rete di distribuzione dell'e-book stesso. Una delle caratteristiche di spicco dell'Adobe Content Server è quella di consentire il prestito dei libri elettronici: fattore al momento non praticabile con altri sistemi software. La versione 2.2 di Adobe Ebook Reader è stata l'ultima; infatti, dall'estate di quest'anno (2003) le tecnologie di questo software sono state integrate con quelle dell'Acrobat Reader che ha così assunto il nome di Adobe

[67] Questo argomento verrà trattato dettagliatamente in seguito, nel paragrafo II.3.3.

Reader divenendo un lettore universale per tutto quanto si basi su PDF: documenti, e-book, album fotografici elettronici prodotti con Photoshop Album etc..

Adobe Reader abbandona l'interfaccia funzionale dell'Ebook Reader a favore di quella più spartana del vecchio Acrobat Reader.

Essendo PDF un formato per la descrizione fisica della pagina, e quindi più incentrato sull'aspetto grafico di questa piuttosto che sul suo contenuto testuale, presenta alcuni limiti per l'uso come formato e-book. Un libro elettronico in formato PDF deve essere creato, avendo in mente il dispositivo su cui questo verrà visualizzato; infatti, essendo praticamente un file vettoriale, il documento PDF potrà soltanto essere ingrandito o rimpicciolito per adeguarsi al dispositivo altrimenti, ad esempio, su schermi piccoli come quelli dei Personal Digital Assistant (PDA), si dovrà ricorrere ad un fastidioso scrolling verticale ed eventualmente anche orizzontale; invece, gli e-book basati sulla marcatura logica e strutturata del testo riescono ad adattarsi alla piattaforma con cui vengono aperti, creando dinamicamente le pagine e adattando l'impaginazione al dispositivo sia questo il monitor di un computer oppure il piccolo schermo di un PDA. Per cercare di ovviare a questo inconveniente a partire dalla versione cinque di Acrobat i file PDF possono contenere tag, che consentono di riformattare il flusso di testo al fine di adattarsi a dispositivi diversi. Tuttavia, essendo il PDF rivolto principalmente alla rappresentazione finale della pagina ed alla sua riproduzione su video o a stampa, difficilmente può possedere la ricchezza di metainformazione e la conseguente versatilità ed adattabilità alle esigenze più diverse di un documento basato su di un linguaggio di markup.

II.3.2. OEBPS

Con l'obiettivo di sviluppare e promuovere uno standard aperto, non proprietario, leggibile da tutti i sistemi ed atto a favorire la diffusione dell'editoria elettronica, è

stato creato nell'ottobre del 1998 l'Open e-book Forum[68] (OEBF), una organizzazione privata internazionale composta da produttori di software e hardware, editori, autori, lettori ed altre associazioni e istituzioni attive nel campo dell'editoria elettronica e non. Fra i membri dell'Open e-book Forum vi sono aziende dell'Information Tecnology, quali Microsoft, Adobe, Overdrive, Palm Digital Media Group, Hewlett-Packard Labs, numerose case editrici come HarperCollins, McGraw-Hill, Random House, vari enti e associazioni tra cui spiccano il National Institute of Standards and Technology (NIST), la Library of Congress, l'American Library Association, l'Association of American Publishers. Sino a poco tempo orsono sono stati membri dell'OEBF le italiane Mondadori, IPM, azienda attiva nel mercato delle telecomunicazioni e dei sistemi automatici di pagamento conosciuta in tutto il mondo per la produzione di telefoni pubblici (tra cui anche il nuovo "Digito" della Telecom presente nelle nostre strade) e produttrice del "My Friend" un versatile e-book reader device, e Somedia società del Gruppo Editoriale l'Espresso impegnata nel settore della Formazione on-line e dell'Editoria Multimediale.

Uno standard formale per gli e-book, chiamato in origine OEB, ma a partire dalla versione 1.2 delle specifiche, OEBPS, è stato elaborato dall'Open e-Book Forum già nel settembre del 1999 con il rilascio della prima versione delle specifiche OEBPS 1.0 (Open eBook Publication Structure specification), il documento in cui vengono descritte le caratteristiche di questo linguaggio per l'editoria digitale. A questa prima versione ne è seguita una seconda riveduta e corretta, la 1.01, nel luglio 2001. Nell'agosto del 2002 è stata rilasciata con la 1.2 la terza e per il momento ultima versione delle specifiche, la cui evoluzione comunque continua.

Il formato OEBPS si basa su alcune tecnologie già consolidate in ambiente internet: XML, HTML, CSS.

Una pubblicazione OEBPS è formata da un insieme di file HTML, i quali comprendono i contenuti della pubblicazione elettronica. Tuttavia, nella

[68] <www.openebook.org>.

pubblicazione OEBPS si ricorre soltanto ad un sottoinsieme del linguaggio HTML 4.0 e CSS2; infatti, non tutti i comandi di questi due linguaggi sono supportati dal formato OEBPS. A partire dalla versione 1.2 delle specifiche il dizionario di markup è divenuto un puro sottoinsieme dello standard XHTML 1.1 (l'evoluzione-conversione in XML del vecchio HTML).

L'insieme di file costituenti la pubblicazione OEBPS, conosciuti anche come pacchetto OEBPS, sono collegati fra di loro ed organizzati per mezzo di un file XML detto *package file* e caratterizzato generalmente dall'estensione ".opf". Le specifiche definiscono la sintassi del vocabolario XML con cui realizzare il *package file.* Il *package file* costituisce la radice dell'albero dei documenti della pubblicazione ed è sostanzialmente un file XML contraddistinto da sei sezioni fondamentali:

- PACKAGE IDENTITY: un identificatore univoco della pubblicazione. Questo elemento è posto immediatamente sotto la dichiarazione di tipo di documento XML.
- METADATA: questo elemento contiene una serie di metainformazioni relative alla pubblicazione. I metadata hanno la funzione di definire delle informazioni utili al reperimento dell'informazione potenzialmente contenuta all'interno della pubblicazione ed alla classificazione e catalogazione della stessa. Tra questi vi sono il titolo dell'opera, il nome dell'autore del testo, dei curatori dell'edizione elettronica, dei traduttori, o di altri collaboratori, una descrizione del contenuto dell'opera, la data di pubblicazione, il genere, il formato, la lingua e le fonti della pubblicazione, e numerose altre informazioni. Per la definizione dei metadata la specifica OEBPS fa ricorso ad un sott'insieme del *Dublin Core metadata element set*[69], uno standard internazionale per i metadata.

[69] Il Progetto Dublin core metadata, è nato nel 1995 nel corso di un convegno promosso dall'Oclc (Online computer library center), una organizzazione senza scopo di lucro che si occupa di catalogazione e dell'allestimento e della commercializzazione di archivi elettronici, con sede a Dublin, in Ohio negli Stati Uniti. Il progetto Dublin Core ha l'obiettivo di favorire l'interoperabilità e migliorare il recupero delle risorse informative su Web da parte di biblioteche, editori, archivi, autori di

- MANIFEST: questo elemento contiene una lista dei file, che costituiscono la pubblicazione (documenti conformi alle specifiche, immagini[70], fogli di stile, etc.); può, inoltre, contenere le dichiarazioni di *fallback*, ossia informazioni necessarie per la visualizzazione di tipi di file non supportati dalle specifiche.
- SPINE: con questo elemento si stabilisce l'ordine di lettura di default dei file, che costituiscono la pubblicazione.
- TOURS: elemento facoltativo, con cui è possibile stabilire percorsi di lettura alternativi.
- GUIDE: ultimo elemento, anch'esso facoltativo, contenente riferimenti a file relativi ad alcuni componenti strutturali della pubblicazione: l'indice, la bibliografia, la copertina e quant'altro presente.

Ecco di seguito ad esempio come potrebbe presentarsi il codice di un documento ".opf":

```
<?xml version="1.0"?>
<!DOCTYPE package
  PUBLIC "+//ISBN 0-9673008-1-9//DTD OEB 1.0 Package//EN"
"http://openebook.org/dtds/oeb-1.0/oebpkg1.dtd">
<package unique-identifier="identificatore_univoco">
<metadata>
    <dc-metadata xmlns:dc="http://purl.org/dc/elements/1.0/"
xmlns:oebpackage="http://openebook.org/namespaces/oeb-package/1.0/">
 <dc:Identifier
id="identificatore_univoco">identificatore</dc:Identifier>
 <dc:Title>Titolo dell'opera</dc:Title>
 <dc:Creator file-as="Nome Cognome"
role="aut">nome autore</dc:Creator>
 <dc:Subject>argomento</dc:Subject>
```

informazione. Esso fornisce una struttura per esprimere metadati, che costituisce la base per un vocabolario comune indipendentemente dall'ambiente in cui si lavora.

[70] Al momento sono solo due i formati grafici supportati dalle specifiche: il JPG ed il PNG.

```
 <dc:Description>descrizione dell’opera</dc:Description>
 <dc:Contributor file-as="Nome Cognome" role="com">nome curatore
edizione elettronica</dc:Contributor>
 <dc:Type>genere dell’opera</dc:Type>
 <dc:Date>data</dc:Date>
 </dc-metadata>
    </metadata>

<manifest>
<item id="toc" href="indice.htm" media-type="text/html" />
<item id="main" href="contenuto.htm" media-type="text/html"/>
<item id="about" href="about.htm" media-type="text/html"/>
<item id="copyright" href="copyright.htm" media-type="text/x-oeb1-
document"/>
<item id="coverstandard" href="copertina.jpg" media-type="image/jpeg"
/>
</manifest>

    <spine>
        <itemref idref="toc" />
        <itemref idref="main" />
    </spine>
    <tours>
        <itemref idref="main" />
      <itemref idref="toc" />
    </tours>
    <guide>
<reference type="toc" title="Indice" href="indice.htm" />
<reference type="other.ms-about" title="About" href="about.htm" />
<reference type="copyright-page" title="Copyright"
href="copyright.htm" />
```

```
<reference type="other.ms-coverimage-standard" title="cover image
standard" href="copertina.jpg" />
   </guide>
</package>
```

Sebbene gli e-book in formato OEBPS possano essere letti attraverso lettori compatibili, quello creato dall'Open eBook Forum non è un formato pensato per l'utente finale ma piuttosto un formato "intermedio", uno standard aperto utile per creare una sorta di "sorgente" del libro elettronico che possa poi essere compilato in vari formati proprietari.

Le specifiche OEBPS al momento attuale non prevedono particolari indicazioni riguardo soluzioni per la protezione della proprietà intellettuale degli e-book OEBPS; comunque l'Open ebook Forum ha da qualche tempo iniziato a collaborare con l'Electronic Book Exchange (EBX), un'organizzazione nata per l'elaborazione collettiva di standard nel campo della protezione dei diritti d'autore nell'ambito dell'editoria elettronica con l'intenzione di contribuire alla formazione di una tecnologia di DRM standard e aperta.

II.3.3. Il formato LIT

Come detto, OEBPS è uno standard aperto che rappresenta soprattutto un formato intermedio per la realizzazione di libri elettronici in formati proprietari integranti sistemi di protezione del contenuto, di cui OEBPS è privo. Proprio sullo standard OEBPS si basa la soluzione Microsoft per il libro elettronico, il formato LIT.

Viene comunemente indicato come LIT il formato Microsoft per i libri elettronici, in quanto ".lit"[71] è l'estensione dei file e-book fruibili grazie al software di lettura degli e-book Microsoft, il Microsoft Reader. Rilasciato in versione per Pocket PC nell'aprile del 2000 e nella versione 1.0 per la piattaforma PC all'inizio di agosto 2000 e giunto oggi alla versione 2.0, il Microsoft Reader è, per usare le stesse

[71] Secondo alcuni il nome di questa estensione deriverebbe da literature

parole con cui lo descrive la software house di Redmond : "un'applicazione software gratuita, sviluppata per consentire un'esperienza di lettura su schermo che si avvicina alla praticità e alla qualità del testo su supporto cartaceo. Microsoft Reader è il primo prodotto in cui sia stata inclusa la tecnologia di visualizzazione brevettata ClearType, che migliora fino al 300% la risoluzione su schermi LCD per fornire un tipo di visualizzazione simile alla stampa."[72]

Microsoft ClearType è una sofisticata tecnica di anti-aliasing, atta a migliorare la leggibilità del testo a video. Come si è già rilevato gli schermi dei computer sono formati da una griglia di pixel disposti su righe e colonne. A causa di questa struttura a "mosaico" dei dispositivi di visualizzazione, quando la risoluzione, ossia il numero di pixel per pollice, del sistema di visualizzazione non è sufficiente, oppure una immagine viene ingrandita sino a render visibile i singoli punti che la costituiscono, si produce un effetto di distorsione grafica, detto Aliasing, caratterizzato dall'aspetto "a gradini" di alcuni elementi grafici quali linee curve o rette diagonali, quindi particolarmente evidente con i caratteri di testo. Per contrastare questo fastidioso effetto "mosaico" sono stati sviluppati degli algoritmi, detti di anti-aliasing o dithering, che sfumano la scalettatura di testo ed immagini. L'artificio, cui ricorrono questi programmi, consiste nell'aggiungere ai bordi dell'immagine dei pixel di minore intensità cromatica rispetto all'immagine stessa, producendo così l'impressione di un arrotondamento dei bordi. Questa tecnica si avvale della tendenza che ha l'occhio umano a mediare l'effetto di un insieme di punti vicini di diverso colore, confondendoli nella percezione di un'unica tinta. Le tecnologie di *sub-pixel font rendering,* quali Microsoft ClearType o Adobe Cooltype, finalizzate al miglioramento della leggibilità dei testi, rappresentano una evoluzione delle tecniche di anti-aliasing che vengono applicate non più al livello dei pixel, ma a quello dei subpixel. Alla base di questi algoritmi grafici vi è il fatto che ogni pixel di uno schermo è in realtà composto da tre subpixel uno rosso, uno

[72] MICROSOFT CORPORATION, *Microsoft Reader – Support for PC*, in *Microsoft.com* <http://www.microsoft.com/reader/it/support/faq/general.asp> (29 ottobre 2003).

verde ed uno blu; partendo, quindi, da questo presupposto si è riusciti a sviluppare delle più accurate tecniche di anti-aliasing che operando a livello di subpixel[73], più piccoli dei pixel, riescono perciò a garantire una migliore nitidezza complessiva nel disegno dei caratteri rispetto alle tradizionali tecniche di anti-aliasing. I benefici di questa tecnologia si riescono ad apprezzare soprattutto sugli schermi a cristalli liquidi (LCD)[74], dove i sub-pixel sono tre rettangoli adiacenti in senso orizzontale.

L'interfaccia del programma richiama da vicino l'aspetto di un libro stampato in formato tascabile, sul modello di quella del Glassbook Reader, poi Adobe Ebook Reader. Il software consente la lettura delle pagine senza la necessità di uno scrolling verticale e la finestra del programma può essere sovrapposta a quella di altri programmi in esecuzione. Una caratteristica pregevole del Microsoft Reader è il supporto per la gestione del *page flow* in modo dinamico. Essendo il LIT un formato basato su di un linguaggio di marcatura logica del testo, il Reader è stato dotato di un motore di lettura dinamico, in grado di creare l'e-book pagina per pagina sulla base delle caratteristiche del dispositivo di visualizzazione. Questa gestione dinamica del layout comporta la capacità di adattare automaticamente la quantità di testo contenuta nelle pagine e la loro dimensione (pur mantenendo un rapporto fisso di 4/3 tra altezza e larghezza) in base alla risoluzione e dimensione dello schermo. In questo modo uno stesso e-book può essere letto indifferentemente su un computer da tavolo, su un portatile o su un palmare senza che si debba scorrere in verticale la pagina. Inoltre, le dimensioni del testo possono essere aumentate o diminuite in base alle esigenze dell'utente.

Il Microsoft Reader prevede la possibilità di poter cercare delle parole all'interno del testo, di evidenziare frasi e di aggiungere delle note. Sin dal momento del lancio del suo software di lettura nel 2000 la Microsoft ha stretto accordi con

[73] Le dimensioni di un subpixel sono un terzo di quelle di pixel.
[74] Negli schermi a tubo catodico (CRT) (quelli di monitor e televisori) i pixel sono punti di fosforo circolari che si sovrappongono, e non aree nettamente distinte come nei display LCD. Dobbiamo tuttavia dire, per esperienza personale, che anche su questi dispositivi si riscontra un, seppur minimo, miglioramento della leggibilità dei caratteri in seguito all'adozione di questa tecnologia.

numerosi importanti editori: in Italia Mondadori è stata la prima casa editrice a vendere e-book per il Microsoft Reader. Sul fronte dei sistemi per la protezione dei diritti di copyright il formato di file di Microsoft Reader impedisce la modifica non autorizzata del contenuto di un e-book; il lettore, inoltre, non prevede alcuna opzione di stampa diretta del testo dei libri elettronici.

La piattaforma DRM per la gestione sicura della vendita di e-book nel formato Microsoft LIT si chiama D.A.S. (Digital Asset Server), ed analogamente ad altre, come ad esempio quella Adobe, si basa su sofisticate tecniche crittografiche. Il sistema D.A.S. opera principalmente su tre livelli di protezione:

Sealed (sigillato): è il livello di protezione base applicato al momento della conversione del file, il quale viene criptato in modo tale da assicurarne l'integrità del contenuto. Di un e-book "sigillato", in pratica qualsiasi file LIT, non può essere modificato il testo né qualsiasi altro contenuto. A questo livello di protezione gli e-book possono essere letti da qualsiasi copia del Microsoft Reader anche se quest'ultimo non è attivato; è possibile, inoltre, copiare tranquillamente il file del libro elettronico.

Inscribed (iscritto): oltre ad essere "sigillati", questi e-book vengono "iscritti" con delle informazioni sull'acquirente, nome, cognome ed altri eventuali dati sensibili, che compaiono sulla pagina di copertina; questi e-book possono essere letti da qualunque Reader, ma garantiscono la possibilità di risalire all'acquirente del prodotto.

Owner Exclusive (proprietà esclusiva): il terzo livello garantisce la protezione massima. Oltre ad essere iscritto il file è ulteriormente criptato per mezzo di un sistema asimmetrico di chiavi, in modo tale che il libro sia associato in modo univoco soltanto ad una determinata copia del Microsoft Reader, che, per accettare libri a questo livello di protezione, deve essere necessariamente attivato. L'attivazione è una procedura mediante la quale viene creata una particolare chiave di decrittazione collegata con l'identità dell'utente e la configurazione hardware del dispositivo di lettura; questa chiave è utilizzata per la decodifica dei

titoli protetti con il sistema Owner Exclusive e serve ad impedire la distribuzione illecita di copie di e-book acquisiti. In pratica, il libro elettronico potrà essere letto soltanto su quei dispositivi e con quelle copie del Reader, in cui è presente la chiave di decrittazione, al momento è possibile l'attivazione di soltanto un certo numero di copie del Microsoft Reader, inoltre questo può essere attivato al massimo soltanto un certo numero di volte stabilito dal produttore del software[75]. Negli e-book Owner Exclusive l'opzione di copia del testo è disabilitata. Questa modalità di protezione dei testi rischia di limitare eccessivamente la libertà dell'acquirente, che non può prestare i propri e-book e che rischia di non poter più leggere in alcun modo i propri libri dopo aver cambiato il proprio hardware o anche soltanto formattato il proprio hard disk, cosa abbastanza comune in ambito informatico, per un numero di volte superiore alle attivazioni disponibili.

Esistono vari strumenti per la creazione di file LIT alcuni commerciali, altri gratuiti. Per tutte le versioni di Word a partire dalla 2000 Microsoft mette a disposizione un piccolo plug-in per il noto programma di videoscrittura, che permettere di salvare file di Word in formato LIT. Sul sito della Microsoft è anche presente un kit per sviluppatori, che desiderano integrare nelle proprie applicazioni la possibilità di salvare in formato LIT.

Il panorama software per questo formato va crescendo di giorno in giorno, oltre alle applicazioni complete, come il software di authoring ReaderWorks della Overdrive, società che fornisce anche servizi nella gestione del sistema Digital Asset Server della Microsoft, particolarmente interessanti sono i compilatori per pacchetti OEBPS che partendo da pubblicazioni conformi allo standard sviluppato dall'Open eBook Forum producono e-book in formato LIT.

L'approccio Microsoft all'e-book è per certi aspetti molto diverso da quello Adobe, ambedue le soluzioni presentano pregi e difetti. Citando fonti Microsoft: "Il formato PDF è molto utile nel caso in cui sia necessario acquisire documenti e

[75] Il numero di attivazioni possibili è variato nel corso del tempo, si è passati da un massimo di due attivazioni contemporanee e quattro totali delle prime versioni del software, al massimo di otto attivazioni (anche contemporanee) dell'ultima versione del Reader.

stampare immagini. Tuttavia, è un formato statico e non favorisce un buon adattamento della pagina, né si adatta alle dimensioni dello schermo in uso, presentando in questo modo alcune limitazioni". Le funzioni di page flow dinamico del programma Microsoft rendono il Reader probabilmente il sistema ideale per una fruizione a video dei libri elettronici; d'altra parte, il formato PDF ha decisamente una marcia in più in tutti quei casi in cui è importante mantenere l'informazione circa l'aspetto grafico della pagina. Il formato LIT, essendo ancora piuttosto giovane, non è in grado di gestire a livelli professionali l'inserimento di grafica e immagini, attività in cui invece eccelle il formato della Adobe e che è molto importante in ambiente scientifico; tra l'altro, il PDF è uno dei pochi formati che garantisce una corretta visualizzazione di simboli matematici prodotti con il tex[76]. I file LIT crittati possono essere letti con Personal Computer con sistema operativo Windows a partire dalla versione 98 e con Pocket PC (dotati di sistema operativo PocketPC2000 o successivo) con installato ed attivato Microsoft Reader; i file LIT non crittati possono essere letti con Personal Computer con sistema operativo Windows a partire dalla versione 98 e con qualsiasi versione dei Pocket PC; probabilmente il più grosso limite di questo formato è la chiusura verso importanti piattaforme informatiche e sistemi operativi quali Linux o Macintosh. Potremmo dire che i due formati concorrenti PDF e LIT sono per certi versi complementari con vantaggi e limiti, che possono emergere sulla base dell'uso cui il libro elettronico è destinato.

[76] Il TeX è un sistema di elaborazione testi realizzato nel 1977 da Donald E. Knuth, qualche tempo dopo è stato realizzato il LaTeX da Leslie Lamport, ed è quest'ultima variante che ha oggi maggior diffusione. Il principale utilizzo di TeX e LaTeX è l'elaborazione di testi contenenti soprattutto formule matematiche.

II.4. HARDWARE PER IL LIBRO ELETTRONICO

Tra le circostanze che nel duemila hanno prodotto il cosiddetto boom dell'e-book è da rilevare l'immissione sul mercato dei dispositivi dedicati per la lettura di libri elettronici della Nuvomedia e dalla Softbook. D'altra parte, allorché gli entusiasmi iniziali si sono stemperati in quello che è stato considerato un clamoroso flop dell'e-book, il mancato sorgere di un'età della lettura digitale è stato spesso imputato all'assenza ed alla scarsa diffusione degli e-book reader device. Il device di lettura sta al libro elettronico come la carta sta al libro tradizionale; si comprenderà bene quindi l'importanza rivestita dall'hardware per la lettura in numerosi aspetti della fruizione e diffusione dell'e-book.

Al momento, visto il sostanziale fallimento dei device dedicati (basti pensare al ritiro dal mercato della Gemstar, il principale produttore di questi dispositivi), il libro elettronico deve fare riferimento ai personal computer, ai palmari ed ai Tablet Pc.

Naturalmente i personal computer da tavolo mancano di quelle caratteristiche di ergonomia, ma soprattutto trasportabilità e maneggevolezza, che dovrebbero contraddistinguere un e-book reader device; migliore la situazione con i notebook, la versione portatile dei pc da tavolo, tuttavia non ancora in grado di garantire un'esperienza di lettura ottimale soprattutto a causa dello sviluppo in orizzontale dello schermo. I dispositivi che oggi vengono considerati più idonei alla fruizione di e-book, sebbene questo non sia il loro unico uso, sono i palmari ed i Tablet Pc; rispetta questo fatto la situazione attuale del mercato dell'editoria elettronica e dei software per la lettura, che proprio verso questi apparecchi vede concentrare sforzi ed iniziative.

II.4.1. Palmari

I palmari o palmtop sono dei piccoli computer, per certi versi evoluzione delle agende elettroniche assai diffuse verso la fine degli anni ottanta, che come dice il

nome "stanno nel palmo di una mano". Delle dimensioni circa di un bloc-notes, sono dotati di uno schermo LCD sensibile al tatto con cui si interagisce per mezzo di uno stilo, che rappresenta il principale dispositivo di input; esistono anche apparecchi dotati di tastiera che però prendono il nome di handled. Sviluppati originariamente per svolgere soprattutto funzioni di PIM (personal information manager) quali l'annotazione di appunti, la gestione di un'agenda etc., riescono oggi a svolgere buona parte delle attività del fratello maggiore, il personal computer. Sono due attualmente le piattaforme che si spartiscono il mercato: Palm OS e Microsoft Pocket PC.

II.4.2. Palm OS

Pioniera ed a lungo leader indiscussa nel settore dei palmari possiamo affermare che la Palm Inc., sin dall'introduzione nel 1996 dei suoi primi palmtop, ha avviato una nuova era nel campo del mobile computing e degli organizer elettronici. Il Palm OS è il fortunato sistema operativo per palmari prodotto dalla Palm Inc.[77], che rappresenta il cuore software dei dispositivi costruiti dalla stessa Palm e da altre società quali Handspring e Sony, che adottano su licenza questo sistema operativo per i propri prodotti. Una delle caratteristiche di maggior spicco del sistema operativo è il sistema di riconoscimento del testo scritto per l'immissione dei dati, Graffiti; il programma si basa su caratteri, per così dire, semplificati, in pratica l'utente deve imparare un nuovo alfabeto ma una volta presa confidenza con il sistema è in grado di immettere dati nel palmare con una discreta velocità e comodità. Ben presto dopo la commercializzazione dei primi palmari Palm, sulla scia del successo del prodotto, cominciarono ad esser offerti su internet testi elettronici da leggere con il palmare, sorsero pure alcune iniziative di editoria elettronica; particolare fortuna ebbe la Peanut Press, che è stato uno dei primi editori specializzati in pubblicazioni elettroniche per i Palm. Gli e-book Peanut

[77] Nell'ottobre del 2003 in seguito ad una ristrutturazione societaria ed all'incorporazione della concorrente Handspring, la società ha cambiato nome dando vita a due differenti società, PalmSource che si occupa di sistemi operativi e software, e PalmOne che è attiva nel settore hardware e si occupa della progettazione e costruzione di palmari e cellulari.

Press adottavano un formato proprietario ed erano leggibili per mezzo di un lettore software distribuito dalla stessa società.

Grazie al grande successo ottenuto nell'aprile del 2001 la società è stata acquisita[78] da parte di Palm Inc. In seguito all'acquisizione Peanut Reader e Peanutpress hanno preso il nome di Palm Reader e Palm Digital Media; da quel momento il Palm Reader viene fornito in dotazione con tutti i nuovi palmari messi in commercio.

Il Palm Reader adotta un sistema assai originale per la protezione del copyright dei propri titoli, anziché basarsi su chiavi di cifratura collegate all'hardware utilizzato, che comporta il rischio di non poter più leggere i propri libri dopo aver cambiato il proprio hardware, il Palm Reader utilizza come chiavi direttamente il nome e il numero di carta di credito dell'utente. Gli e-book possono essere dunque trasferiti e duplicati liberamente. La scarsa probabilità che l'acquirente del libro sia disposto a diffonderne copie, dato che facendolo distribuirebbe anche i propri dati personali e il numero della propria carta di credito, costituisce una valida garanzia contro le copie non autorizzate. Questo tipo di protezione del libro permette all'acquirente legittimo una grande libertà nell'uso dell'e-book acquistato, che può essere facilmente trasferito da un dispositivo all'altro. Il formato di codifica utilizzato dagli e-book per Palm Reader si basa al momento sul cosiddetto Palm Markup Language (PLM), un linguaggio sviluppato autonomamente dalla Palm e abbastanza diverso nella sintassi dai linguaggi di marcatura standard. Gli strumenti di sviluppo, realizzati in Java, sono disponibili per tutti i principali sistemi operativi.

Sono centinaia i titoli, che la libreria virtuale offre ai propri lettori: dalle guide di viaggio alle raccolte di giurisprudenza, dai dizionari ed enciclopedie ai grandi classici della letteratura anglo-americana, oltre ovviamente a numerose opere di narrativa contemporanea.

[78] Poco tempo prima Peanut Press (fondata nel 1998) era già stata acquisita da netLibrary, un importante venditore di e-book statunitense.

II.4.3. Pocket PC

Sull'onda del successo dei palmari con sistema operativo Palm OS, anche Microsoft ha deciso di lanciarsi in questo lucroso mercato con un proprio prodotto. Già nel 1996 la società di Redmond aveva iniziato a commercializzare un sistema operativo, chiamato allora Windows CE, per dispositivi portatili; tuttavia, il prodotto era orientato soprattutto verso la categoria degli handled pc, apparecchi con tastiera. Per anni il business dei nuovi PDA (personal digital assistant) è stato dominato dai prodotti con sistema operativo Palm OS, con quote di mercato intorno al settanta percento. Con l'obiettivo di insidiare la leadership Palm proprio nel settore dei palmari, nella primavera duemila Microsoft ha rilasciato la versione 3.0 del sistema operativo Windows CE, che da questa versione è stato chiamato Pocket PC alludendo non soltanto al lancio di un nuovo sistema operativo, ma all'avvio di una nuova piattaforma per il mobile *computing*. Il Pocket PC include versioni ridotte di alcuni dei principali programmi del sistema operativo Windows e della famosa suite Microsoft Office: Pocket Excel, Pocket Word, Microsoft Reader, Windows Media Player, Esplora File, Pocket Internet Explorer, Calendario, Contatti, Attività; si è cercato in pratica di creare con il nuovo prodotto una versione in piccolo "pocket" dei personal computer da tavolo, da qui quindi il nome Pocket PC.
Le principali armi che la società di Bill Gates ha sfoderato contro il proprio avversario si chiamavano Windows Media Player e Microsoft Reader, applicazioni che venivano distribuite di serie con il sistema operativo. Le offerte che la Microsoft presentava contro il prodotto concorrente erano una avanzata gestione dei contenuti multimediali audio e video grazie a Windows Media Player, ed una migliorata esperienza di lettura grazie al Microsoft Reader. Anche per il big di Redmond le funzioni di e-book reader device rivestono un ruolo di primo piano per questa categoria di prodotti. La versione attuale del sistema operativo è la Pocket PC 2003. Al momento le due software house rivali si marcano stretto ed ormai anche il Palm OS offre funzioni pressoché analoghe a quelle di Pocket Pc

II.4.4. Considerazioni sulla piattaforma palmare

La piattaforma palmare è da molti criticata come non idonea alla lettura; ad esempio, così si esprime Harold Bloom, scrittore e professore dell'Università di Yale: "leggere un libro su un computer che sta in una sola mano è una vera sofferenza, fisica e stilistica. Lo schermo è piccolo, contiene troppe poche parole, si deve premere un bottone ogni due per tre per cambiare pagina, e l'orecchio interiore che si attiva con Shakespeare o Jane Austen, riducendo al minimo la fatica visiva della lettura, con un e-book rimane sordo come un sasso"[79]. Se è vero che al momento il palmare non rappresenta il massimo dell'ergonomia per la lettura, è anche vero che soprattutto per quanto riguarda le opere di consultazione è uno strumento assai agevole e versatile.

Nonostante lo schermo troppo piccolo i palmtop sono la soluzione economicamente più vantaggiosa per la fruizione di e-book, grazie anche alla recente introduzione di modelli a basso costo.

D'altra parte, paradossalmente, uno dei punti di forza dei palmari sono proprio le sue dimensioni contenute; comincia oggi ad esserci chi, soprattutto fra professionisti costretti spesso a viaggiare per lavoro, invece di portarsi dietro il peso di un libro preferisce caricare un e-book, magari di narrativa, sul proprio palmare, che è anche uno strumento di lavoro, in modo da "sfogliarne" qualche pagina nei tempi morti: ad esempio, in treno, in aereo, in banca. Non si spiegherebbe, d'altro canto, la fortuna, ovviamente nei termini della attuale realtà dell'editoria digitale, di numerose iniziative di editoria elettronica rivolte proprio al mercato dei palmari. Una delle basi del successo di queste esperienze sta proprio nella grande diffusione che hanno questi apparecchi. Tutto questo avvalora la tesi secondo cui una vera e propria affermazione del libro elettronico è subordinata alla capillare diffusione di dispositivi per la lettura.

---

[79] STEFANO PORRO, *Il libro elettronico? Bocciato*, in *Clarence Cultura e Spettacolo* <http://www.clarence.com/contents/culturaspettacolo/societamenti/speciali/000712ebook/001.html> 13 luglio 2000, (17 novembre 2003).

II.4.5. My Friend

Si ricollega per certi versi ai palmari, in quanto basato sul sistema operativo Pocket PC, il primo, ed al momento unico, e-book reader device prodotto in Italia, il *My Friend* della napoletana IPM-Net[80]. Delle dimensioni all'incirca di un libro in edizione tascabile (190 x 200 x 30 mm) e dal peso di poco meno di un chilogrammo, il My Friend è dotato di un display tattile (touch-screen) a colori retroilluminato da 7.5" ad alta definizione (640 x 960, 150 dpi) per garantire una lettura naturale e confortevole; grazie ad una memoria di 32 MB StrataFlash e di 64 MB SDRAM (espandibile per mezzo di schede SmartMedia e PC Card), è in grado di racchiudere in sé oltre ottanta testi, essendo basato sul sistema operativo Pocket PC può inoltre svolgere tutte le funzionalità di un computer palmare. Il Dispositivo integra al proprio interno un Modem 56K V.90 per la connessione ad internet. Il lettore, predisposto per la connettività wireless, integra pure un lettore di smart card in formato SIM (come quella dei telefonini), che viene utilizzato per l'autenticazione dell'utente e per la sicurezza delle transazioni su Internet. Il My Fryend essendo basato su Windows CE 3.0, quindi la prima versione Pocket PC, purtroppo non è in grado di leggere e-book protetti con il sistema Owner Exclusive, ma soltanto Sealed ed Inscribed. Per ovviare a questo problema l'azienda napoletana ha recentemente (fine 2003) stipulato un accordo con la società francese Mobipocket[81] produttrice dell'omonimo software per la lettura di libri elettronici. Il Mobipocket reader è un e-book reader software disponibile per praticamente tutti i sistemi operativi per palmari esistenti (Palm OS, Franklin OS, Pocket PC, EPOC 32 ed altri), oltre al proprio formato proprietario, caratterizzato

[80] IPM-NET S.p.A. Società controllata da IPM Group, è nata nel 2000 per la progettazione, produzione e commercializzazione di terminali multimediali per l'accesso semplice e sicuro ad Internet (Internet Appliances), per lo sviluppo di applicazioni con smartcard per la sicurezza nelle transazioni e per l'offerta di servizi Internet based per l'editoria elettronica (www.ipm-net.com).

[81] MOBIPOCKET è l'unica società che opera nei settori di software e tecnologia che abilita la lettura e la distribuzione sicura di testo elettronico attraverso tutte le piattaforme OS esistenti. MobiPocket si occupa anche della distribuzione in linea di software e-book e e-content tramite il proprio sito web. La Società, fondata in marzo 2000, ha sede a Parigi, Francia ed è finanziata parzialmente dalla Viventures. Dal suo inizio, la Società ha fatto leva notevolmente su VU per i contenuti sinergici e ha stabilito l'associazione con Vivendi Universal Publishing.
(Fonte: <http://www.ipm-net.com/news/mobypocket.htm>)

dall'estensione PRC, supporta formati Palm Doc PDB, XML/HTML/CSS ed il formato TXT. Il formato e-book PRC è come il Microsoft LIT un formato compilato a partire da un pacchetto OEBPS ed integra un sistema di DRM a differenti livelli. Gli e-book in formato PRC vengono venduti, oltre che sul sito della Mobipocket, da numerose librerie on-line. L'accordo con la società francese prevede, oltre l'installazione del Mobipoket Reader sul dispositivo, l'adozione, da parte di IPM-NET, del sistema di Digital Right Management di Mobipocket, con l'ambizione di cercare di offrire una soluzione hardware/software avanzata che si possa collocare come punto di riferimento a livello mondiale nel mercato della lettura elettronica e nella consultazione di handbook sia per uso personale che business. Recentemente l'azienda napoletana ha rilasciato una versione avanzata del My Friend, l'M800 che integra un modulo telefono GSM/GPRS per voce, dati, SMS.

Nel corso del 2003 la IPM-NET in accordo con il Ministero della Istruzione dell'Università e della Ricerca, ha varato il progetto IPM-Scuola: il libro elettronico per la scuola; scopo dell'offerta è quello di consentire alle scuole di accedere a beni e servizi nel campo delle nuove tecnologie dell'informazione e della comunicazione a condizioni agevolate; vengono offerti a scuole, insegnanti e studenti sconti a partire dal quaranta percento sull'acquisto del My Friend e dell'M800; inoltre, la società si impegna ad offrire assistenza per creare in collaborazione con scuole ed università la struttura necessaria a realizzare delle biblioteche digitali, cui accedere in modalità wireless grazie all'e-book reader device.

II.4.6. Tablet PC

Un dispositivo che sembra avere le carte in regola per affermarsi in futuro come e-book reader device è a nostro giudizio il Tablet PC.

Immessi sul mercato nel novembre del duemiladue, i Tablet PC sono sostanzialmente dei particolari computer portatili che cercano di riprodurre in un apparecchio digitale il tradizionale approccio "carta e penna".

Esteriormente un Tablet PC si presenta come una tavoletta, o meglio una lavagnetta, delle dimensioni circa di un foglio A4 in cui (mantenendo l'analogia con la lavagna) la superficie d'ardesia è stata sostituita da un pannello a cristalli liquidi sensibile al tocco.

La configurazione tipo di questi dispositivi è costituita da un display LCD touch-screen (sensibile al tocco) di dimensione oscillante fra gli otto ed i dodici pollici, un processore a basso consumo, un hard disk, supporto per connessioni wireless[82] e pressoché tutte le connessioni tipiche di un computer portatile; al momento l'unico sistema operativo per questi apparecchi è *Windows XP Tablet PC Edition* della Microsoft[83].

Esistono due diversi tipi di Tablet PC, quelli "puri" costituiti unicamente dal display LCD e da una cornice che ospita alcuni tasti e controlli, e quelli "convertibili", dall'aspetto molto simile agli attuali PC portatili, dotati anche di tastiera, tuttavia la tastiera può essere ruotata di 180° dietro lo schermo in modo da poter utilizzare l'apparecchio come un Tablet Pc tradizionale.

Il Tablet PC offre tutti i vantaggi ed è in grado di svolgere tutte le tipiche attività di un PC portatile; la principale differenza con i notebook tradizionali, a parte la possibile assenza di una tastiera, sta nel fatto che il principale dispositivo di input è una penna digitale che serve per controllare il computer e per immettere testo utilizzando la propria scrittura a mano libera. Grazie alla tecnologia battezzata da Microsoft "Digital ink" è possibile con il proprio Tablet Pc inserire testo nelle applicazioni scrivendo a mano utilizzando la penna digitale; inoltre, le informazioni immesse con la penna elettronica possono essere utilizzate immediatamente in diverse applicazioni.

Numerose caratteristiche sia dell'hardware, sia del sistema operativo candidano prepotentemente questo dispositivo come principale e-book reader device per

[82] Col termine Wireless (letteralmente: senza cavi) ci si riferisce ad una tipologia di trasmissione dati in cui i segnali viaggiano nello spazio e non su fili o cavi di trasmissione. In un sistema wireless la trasmissione avviene principalmente per mezzo di onde radio o impulsi infrarosso .

[83] Nel momento in cui scriviamo non è stata ancora rilasciata una versione del sistema operativo localizzata per il mercato italiano, Windows XP Tablet PC Edition è attualmente disponibile soltanto nelle versioni in lingua inglese, tedesca, francese, giapponese, cinese e coreana.

l'immediato futuro. Il Tablet Pc ricalca da vicino l'impostazione di base dei lettori dedicati per e-book, lo schermo si sviluppa in verticale come le pagine dei libri, grazie al display sensibile al tatto si è in grado di usare, nella lettura dei testi elettronici, la penna digitale, alla stessa stregua di una matita, per inserire note, appunti oppure evidenziare parti di interesse. Come detto si cerca con il "Pc Tavoletta" di aggiungere ad un apparecchio elettronico il tradizionale approccio "carta e penna", potenziandolo con gli strumenti dell'informatica; per il testo scritto a mano è possibile effettuare ricerche, cancellare frasi con una gomma elettronica e personalizzare stili, sfondi e colori dell'inchiostro digitale.

Al fine di offrire un'esperienza di lettura quanto più possibile agevole e naturale il sistema operativo sfrutta la tecnologia *Microsoft ClearType*. Un'altra delle importanti caratteristiche tecniche dei Tablet Pc è il riconoscimento vocale, le applicazioni possono essere controllate con la voce invece che tramite mouse, tastiera o penna elettronica; inoltre, il testo da immettere nelle applicazioni può anche essere dettato.

È bene precisare che allo stadio attuale della tecnologia non è tutto oro quello che luccica, in particolare i software per il riconoscimento della scrittura naturale dimostrano scarsa precisione e nemmeno le funzionalità di riconoscimento vocale si dimostrano particolarmente efficienti, non a caso la maggior parte dei Tablet PC venduti appartiene alla tipologia dei cosiddetti "convertibili".

Nonostante tutto il Tablet Pc rappresenta sicuramente un importante passo avanti dal punto di vista dell'ergonomia per la fruizione di testi digitali; basti pensare che la libreria virtuale di Barnes & Noble, poco prima del ritiro del colosso editoriale dal settore del libro elettronico, aveva lanciato una sezione dedicata agli e-book per Tablet Pc, inoltre con più di 6.000 unità vendute soltanto in Italia in soli 8 mesi[84] (periodo dicembre 2002 giugno 2003)[85], i Tablet PC dimostrano di aver

[84] GSMBOX S.P.A., "Tablet PC: *una rivoluzione* tecnologica, in *GSMBOX.IT mobile news* <http://it.gsmbox.com/news/mobile_news/all/96723.gsmbox>
9 luglio 2003, (28 ottobre 2003).

suscitato un notevole interesse in numerosi settori: ad esempio in Italia, sono state create soluzioni per gli informatori scientifici, per la gestione del percorso clinico di pazienti ricoverati in ospedale, per l'automazione della forza vendita e per la gestione magazzino.

Tuttavia, a livello mondiale ad un anno di distanza dal lancio del nuovo prodotto i livelli di vendita sono stati tutt'altro che entusiasmanti con circa centomila unità vendute in tutta Europa, Medio Oriente e Africa, vale a dire meno dell'uno per cento del mercato dei notebook. In Italia, invece, si è registrato un lieve aumento delle vendite, il due percento. Secondo molti analisti le attuali difficoltà di avvio per questa piattaforma sono dovute al prezzo superiore rispetto ai computer portatili, disavanzo di prezzo provocato in particolar modo dal costo delle licenza del sistema operativo Microsoft.

Se si considera che una delle principali cause, che ha limitato l'espansione del libro elettronico è stata proprio la scarsa diffusione dei dispositivi per la lettura, è deducibile che è legittimo sperare in un ampliamento del mercato e-book parallelamente alla crescita di piattaforme per il mobile computing, Tablet Pc, palmari, cellulari avanzati adatti anche a svolgere oltre alle proprie normali funzioni, anche quella di e-book reader device. A nostro giudizio allo stadio attuale della tecnologia, l'e-book reader device ideale è un Tablet PC o un palmare delle dimensioni all'incirca di un libro tascabile, con schermo ad alta definizione e una buona autonomia delle batterie, in modo da poter associare alla potenza del computer nel trattamento delle informazioni testuali la maneggevolezza di un dispositivo versatile e poco ingombrante.

II.4.7. Prospettive future

Sebbene l'attuale tecnologia sia già prodiga di continue innovazioni tecnologiche, l'immediato futuro si prepara ad offrirci alcuni interessanti soluzioni che forse

[85] A riprova di quanto detto si tenga presente che nel periodo in questione i dispositivi in questione era venduti con la versione inglese di Windows XP, la Tablet PC Edition, non essendo ancora disponibile la versione localizzata per il mercato italiano.

potranno rappresentare un decisivo passo avanti dei supporti per il libro elettronico. Nonostante notevole sia l'evoluzione avuta in questi ultimi anni dai display a cristalli liquidi, che si avviano ormai a soppiantare del tutto gli ingombranti monitor a tubo catodico, una nuova tecnologia nota con la sigla OLED, acronimo che sta per Organic Light Emitting Diode, si appresta ad invadere il mercato. I display OLED garantiscono una nitidezza ed una definizione delle immagini superiore alla già buona qualità degli schermi a cristalli liquidi. Basati sui principi della bioluminescenza, i display OLED per una serie di motivi tecnici e commerciali si preannunciano oggi come una delle tecnologie di maggior successo del prossimo futuro. Una delle incarnazioni più sorprendenti degli schermi OLED sarà probabilmente costituita da pannelli-monitor dallo spessore assai ridotto (intorno agli 0,2 millimetri) flessibili quasi come stoffa. In tal senso già da un certo tempo alcuni avanzati laboratori di ricerca lavorano allo sviluppo della cosiddetta e-paper, la carta elettronica ossia display sottilissimi e pieghevoli. Sono due le società oggi particolarmente attive in questo settore la Xerox, con il suo progetto Gyricon, e la E-ink, start-up nata da una costola del prestigioso Massachusetts Institute of Technology di Boston.

La soluzione e-paper della Xerox è costituita da un sottile foglio di plastica elastica trasparente al cui interno sono immerse milioni di sfere bicolori, perlopiù bianche e nere, grandi come un granello di sabbia, per mezzo dell'applicazione di una carica elettrica le sfere vengono ruotate in modo da presentare una delle due facce colorate; orientando opportunamente i milioni di sferette distribuite sulla superficie del foglio plasticato vengono prodotte le immagini da visualizzare, per certi versi i microgranuli bicolori possono essere considerati come i pixel dei display tradizionali.

Simile a Gyricon nel principio di funzionamento è la tecnologia E-ink. La carta elettronica della E-ink è anch'essa costituita da un pannello plastico flessibile, in cui sono annegate milioni di minuscole sfere chiamate microcapsule. Ognuna di esse ha un involucro trasparente ed è riempita da un liquido trasparente, in cui

sono sospese delle microscopiche particelle: alcune di colore bianco caricate positivamente, altre di colore scuro caricate negativamente. Grazie all'applicazione di adeguate cariche elettriche vengono fatte emergere a secondo delle necessità le particelle bianche o nere, così da formare i caratteri del testo e le immagini. Oltre alla estrema maneggevolezza e trasportabilità un importante vantaggio di queste tecnologie è rappresentato dal fatto che mentre un tradizionale display consuma con continuità energia elettrica per mantenere le informazioni a video, queste soluzioni assorbono energia solo nel momento di cambiare l'immagine, ciò rappresenta indubbiamente un vantaggio dal punto di vista dei consumi.

Se queste tecnologie riusciranno a superare alcuni dei problemi che al momento le affliggono i dispositivi del futuro per la fruizione di e-book potrebbero anche essere strumenti sottili, portatili, flessibili, arrotolabili in grado di visualizzare testo e immagini.

II.5. STAMPA SU RICHIESTA

Nel capitolo sull'hardware per il libro elettronico ho affermato che "il device di lettura sta al libro elettronico come la carta sta al libro tradizionale"; tuttavia, almeno per il futuro prossimo, sembra proprio che continuerà ad essere la carta il medium principale cui sarà affidato il testo, compreso quello dei libri elettronici.

"Oggi si tende a presupporre che i libri e altri testi digitali verranno letti sopratutto sullo schermo del computer o su strumenti di lettura da tenere in mano, come i Palm Pilots o i lettori Gemstar. Ma per ora non è ancora emerso un mercato per i libri da leggere su schermo, e a mio avviso questo potrebbe anche non diventare mai il modo più diffuso di distribuire i libri on line. Credo invece che la prospettiva più probabile sia che la maggior parte dei file digitali vengano stampati e rilegati su richiesta in punti vendita dotati di macchine - per ora in prototipo - che nel giro di minuti sono in grado di creare a bassissimo costo singole copie praticamente indistinguibili dai libri fabbricati dall'industria editoriale."[86]

Cosi si esprime Jason Epstein nel suo intervento al convegno virtuale tenutosi sul sito www.text-e.org "Schermi e reti, verso una trasformazione della scrittura?"[87], presagendo un futuro probabilmente roseo per la stampa su richiesta . Il Print-on-demand o stampa su richiesta è una tecnologia in base alla quale il contenuto dei libri è archiviato in formato digitale in una struttura hardware, collegata con un sistema di stampa digitale ad alta qualità ed alta velocità per mezzo del quale è possibile ottenere un numero indefinito di copie stampate e rilegate del libro (anche una sola).

[86] JASON EPSTEIN, *Leggere: il futuro digitale*, in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=13>
trad. it. a cura di Delfina Vezzoli, l'articolo era già apparso nel The New York Review of Books n° 11 vol 48 del 5 luglio del 2001 con il titolo *Reading: The Digital Future*.

[87] Il convegno virtuale è stato organizzato dalla Biblioteca Pubblica di Informazione del Centre Pompidou, dall'Institut Jean Nicod (CNRS) e dall'Associazione EURO-EDU con il concorso della società GiantChair sotto il patrocinio dell'UNESCO, il convegno si è tenuto sul sito <www.text-e.org>, che ospita gli interventi dei relatori.

Sono numerosi i vantaggi che questa tecnologia offre ad editori, autori e lettori. In primo luogo grazie alla stampa su richiesta è possibile tornare a rendere disponibili titoli ormai fuori stampa.

"In Italia, dove i titoli esauriti nel 1999 sono stati 29.374, si stima che una percentuale fra il 15 e il 20% dei titoli, che pure risultano in commercio, non siano in realtà disponibili, causando con ciò non solo disservizi e aumento di costi di gestione degli ordini ma soprattutto perdita di fatturato per le case editrici"[88].

Un altro grande vantaggio di questa tecnologia è che gli editori vengono liberati dal peso e dai consistenti costi dei magazzini, il parco opere di una casa editrice può essere costituito da bit e solo all'occorrenza il libro viene "materializzato". Sono ovvi anche i vantaggi per quanto riguarda il trasporto ed i tempi di consegna dei titoli. È ipotizzabile che in un futuro più o meno prossimo vedano la luce dei chioschi informatici per il print-on-demand, una sorta di distributori automatici self-service del libro, un po' come già avviene per bibite e merendine.

Un importante contributo allo sviluppo di sistemi di stampa su richiesta controllati da remoto potrà forse darlo in futuro l'Internet Printing Protocol (IPP). Un nuovo protocollo internet, che si integrerà a quelli esistenti, grazie al quale una stampante collegata in rete potrà ricevere comandi da qualsiasi computer collegato, si comporterà in sostanza come un server. Ancora in via di definizione, lo sviluppo del protocollo è affidato al Printer Working Group[89], un consorzio internazionale costituito da produttori di hardware e software come Microsoft, Sun e IBM, e da enti che si occupano di standardizzazione e protocolli Internet.

Sono numerose le iniziative per il print-on-demand che in questi ultimi anni hanno visto la luce; in Italia una delle prime è stata *Lampi di stampa*[90], varata nel 1998 dall'Editrice Bibliografica, nota in Italia soprattutto in quanto titolare per il nostro paese della gestione ed assegnazione dei codici ISBN (International Standard Book Number), dalla Legoprint, una importante azienda grafica italiana, e da

[88] Cfr. BRUNELLA LONGO, *La nuova editoria*, Milano, Editrice Bibliografica, 2001, p. 165.
[89] <www.pwg.org>.
[90] <www.lampidistampa.it>.

Messaggerie libri, il maggiore distributore italiano. Lampi di stampa è una società che, utilizzando le tecnologie digitali per il print-on-demand, si occupa di stampare su richiesta, in accordo con gli editori o con chi ne detiene i diritti (autori, biblioteche, università, ecc.) i libri esauriti o non più disponibili nel mercato librario. Come leggiamo nelle pagine del sito, l'azienda offre agli autori e agli esordienti (in collaborazione con la società Libuk[91], la prima libreria virtuale italiana che mette a disposizione la versione e-book di libri di varie case editrici e di opere classiche e contemporanee inedite o fuori commercio) la possibilità di pubblicare i propri romanzi, saggi, poesie, inserendoli successivamente nel mercato librario.

Agli editori offre la possibilità di riproporre molti titoli esauriti, stampandoli su richiesta, anche una copia per volta, oppure in microtiratura (a partire da un minimo di 100 copie) distribuendoli alle librerie mediante Messaggerie Libri.

Alle università offre la possibilità di stampare in digitale, dietro richiesta, i testi e le dispense dei corsi, oltre a tutte quelle pubblicazioni che potrebbero avere una domanda commercialmente significativa.

Alle biblioteche offre la possibilità di digitalizzare e ristampare in anastatica sia quei volumi che, per la loro frequente consultazione sono maggiormente soggetti a deterioramento, sia quelle opere di grande interesse particolarmente richieste dal pubblico, ma non più disponibili nelle librerie.

Agli enti locali offre la possibilità di recuperare alcune pubblicazioni di rilevante interesse storico-locale, culturale o artistico, da loro edite e disponibili solo in copie d'archivio.

Il catalogo di vendita di Lampi di Stampa viene diffuso presso tutte le librerie. Legoprint si occupa della produzione dei libri, mentre Messaggerie Libri si occupa di tutti gli aspetti riguardanti la raccolta degli ordini e la distribuzione dei titoli alle librerie. Tra gli editori che si giovano dei servizi della società milanese vi sono: Hoepli, Il Saggiatore, Iperborea, Laterza, Longanesi, Tea, Zanichelli, Garzanti.

[91] <www.libuk.com>.

Molte delle soluzioni per il print-on-demand esistenti si servono della tecnologia Adobe Postscript e, quindi, anche del formato PDF per l'archiviazione digitale dei titoli.

La stampa su richiesta rappresenta soltanto una delle incarnazioni possibili del libro elettronico, un e-book, se di e-book si può parlare, per il print-on-demand deve necessariamente avere caratteristiche di descrizione della pagina dal punto di vista tipografico adeguate alla produzione di un output di stampa di qualità (in ciò il PDF è particolarmente efficace) cosa che però talvolta non corrisponde ad una buona fruibilità a video. Riteniamo che il primo stadio di un e-book debba essere la codifica per mezzo di un linguaggio di marcatura logica del testo basato su XML, in modo da poter avere da un unico sorgente diversi output; grazie a tecnologie come i fogli di stile XSLT, sarà in seguito possibile ottenere i diversi formati finali adeguati alle varie esigenze, stampa, riproduzione a video etc..

Desideriamo concludere questa breve carrellata sulla stampa su richiesta con le parole di Umberto Eco, che interrogato nell'ambito della conferenza virtuale "Schermi e reti, verso una trasformazione della scrittura?" già citata, sul print-on-demand illustra una sua simpatica previsione della libreria del futuro: "ci sono infiniti motivi per cui il *print on demand* potrebbe essere culturalmente ed economicamente molto positivo, anzi, ho già detto che non faccio profezie, ma se proprio ne dovessi fare una da fantascienza vedrei una libreria del futuro che ha soltanto, come certi luoghi di videocassette pornografiche, la finta cassetta con la copertina e poi uno va a chiedere di stamparla... questo forse è un po' esagerato perché il piacere della libreria sta ancora nello sfogliare, annusare il libro, ma potrebbero bastare alcune copie pilota per la libreria, anche quelle *printed on demand*, senza passare per la tipografia, e poi in cinque minuti il cliente può avere il libro. La mia previsione è che l'ultimo libro di Stephen King probabilmente sarebbe stampato perché la gente deve comperarlo come panini in fretta, mentre

se uno vuole leggere *Thérèse Desqueyroux* di Mauriac, che magari è *out of print,* può farselo stampare."[92]

[92] UMBERTO ECO, *Autori e autorità*, in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=11>.

II.6. CRITTOGRAFIA E FIRME DIGITALI

È ormai da qualche anno entrata a far parte della già folta schiera di acronimi relativi al mondo internet la sigla DRM, abbreviazione che sta per Digital Rights Management, e che viene utilizzata per indicare tutte le metodologie volte alla gestione dei diritti dei prodotti intellettuali. DRM è la risposta dell'industria al crescere dell'importanza dei canali telematici come Internet per la distribuzione di beni digitali. Una delle caratteristiche degli oggetti digitali è l'estrema riproducibilità, se questo è un vantaggio in numerose circostanze, tuttavia rappresenta indubbiamente un problema allorché l'oggetto digitale è portatore di contenuti soggetti a copyright.

Questo problema si è palesato con tutta la sua evidenza in occasione della cosiddetta guerra degli Mp3.

II.6.1. La guerra degli Mp3

Mp3[93] è un formato digitale audio compresso, ottenuto per mezzo di un sistema di codifica che consente di ridurre notevolmente le dimensioni in byte di un file audio e, quindi, di renderlo più facilmente trasmissibile. Sebbene le specifiche del formato fossero state rilasciate già nel 1993, tuttavia soltanto intorno al 1998, quando la potenza dei computer domestici era ormai adeguata alle onerose esigenze di calcolo degli algoritmi di codifica e decodifica, si è assistito ad una vera e propria esplosione dell'Mp3. Congiuntamente all'affermazione dell'Mp3 si registrava in quegli anni la crescita esponenziale della rete internet, e frattanto iniziava pure la diffusione dei masterizzatori per Cd-Rom. Vennero a coincidere sul finire degli anni novanta, quindi, una serie di circostanze che produssero una crescita enorme ed inaspettata, della pirateria musicale. Non è che prima di allora

[93] Il termine Mp3 deriva dalla contrazione di MPEG Audio Layer 3. Le specifiche del formato Mp3, il cui sviluppo era iniziato nel 1987 in Germania per opera del Fraunhofer IIS, sono state rilasciate nel 1993 da parte del Mpeg (Moving Picture Expert Group) gruppo di lavoro operante in seno all'ISO incaricato di mettere a punto dei sistemi completi per la codifica e la decodifica di video e di audio compressi, da definire come standard da applicare ai nuovi media quali TV digitale, DVD, Internet.

non vi fossero fenomeni di pirateria; tuttavia, il reato della copia non autorizzata era perpetrato principalmente per mezzo delle audiocassette, elemento che richiedeva in primo luogo la necessità dell'acquisto di un supporto materiale per la registrazione, l'audiocassetta, ed in ogni caso la procedura di copia comportava uno scadimento qualitativo; in secondo luogo ove non si fosse deciso di approvvigionarsi sul mercato delle cassette pirata, che pur sempre avevano un certo costo, la possibilità di scambiarsi musica era limitata al giro dei propri amici. Sino a quel momento si potrebbe dire che per le case discografiche e gli artisti il fenomeno della copia non autorizzata era, per così dire, entro limiti fisiologici, d'altra parte era possibile un'attiva lotta contro le grandi organizzazioni criminali dedite alla distribuzione illegale di musica. La situazione della scambio illegale di brani musicali invece è cambiata drasticamente, assumendo le dimensioni di un vero e proprio fenomeno di massa, con volumi di scambio che hanno assunto dimensioni preoccupanti per i protagonisti del mercato musicale, in seguito alla diffusione delle reti Peer to Peer (abbreviato P2P). Quella P2P è una particolare architettura di rete come dice il nome "da pari a pari", che non è basata sulla tradizionale impostazione client/server tipica della rete internet, bensì su di un tipo di connessione, per così dire "paritaria", in cui, in parole molto povere, i singoli computer della rete svolgono allo stesso tempo funzioni di client e di server; questo tipo di connessione scavalca completamente la problematica del sovraffollamento dei server in caso di traffico intenso. Le reti Peer to Peer sono state utilizzate pressoché esclusivamente per creare sistemi di "Filesharing", ossia sistemi per lo scambio di file. Nel settembre del 1999, un programmatore di appena diciotto anni, Shawn Fanning, mise a disposizione un software gratuito per la condivisione di file musicali via Internet: Napster. La coincidenza dell'affermazione della tecnologia Mp3 che consentiva di creare dei file di dimensioni ridotte, quindi adatti a viaggiare su reti a bassa ampiezza di banda, come era allora internet per la maggior parte degli utenti (ADSL era ancora in fase di sviluppo all'epoca), con la diffusione delle tecnologie per il Filesharing, come

detto produssero un fenomeno di massa, quasi una moda, con più o meno consistenti riduzioni degli introiti delle case discografiche a causa del proliferare delle copie illegali di musica digitale. Rispetto ai tempi delle audiocassette l'unico, esiguo, costo che si doveva affrontare per approvvigionarsi di musica in modo illegale era quello della connessione, inoltre il poter accedere ad una rete diffusa su praticamente tutto il pianeta dava la possibilità di avere accesso ad un archivio musicale pressoché infinito. Le varie società per la tutela dei diritti d'autore, prima fra tutte la RIAA, la Recording Industry Association of America, associazione dei discografici americani omologa per certi versi alla nostrana SIAE, intrapresero un'imponente crociata contro Napster ed i sistemi di Filesharing. La battaglia fu combattuta su più fronti quello legale e quello tecnologico. Furono intentate numerose cause contro i gestori di reti per lo scambio file e soprattutto contro Napster, cosa che portò nel 2001, dopo numerose limitazioni, alla chiusura del sito Napster ed alla cessazione delle attività della sua rete; sul fronte tecnologico ci si attivò per la produzione di tecniche atte ad impedire la duplicazione illegale dei Cd audio e l'estrazione in formato Mp3 delle tracce, tecnologie che però sono state tutte sistematicamente aggirate. Nel 1998 sotto la spinta della lobby dell'industria dell'intrattenimento, il congresso americano ha varato il Digital Millenium Copyright Act, un provvedimento legislativo per la protezione della proprietà intellettuale nell'era digitale, che offre ampi, secondo taluni, eccessivi, poteri a titolari dei diritti d'autore per la difesa degli stessi. La cosiddetta guerra degli Mp3 non si è conclusa con la chiusura di Napster, infatti al capostipite dei software di filesharing, sono seguiti numerosi epigoni che permettono non soltanto lo scambio di Mp3 ma anche di qualsiasi tipo di oggetto digitale, software, immagini, video, etc.[94]. La battaglia per la protezione dei diritti d'autore imperversa a tutt'oggi, ultimamente (2003) le case discografiche, giovandosi dei poteri concessigli dal

[94] Occorre notare che la chiusura di Napster è stata possibile in quanto questo sistema si basava su di un'architettura di rete ibrida, che ricorreva ancora ad un sistema di server per indirizzare le richieste degli utenti mentre la connessione P2P era usata per lo scambio files. È quindi bastato oscurare i server centrali per bloccare l'intero sistema. Ciò non sarebbe possibile con reti peer to peer pure quali Gnutella, dove non esistendo dei server centralizzati, ma essendo la rete stessa costituita dai computers degli utenti, si dovrebbero bloccare i computer di tutti gli utenti per arrestare la rete.

Digital Millenium Copyright Act, dopo aver costretto i provider internet a fornire i nomi di loro utenti colpevoli di attività di Filesharing, hanno iniziato a denunciare i singoli utenti chiedendo risarcimenti stratosferici, al momento in cui scriviamo la vicenda, che ha suscitato grande scalpore, è ancora in corso. Nonostante tutto ciò, in questi ultimi anni lo scambio file via internet si è esteso anche ai film, complice l'introduzione di nuovi efficaci formati di compressione video, una sorta di Mp3 per i il video, e la diffusione del DVD. Forti della precedente esperienza delle major discografiche, i soggetti del mercato cinematografico hanno cercato di tutelarsi al meglio contro la pirateria inserendo un sofisticato sistema di protezione nei DVD, tuttavia ben presto anche questo è stato violato ed i software per la decodifica dei filmati contenuti nei DVD hanno iniziato a circolare su internet. Inoltre in modo rocambolesco, si dice sia stato trafugato da un hacker francese, è stato diffuso sul web un codec (un programma per la codifica video) sviluppato da Microsoft per la trasmissione di video in rete, che consentendo alti gradi di compressione senza un eccessivo scadimento qualitativo, ben si presta a ricomprimere i filmati estratti dai DVD grazie ai software di sprotezione di cui abbiamo appena detto, così anche i film hanno iniziato a circolare nelle reti di Filesharing, complice l'affermazione della banda larga, Adsl fibra ottica etc.. Si può dire che una sorta di nuovo fenomeno Mp3 si sta ricreando in questi anni per il video, ma la battaglia tra titolari dei diritti e pirati è ancora incorso ed imperversa più aspra che mai. Forse il problema più grande della pirateria digitale veicolata attraverso le reti P2P, è il fatto che questa attività non è percepita come un qualcosa di illegale o immorale, bensì come una pratica normale, è tale la diffusione del fenomeno e la consuetudine tra gli utenti dei software di filesharing allo scaricare materiale di ogni genere, che probabilmente ci si deve scontrare con un problema di natura culturale. Le case discografiche si trovano a dover combattere non più soltanto contro ben precise organizzazione criminali, ma sempre più spesso contro i propri clienti.

II.6.2. DRM e crittografia

Tutta l'industria dei contenuti,in particolar modo quella discografica principale vittima della pirateria digitale, ha dunque avvertito una crescente necessità di trovare meccanismi che le permettano di proteggere le proprietà intellettuali di quei beni che, già oggi, possono essere distribuiti e venduti attraverso il Web.

La risposta a questa esigenza è venuta dalle tecnologie di DRM, focalizzate nel risolvere i problemi legati alla copia non autorizzata di musica, film o e-book, per mezzo di tecniche crittografiche, firma digitale e gestione delle licenze.

La crittografia è la scienza che si occupa dell'elaborazione di sistemi di scrittura segreti intellegibili soltanto da chi è a conoscenza del codice usato per comporla, detto chiave di decodifica. L'uso di tecniche crittografiche ha origini antiche, Svetonio nella Vita dei Cesari racconta che Giulio Cesare usava per le corrispondenze con i suoi generali un sistema di sostituzione molto semplice, egli cambiava ogni lettera del messaggio con la lettera che la segue di tre posizioni più avanti nell'alfabeto, la A diventava D, la B diventava E, la C diventava F e così via fino alle ultime lettere che sono cifrate con le prime. Usando il sistema di Cesare la frase *libro elettronico* diventerebbe *oleur hohwwurqlfr*, la chiave di codifica del messaggio è "3", basta che al destinatario del messaggio venga comunicata questa chiave perché sia in grado di decifrarlo. Nel sistema di Cesare la chiave usata per codificare è uguale a quella per decodificare, in questo caso si parla di crittografia simmetrica; nel corso dei secoli sono stati sviluppati numerosi sofisticati sistemi crittografici basati sulle più disparate tecniche di sostituzione e trasposizione delle lettere del messaggio, tutti però si basavano sulla crittografia simmetrica. È una scoperta abbastanza recente la crittografia asimmetrica. Il punto debole dei sistemi di cifratura tradizionali è sempre stato la necessità di comunicare in modo riservato la chiave di decodifica con la certezza che nessuno ne venga a conoscenza, problema divenuto particolarmente rilevante nel secolo appena trascorso con lo sviluppo delle comunicazioni a distanza, prima con la radio ed ultimamente con le reti informatiche. Una soluzione a questo problema è venuta

dalla crittografia asimmetrica, nel 1976 gli studiosi Diffie ed Hellman hanno descritto un protocollo per lo scambio di una chiave segreta sopra un canale insicuro, il sistema è detto asimmetrico in quanto è basato su l'uso di due chiavi generate in modo che sia impossibile ricavarne una dall'altra. Le due chiavi vengono chiamate pubblica e privata: la prima serve per cifrare e la seconda per decifrare. Il primo sistema pratico di crittografia a chiavi pubbliche basato sui concetti proposti da Diffie ed Hellman fu sviluppato nel 1978 da tre professori: Ronald Rivest, Adi Shamir e Leonard Adleman, che battezzarono la propria tecnica di cifratura RSA, dalle iniziali dei tre autori. Il metodo RSA si basa sulla fattorizzazione di interi di grandi dimensioni ed altri complessi calcoli matematici per creare le due chiavi pubbliche e privata. Le chiavi sono matematicamente collegate tra di loro, teoricamente sarebbe possibile risalire da l'una all'altra, tuttavia allo stadio attuale della tecnologia informatica i calcoli necessari per eseguire questa operazione impiegherebbero centinaia di anni. Riteniamo possa essere utile un piccolo esempio per aiutare a comprende come funzionino praticamente i sistemi di crittografia asimmetrica. Supponiamo che io debba intraprendere una corrispondenza segreta con un'altra persona, entrambi dobbiamo prima scambiarci le rispettive chiavi pubbliche, quindi se io devo mandare un messaggio al mio interlocutore provvederò a crittarlo con la sua chiave pubblica ed inviarglielo, ricevutolo egli lo decritterà usando la sua chiave privata. Nel momento in cui un messaggio è codificato con una chiave questo potrà essere decodificato soltanto grazie alla chiave corrispondente, quindi una volta che ho codificato il messaggio con la chiave pubblica del mio interlocutore non potrò più accedervi, solamente lui con la propria chiave privata potrà aprirlo. Quando il mio interlocutore deciderà di rispondermi adotterà il medesimo procedimento, codificherà con la mia chiave pubblica e in seguito io provvederò alla decodifica con la mia chiave privata. Usando una fantasiosa analogia potremmo paragonare la crittografia asimmetrica ad un lucchetto che invece di avere un'unica chiave che lo chiude e lo apre, ha due chiavi la prima è in grado solamente di chiudere il

lucchetto, la seconda è in grado di aprirlo, ma solo se è stato chiuso correttamente con la prima. Come detto si basano principalmente sui principi della crittografia asimmetrica le tecnologie di DRM, semplificando estremamente possiamo dire che nelle transazioni online i software per la lettura generano due chiavi una pubblica con cui viene crittato il libro elettronico ed una privata che viene spesso associata in modo univoco con un determinato dispositivo hardware in modo tale che l'e-book possa essere letto solo su quello specifico apparecchio.

II.6.3. Firma digitale

La crittografia oltre alla tutela dalla copia non autorizzata fornisce importanti soluzioni per altre due legittime esigenze: la tutela del consumatore e la tutela della proprietà intellettuale in ambito digitale. Come è possibile essere certi delle informazioni cui si accede, come si può fare a sapere che quanto si legge non è stato modificato da qualcuno, come ci si può assicurare della provenienza di un testo? Ogni autore è naturalmente interessato che la sua opera sia sempre legata al suo nome e non venga alterata in alcun modo. Anche a queste esigenze dà una risposta la crittografia asimmetrica per mezzo della firma digitale. La crittografia a chiave pubblica/privata non viene usata soltanto per la trasmissione di informazioni riservate ma anche, sfruttando in modo inverso il principio delle chiavi, per la firma digitale dei documenti elettronici. Abbiamo detto che le due chiavi pubblica e privata sono collegate, tutto ciò che viene codificato con la chiave pubblica può essere decodificato solo con la corrispondente chiave privata, viceversa quanto è codificato con la chiave privata può essere decodificato solo con la relativa chiave pubblica. Se ad esempio cifriamo un messaggio con la nostra chiave privata, quel messaggio cifrato potrà essere letto da tutte le persone che possiedono la nostra chiave pubblica. Se queste riusciranno a decifrare il messaggio avranno quindi la certezza che è stato inviato dal proprietario della chiave pubblica usata per la decifratura, ossia da noi. Tutto è garantito dal principio base della crittografia asimmetrica, dal fatto cioè che la chiave segreta

decifra solo ciò che è stato cifrato con la rispettiva chiave pubblica e viceversa.[95] In realtà in questo caso l'unica certezza che abbiamo è che il messaggio da noi ricevuto proviene dal possessore di una certa chiave privata, ma come si fa a esser certi che chi ci scrive sia effettivamente chi afferma di essere?

Per garantirci dell'identità di un individuo, ossia che sia proprio quell'individuo ad avere disponibilità di una certa chiave privata occorre che una terza parte, la cosiddetta Autorità di Certificazione, emetta un certificato personale da associare alla firma digitale di quel soggetto. Un certificato personale è il corrispettivo digitale della carta d'identità, come questa certifica la nostra identità, di cui si fa garante la Pubblica Amministrazione, così questa garantisce l'identità del proprietario di una certa chiave privata, di cui si fa garante un'Autorità di Certificazione. Una Autorità di Certificazione è semplicemente una entità che riceve un insieme di informazioni, le verifica e le garantisce rispetto a una terza parte, in pratica una sorta di notaio telematico[96].

Un certificato emesso da una Autorità di Certificazione è firmato con la chiave pubblica dell'Autorità di Certificazione e tipicamente contiene:

- la chiave pubblica del possessore;
- i dati anagrafici del possessore, se il possessore è una persona fisica, altrimenti se invece è un server web sarà presente l'indirizzo web e il nome della compagnia proprietaria del server;
- la data di scadenza della chiave pubblica;
- il nome dell'Autorità di Certificazione che ha emesso il certificato;
- la firma digitale della Autorità di Certificazione che ha emesso il certificato.

[95] Questo sistema presenta il rischio che qualcuno si possa impadronire della chiave privata di un soggetto e spacciarsi per lui, per evitare tutto ciò la chiave privata è spesso protetta da password o PIN, di fatto è molto più sofisticato falsificare una firma digitale che una tradizionale carta d'identità, è comunque ovvio che è opportuno custodire con cura la nostro chiave privata così come facciamo con la nostra carta d'identità.

[96] In Italia la cosiddetta legge Bassanini (Legge n° 59 del 15 Marzo 1997) ha dato valore legale alla firma digitale, e ne ha regolamentato le procedure per il territorio nazionale, stabilendo tra l'altro un Albo pubblico per le Autorità di Certificazione. Esistono da tempo numerosi enti privati che svolgono funzioni di Autorità di Certificazione attivi soprattutto a livello internazionale, i certificati digitali sono infatti molto utilizzati nelle transazioni sicure su internet.

La firma digitale dell'Autorità di Certificazione apposta al certificato è per così dire il corrispettivo digitale dei timbri apposti alla carta d'identità. La firma digitale oltre a permettere di verificare la provenienza di un documento consente anche di accertarsi che il documento non sia stato alterato o modificato da qualcun altro, anche, per ipotesi, dallo stesso destinatario. Per far ciò si ricorre ad una particolare tecnica chiamata funzione di Hash o Hashing. La funzione di Hash è un algoritmo che partendo da un documento di qualsiasi dimensione lo elabora e produce un codice di dimensione fissa, una sorta di impronta digitale del file. L'hash non è altro che un codice, una sequenza di bit, che viene utilizzata per vedere se dei dati (una e-mail, un certificato digitale, o qualsiasi altra cosa) sono stati in qualche modo modificati da qualcuno. Ad esempio se creiamo un file di testo con questo contenuto "Libro elettronico e-book, prova hash." il suo codice di hash calcolato con l'algoritmo SHA1 sarà "76836460fcac1258254e6d00961c5bf59db93ae3", se eliminiamo il punto finale dalla frase di esempio il nuovo valore di hash sarà "854d99cca59c7e2eabeb3ba530e34fdbb22efe5c" come si può vedere il valore è completamente diverso. L'hashing produce codici di dimensione fissa indipendentemente dalla grandezza del documento. La funzione di Hash è conosciuta anche come *one way hash* in quanto dato il valore di hash è impossibile risalire al documento. Tipicamente una firma digitale contiene, crittografati con la sua chiave privata, i dati del soggetto firmatario, eventuali certificati che ne attestano e garantiscono l'identità, e l'hash del documento che assicura che questo non è stato modificato, infatti nel caso in cui il documento venisse modificato l'hash non corrisponderebbe più con quello indicato nella firma, e quindi la firma stessa sarebbe non valida. Come si può comprendere le tecnologie di firma digitale costituiscono degli strumenti molto utili anche in ambito e-book, per la certificazione della originalità, della integrità e della provenienza dei libri elettronici. Ad esempio il formato Microsoft Lit integra funzioni di Hash per impedire che gli e-book possano essere alterati, infatti in caso di modifiche al file il

Reader non permette di aprirlo. Tra le cause che hanno frenato la diffusione dell'e-book vi è stato il timore della pirateria da parte di editori ed autori. Sino ad oggi i sistemi di protezione dei diritti d'autore spesso non si sono rivelati abbastanza solidi, è indubbio che la sfida del futuro per il mercato del libro elettronico si chiama DRM, sfida che si prospetta tutt'altro che facile, infatti uno dei motti che circola nelle comunità hacker della rete è: "ciò che è scritto può essere letto".

II.7. VANTAGGI E LIMITI DEL LIBRO ELETTRONICO

Il libro elettronico è un oggetto digitale, così infatti lo definisce l'Open eBook Forum nel documento *A Framework for the Epublishing Ecology*: "un'opera letteraria sotto forma di oggetto digitale, costituito da uno o più identificatori standard, un insieme di metadati e un blocco di contenuto monografico, realizzata per essere pubblicata e consultata mediante dispositivi elettronici". Buona parte dei vantaggi e dei limiti del libro elettronico risiedono nella sua "esistenza digitale" ovvero nel fatto che sia un oggetto costituito di bit. Nel paragrafo "bit ed atomi" abbiamo elencato genericamente le caratteristiche dell'oggetto digitale, desideriamo ora riprendere quell'elenco per illustrare più da vicino le proprietà dell'e-book.

II.7.1. Immaterialità

L'immaterialità rappresenta probabilmente la principale caratteristica degli oggetti digitali e quindi anche dell'e-book, come vedremo nel prosieguo della trattazione; questo comporta dei vantaggi sia per quanto riguarda la trasmissibilità , sia per quanto riguarda la conservazione dei libri elettronici. È da valutare se l'immaterialità dell'e-book costituisca un vantaggio per l'annoso problema della tutela ambientale. "Ogni anno il pianeta perde 14 milioni di ettari di copertura arborea, una superficie che equivale più o meno alla Grecia. L'industria della carta contribuisce alla deforestazione nella misura di circa il 20%, trasformando in pasta di cellulosa il 42% di tutto il legname prelevato a scopi industriali"[97]. Inoltre numerose statistiche dimostrano che la diffusione delle office automation e dei dispositivi di stampa personale, hanno favorito la crescita del consumo mondiale di carta. Paradossalmente l'entrata dell'informatica negli uffici e nelle case ha prodotto un incremento dei consumi di carta, sopratutto della carta formato A4,

[97] ANDREA ADDOBBATI, *Incartati nella rete. Inchiesta sui consumi di carta nell'epoca di Internet*, in *Athenet On Line, notizie e approfondimenti dall'Università di Pisa N°4 settembre 2001*
<http://www.unipi.it/athenet/04/articoli/0004Carta_box3A.html>
19 settembre 2002, (1 dicembre 2003).

quella delle stampanti e delle fotocopiatrici. La produzione della carta è un'attività altamente affamata di energia, comporta uno spaventoso consumo di acqua e rilascia nell'ambiente sostanze tossiche e inquinanti. Come ben si può comprendere un maggior uso del digitale per la trasmissione, fruizione e conservazione delle informazioni testuali (documenti, libri, etc.) potrebbe dare un valido contributo alla lotta per la tutela dell'ambiente.

II.7.2. Trasmissibilità

La natura materiale del libro tradizionale fa sì che sia soggetto a tutti i problemi della distribuzione fisica. Una volta stampato il libro deve essere fatto arrivare nelle librerie, con tutti i costi e le problematiche logistiche che questo comporta. Il libro elettronico ha nella grande trasportabilità uno dei suoi più grandi vantaggi. Attraverso l'etere, cavi telefonici, fibre ottiche, un libro elettronico in pochi secondi può esser trasmesso da un capo all'altro del pianeta, a costi sensibilmente inferiori a quelli che si affronterebbero per il trasporto di un libro cartaceo. Considerando che le attività di trasporto tradizionale contribuiscono in modo consistente all'inquinamento ed al consumo energetico, da questo punto di vista il libro elettronico ed eventualmente il Print on Demand, sono sicuramente da prendere in considerazione tra le soluzioni per la creazione di un nuovo modello di sviluppo sostenibile per l'ambiente.

II.7.3. Conservazione

I bit non occupano un vero e proprio spazio fisico se non quello del supporto, che li ospita. I libri tradizionali vengono stoccati in magazzini che hanno dei costi per gli editori, le biblioteche si scontrano spesso con la carenza di spazio e col crescere delle dimensioni della propria collezione libraria sorge il problema dell'ampliamento dei locali. Con i file digitali questo problema è praticamente irrilevante. Ma alla questione della conservazione nello spazio si affianca quello della conservazione attraverso il tempo. Dobbiamo all'attività instancabile dei monaci dei monasteri medievali se il patrimonio testuale dell'antichità è almeno in parte giunto sino a

noi. Nel corso dei secoli le biblioteche si sono adoperate per la preservazione e la diffusione della conoscenza, adeguandosi nel tempo alle successive trasformazioni delle tecnologie di produzione e riproduzione dei libri. Al libro, importantissimo strumento per la crescita della nostra civiltà, è stato spesso affidato il compito di custode nel tempo della cultura umana. Anche nell'era del digitale si ripropone la sfida della preservazione nel tempo, alcuni papiri custoditi dalle sabbie del deserto sono sopravissuti per ben due millenni, oggi corriamo il rischio che a causa della rapida obsolescenza tecnologica testi prodotti appena qualche decennio prima divengano inaccessibili nel giro di breve tempo. In questo senso è importante l'adozione di tecnologie e formati standard che diano garanzie di persistenza nel tempo, formati come l'ASCII prima e l'XML oggi, hanno quelle caratteristiche di indipendenza dall'ambiente tecnologico e durevolezza che li rendono adeguati alla preservazione dell'informazione nel tempo[98]. L'informazione digitale è virtualmente eterna, infatti affidata a supporti durevoli è in grado di riproporre inalterata gli stessi contenuti tanto oggi quanto fra duemila anni. La carta di cellulosa ha una vita limitata, con gli anni ingiallisce e tende a sbriciolarsi, la digitalizzazione di alcuni testi stampati dopo la seconda metà dell'ottocento quando si è smesso di stampare su carta di stracci, che aveva una maggior resistenza, per passare alla carta di legno, è divenuta oggi una vera e propria necessità per la preservazione di tanti volumi che vanno rapidamente deperendo. I supporti di memorizzazione magnetici e ottici da noi utilizzati per l'archiviazione dei documenti digitali hanno un'aspettativa di vita limitata e possono essere soggetti a danni e perdita di dati. Per tal motivo è opportuno che vi siano delle istituzioni che si occupino della conservazione nel tempo dell'informazione digitale, sia riversando periodicamente i dati dai vecchi supporti su supporti nuovi (refreshing), sia adottando soluzioni specifiche per la preservazione nel tempo. In questi anni numerose biblioteche stanno sperimentando nuovi supporti di memorizzazione, che sembrano garantire

[98] Per una approfondita trattazione delle problematiche relative alla conservazione dei documenti digitali vedi: ALBERTO SALARELLI, ANNA MARIA TAMMARO, *La biblioteca digitale*, Milano, Editrice Bibliografica, 2000, pp. 166 - 172.

una durata di vita decisamente maggiore rispetto ai supporti tradizionali. Ad esempio la Biblioteca Nazionale francese ha iniziato ad adottare per l'archiviazione digitale del proprio patrimonio librario il *Century Disc* un particolare tipo di disco ottico in vetro temperato, rivestito di oro e alluminio e da una lega di nichel e rame, che ha la caratteristica di poter resistere a numerosi possibili disastri ambientali ed a temperature che vanno dai -150° a +350° Celsius, questi dischi garantiscono una durata nel tempo che, come dice il nome, va ben oltre il secolo.

II.7.4. Riproducibilità

La grande riproducibilità degli oggetti digitali può esser vista sia come un limite sia come una opportunità nel caso del libro elettronico. Abbiamo già visto che il principale compito delle tecnologie di DRM è proprio quello di impedire la copia non autorizzata degli e-book, il timore della pirateria ha da una parte frenato i protagonisti del mercato editoriale, dall'altra ha spinto i produttori di software ad integrare nei propri prodotti dei sistemi di protezione, che spesso finiscono per limitare eccessivamente la libertà del lettore onesto e l'ergonomia stessa del libro elettronico, che, come abbiamo visto, in taluni casi non può essere prestato, può essere letto soltanto su determinati supporti, ed ha tutta una serie di altre limitazioni che sicuramente non favoriscono la diffusione di questo nuovo strumento. Fa da contraltare all'insuccesso dell'e-book commerciale il successo delle biblioteche digitali. Se la riproducibilità è un limite per il mercato è invece una grande opportunità per la diffusione della cultura e per quelle istituzioni, le biblioteche, paladine di questa attività. "In generale, per biblioteca digitale si intende una collezione di documenti digitali (sia prodotti mediante digitalizzazione di originali materiali, sia realizzati ex novo) accessibile mediante canali telematici ed eventualmente affiancata da strumenti informatici per la catalogazione dei documenti stessi, e per la ricerca delle informazioni"[99]. Il digitale fornisce nuove importanti possibilità alla diffusione della cultura, un libro digitale non sarà mai

[99] FABIO CIOTTI, GINO RONCAGLIA, *Il mondo digitale. Introduzione ai nuovi media*, Roma-Bari, Editori Laterza, 2001[4], p.379.

indisponibile, sarà sempre possibile fornirne una copia all'utente, in questo caso come si vede la riproducibilità è un vantaggio, inoltre le biblioteche digitali grazie alla telematica sono raggiungibili da qualsiasi luogo in cui sia disponibile una connessione alla Rete. Dove il fine primo non sia il lucro bensì la promozione culturale, il libro elettronico riesce ad esplicitare al meglio tutte le sue potenzialità; oggi sono proprio le biblioteche, nella loro incarnazione digitale, le principali destinatarie ed utilizzatrici dell'e-book e dell'informazione digitale in genere. Le cifre attestano un grande successo di queste iniziative di digitalizzazione, là dove il mercato invece arranca e stenta ad avviarsi. A titolo di esempio riportiamo l'esperienza della E-book Library organizzata dall'Electronic Text Center della University of Virginia[100], questa biblioteca digitale offre al pubblico al momento in cui scriviamo ben 1.800 e-book di opere libere dai diritti d'autore appartenenti ovviamente sopratutto alla cultura anglosassone; i formati utilizzati sono il Microsoft Lit e Aportis Doc per lettori Palm. Dall'agosto duemila quando il sito ha aperto i battenti sino a maggio duemiladue ben ottomilioni e mezzo di e-book sono stati scaricati dal sito con una media (in continuo aumento) di quasi novemila download al giorno. Molte delle biblioteche digitali che in questi anni hanno visto la luce si vanno sempre più orientando verso XML, la nuova lingua franca del Web, come formato di memorizzazione dei propri testi. L'XML permette sia di effettuare delle interrogazioni anche complesse sui testi sia di produrre dei formati di output adatti ad essere fruiti su dispositivi e con software diversi.

II.7.5. Duttilità

Un oggetto digitale è molto duttile, i documenti digitali possono essere facilmente combinati, elaborati per mezzo di software. Nel caso dei libri elettronici il grosso vantaggio del digitale è dato dai software per *l'information retrival*, che nel giro di pochi secondi ci permettono di effettuare complesse interrogazioni anche su

[100] <http://etext.lib.virginia.edu/ebooks/>.

consistenti corpus librari, operazioni che invece richiederebbero molto tempo o sarebbero quasi impraticabili coi sistemi tradizionali. Un testo digitale è versatile e riutilizzabile, può esser per esempio combinato con suoni ed immagini per produrre prodotti multimediali. Essendo un oggetto informatico, il testo digitale, può essere trattato con gli strumenti della tecnologia informatica, sono quindi innumerevoli gli usi cui può essere adattato, dai software di sintesi vocale a quelli per edizioni critiche, dai motori di ricerca alle applicazioni per la lettura di testi paralleli, solo per citarne alcuni.

Il libro elettronico è oggi una tecnologia giovane, sono ancora molti i passi avanti che si devono fare sia dal punto di vista concettuale sia da quello puramente tecnico, tuttavia è stata aperta una nuova strada che potrà fornire valide opportunità alla cultura e quindi alla libertà.

II.8. COME ABBIAMO LAVORATO

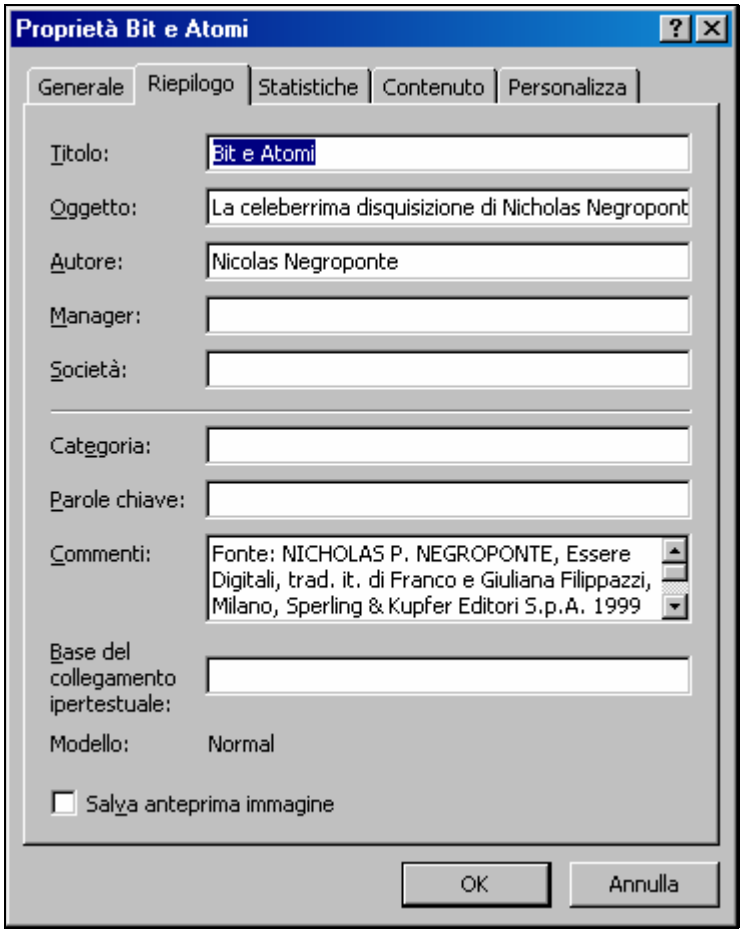
A titolo esemplificativo delle potenzialità del testo digitale desideriamo illustrare il procedimento da noi seguito per l'archiviazione ed il trattamento delle fonti, che abbiamo adoperato per l'elaborazione di questa relazione. Nella celeberrima dicotomia echiana fra "apocalittici" ed "integrati" riteniamo di poter ascrivere la nostra posizione (spero senza integralismi) a quella degli integrati. Possedendo una buona confidenza col computer abbiamo cercato di organizzare il lavoro in modo tale da essere in varie fasi agevolati dal "cervello elettronico"; tuttavia, lo sforzo della valutazione e dell'organizzazione dei materiali oltre che dell'elaborazione e stesura finale della relazione è stato inesorabilmente frutto della fatica del nostro "cervello umano". Ecco in dettaglio come si è proceduto[101].

La prima fase dell'elaborazione di ogni relazione è notoriamente la raccolta e la catalogazione dei materiali. Per la collezione delle fonti ci siamo giovati ampiamente di internet che abbiamo scandagliato in lungo ed in largo con l'ausilio del motore di ricerca Google[102], in assoluto il nostro preferito. Siamo, inoltre, riusciti grazie al prezioso aiuto della Professoressa Lucrezia Lorenzini a costituirci una buona bibliografia di testi "tradizionali".

Per quanto riguarda la catalogazione del materiale in nostro possesso, sebbene, visto l'argomento trattato, una parte delle nostre fonti fosse già in formato digitale (e-book, pagine web), abbiamo provveduto anche per le fonti su supporto cartaceo (libri, riviste, atti) a digitalizzare le parti per noi di interesse. Per far questo ci siamo servito di un vecchio scanner[103] piano formato A4 della Primax e di un ottimo software OCR[104] di produzione russa ABBYY FineReader 6. Abbiamo poi

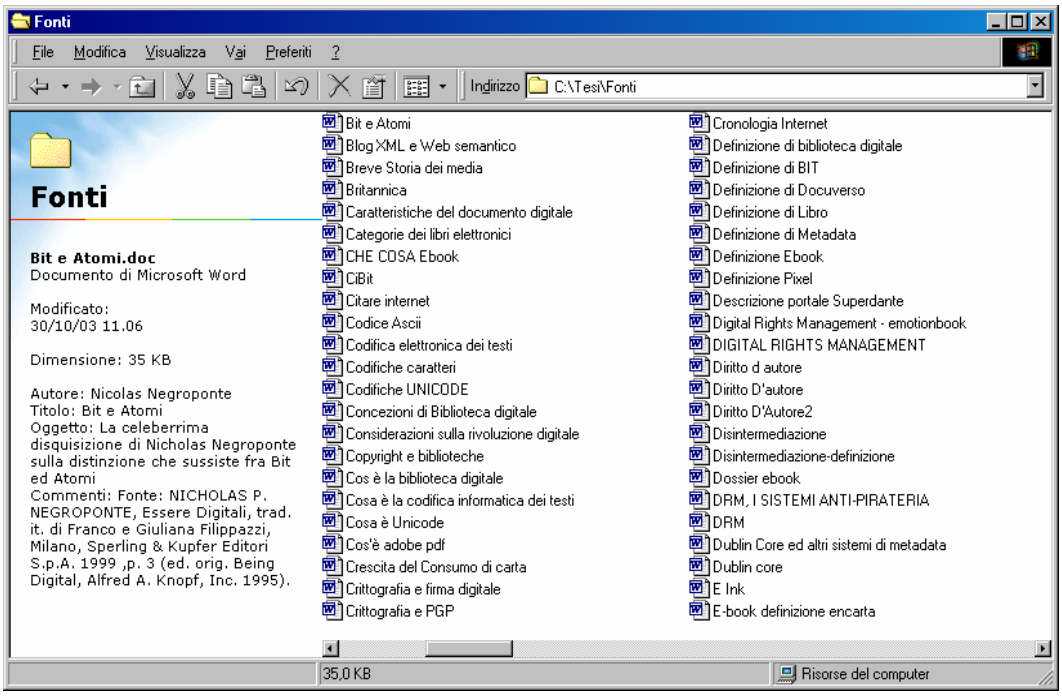
[101] Il sistema di lavoro descritto in queste pagine fa ovviamente riferimento all'elaborazione della parte puramente teorica della relazione che state leggendo.
[102] <www.google.it>.
[103] Lo scanner è una periferica di input per computer che, attraverso un sistema di rilevazione ottica, consente di trasformare documenti su supporto materiale (testo ed immagini stampate, fotografie, diapositive etc.) in immagini digitali.
[104] Col nome di OCR (Optical Character Recognition, traduzione: riconoscimento ottico dei caratteri) si indica una categoria di programmi la cui funzione è quella di trasformare una immagine grafica,

provveduto a trasformare i documenti elettronici acquisiti da internet oppure frutto della digitalizzazione dal cartaceo, in formato Microsoft Word 2002, ciò ci ha permesso di effettuare una schedatura di tutte le nostre fonti, infatti tale formato consente di applicare delle metainformazioni al file. La scheda delle proprietà del file accessibile dal menù file/proprietà del programma si presenta così:



il vantaggio di usare tale funzione consiste soprattutto nel fatto che queste informazioni sono semplicemente accessibili da sistema operativo (nel nostro caso Windows 98 SE). La seguente immagine illustra come si presenta da Risorse del Computer la cartella contenete le nostre fonti:

generalmente acquisita con uno scanner, contenente del testo in un file di testo modificabile col computer.



come si può vedere sulla sinistra appaiono le informazioni riguardanti la fonte, da noi inserite al momento della catalogazione.

Abbiamo optato per un formato di un software di videoscrittura anche perché così abbiamo potuto agevolmente aggiungere evidenziazioni alle parti più significative usufruendo delle funzioni del programma. Per quanto riguarda le fonti in formato e-book (pdf e lit) non abbiamo sentito la necessità di una conversione in formato Word in quanto abbiamo potuto utilizzare i funzionali strumenti messi a disposizione dai rispettivi programmi di lettura degli e-book.

Man mano che la consultazione delle fonti procedeva e cominciavamo ad intravedere connessioni tra le stesse, abbiamo realizzato una scaletta con collegamenti ipertestuali che puntavano ai vari testi o a determinate parti di questi. Sebbene per far ciò fossero numerose le possibilità a nostra disposizione, esistono software ad hoc e lo stesso Word consente di impostare iperlink, abbiamo preferito creare delle pagine Html vista la nostra buona conoscenza del linguaggio.

In ambito informatico circola uno scherzoso adagio che dice che esistono solo due tipi di dati: quelli di cui esiste il backup[105] e quelli che non sono stati ancora persi;

[105] Il termine backup indica la copia di riserva di un programma o di un gruppo di dati realizzata nel caso in cui l'originale venisse danneggiato o distrutto.

rendendoci conto del rischio di poter perdere informazioni per noi molto importanti ci siamo da subito preoccupati di effettuare quotidiani salvataggi dei dati in nostro possesso, ci siamo serviti per questo di due cd riscrivibili, uno da usare i giorni pari uno quelli dispari, e di una porzione di spazio nel nostro secondo hard-disk. Inoltre non volendo rimanere vincolati da un formato proprietario accessibile esclusivamente con un unico programma, abbiamo salvato anche, come copia di backup, tutti i documenti da noi convertiti in formato Word in HTML oppure RTF, in modo da non rimaner preclusi dall'accesso ai nostri dati, magari in un momento importante in cui non avremmo potuto fermarci per cercare di risolvere il problema, da un eventuale danneggiamento di Microsoft Word.

Un importante aiuto ci è venuto dal computer nella fase cruciale della nostra relazione, quella della redazione, infatti per destreggiarci al meglio nel cospicuo numero di fonti a nostra disposizione ci siamo giovati di un software "The Sleuthhound!", un vero e proprio motore di ricerca per documenti digitali. Grazie a questo programma siamo riusciti ad interrogare a fondo ed in breve tempo tutti i materiali a nostra disposizione. Il software supporta gli operatori boleani per affinare la ricerca, ad esempio come abbiamo visto il termine e-book compare nei testi nella doppia dizione con trattino e senza, a noi è bastato indicare come parole chiave della nostra ricerca "ebook OR e-book" per ottenere tutte le pagine in cui questo termine era contenuto indipendentemente dalla grafia. Il programma inoltre già dall'interfaccia di ricerca permette di accedere al contenuto testuale del documento evidenziando le occorrenze trovate nel testo, cosicché si è subito in grado di avere un'idea generale del contenuto della fonte e eventualmente decidere di aprirla se necessario. Il programma supporta i principali formati dei documenti digitali doc, pdf, txt, html solo per citarne alcuni.

Nella stesura di questa relazione ci siamo giovati anche di alcuni strumenti di consultazione in formato elettronico, il Dizionario della lingua italiana De Mauro e l'Enciclopedia Microsoft Encarta, inoltre, dato che numerose nostre fonti erano in inglese, ci siamo avvalsi largamente del programma Babylon, un particolare

software di traduzione grazie al quale basta clickare su di una parola per ottenerne immediatamente la traduzione. Da quanto detto, è possibile evincere chiaramente che già oggi, seppure con strumenti ancora troppo giovani, si possono ottenere, grazie alla tecnologia informatica, importanti vantaggi dall'utilizzo dei testi digitali.

PARTE TERZA

III.1. INTRODUZIONE

In questa terza ed ultima parte illustreremo dettagliatamente il lavoro svolto per la codifica elettronica del romanzo Baltico di Matteo Collura ed esporremo i risultati di quello che potremmo definire un esperimento di codifica. Si è ipotizzato di dover codificare un libro da inserire in un archivio elettronico della Cattedra di Filologia e Letteratura Siciliana ovvero in un'ipotetica realizzanda biblioteca digitale della letteratura siciliana. Alla codifica propriamente detta si sono affiancati tutta una serie di lavori, per cosi dire, collaterali al testo codificato, finalizzati alla fruizione, distribuzione e studio del testo stesso ed eventualmente destinati ad essere integrati nelle strutture dell'archivio digitale. Nel corso della codifica si sono affrontati problemi di varia natura e si sono trovate diverse soluzioni, in alcuni casi originali ed alternative rispetto alle pratiche comuni. Abbiamo cercato di non allontanarci troppo dalle orme di chi questi sentieri ha calcato prima di noi, pertanto ci siamo sforzati di mantenere una certa coerenza con le scelte e le soluzioni adottate nelle poche esperienze italiane analoghe.
Nelle pagine che seguono sono illustrati nel dettaglio le fasi di tutto questo lavoro ed i risultati che riteniamo di aver ottenuto.

III.1.2. Scelte

Una volta identificati gli obiettivi da perseguire, ossia la digitalizzazione e la codifica di un testo unitario con finalità di conservazione, analisi e distribuzione da inserire nell'archivio elettronico della Cattedra di Filologia e Letteratura Siciliana ed eventualmente poi da integrare in una possibile biblioteca digitale, si sono dovuti individuare gli strumenti con cui poter raggiungere gli obiettivi prefissatici.

Abbiamo detto che una delle finalità che abbiamo tenuto a mente nell'elaborazione del nostro lavoro è stata la possibilità di poter integrare l'opera da noi codificata in una biblioteca digitale appare, dunque, opportuno spendere qualche parola sul concetto di biblioteca digitale e quanto ad esso sottende.

III.1.2. Sul concetto di biblioteca digitale

La dissertazione teorica sul tema della biblioteca digitale imperversa in questi anni più accesa che mai ed una elaborazione chiara ed universalmente condivisa del concetto di biblioteca digitale sembra ancora di là da venire. Superficialmente, sulla scorta di quanto già fatto con il termine e-book, possiamo identificare la biblioteca digitale sulla base di una emulazione avanzata in ambito informatico delle strutture, organizzazioni, servizi, funzionalità e finalità della biblioteca tradizionale, tenendo ovviamente presenti le necessarie diversità che la natura elettronica e telematica di questa nuova entità comportano. Non qualsiasi archivio di testi elettronici, quindi, può esser considerato biblioteca digitale, sostiene giustamente Fabio Ciotti: "il contenuto informativo di una biblioteca si distingue da un generico insieme di documenti in quanto dotato di un'*organizzazione complessiva* dovuta ad un agente intenzionale distinto dai creatori dei singoli documenti in essa contenuti. Tale organizzazione si manifesta nella biblioteca tradizionale mediante la classificazione, la soggettazione e l'indicizzazione. Questi strumenti, infatti, costruiscono una rete virtuale di relazioni tematiche e concettuali tra i documenti presi come unità"[106]. Una delle principali funzioni delle biblioteche tradizionali è infatti la catalogazione ed organizzazione del corpus documentale, un tale tipo di struttura deve poter essere ravvisato anche nella sua versione digitale, come vedremo meglio in seguito, la possibilità di definire un'adeguata rete di relazioni interdocumentali ed intradocumentali è legato anche alla potenza rappresentazionale del linguaggio adoperato per la codifica dei

[106] Cfr. FABIO CIOTTI, *TEI in digital library: an italian case*, intervento al *TEI Members Meeting*, Pisa 17-18 Nov 2001, in TEI Consortium Website
<http://www.tei-c.org.uk/Members/2001-Pisa/Talks/ciotti.htm>
(10 febbraio 2004).

documenti e del sistema di metadati ad esso associato. Ma, la biblioteca sia quella tradizionale, quanto quella digitale, non esaurisce le sue funzioni esclusivamente nell'organizzazione e catalogazione dei testi, sebbene sia questa un'attività di grande rilievo, bensì si esplica in tutta una serie di servizi e finalità di notevole importanza. Sottolinea questo fatto la definizione che di biblioteca digitale offre Anna Maria Tammaro nel suo libro, scritto insieme ad Alberto Salarelli, dal titolo "La biblioteca digitale"; la studiosa propone come una delle migliori sin ora elaborate quella usata al *Workshop on distribued knowledge work environments* svoltosi dal 9 all'11 marzo 1997 a Santa Fe, in New Mexico: "II concetto di biblioteca digitale non è quello di una collezione digitale dotata di strumenti di gestione dell'informazione. È piuttosto uno spazio in cui mettere insieme collezione, servizi e persone a supporto dell'intero ciclo di vita della creazione, uso, preservazione di dati, informazione e conoscenza."[107]

Come si evince il problema è abbastanza complesso e la discussione teorica è ad oggi ancora assai viva e controversa. A noi basti dire che per il lavoro di digitalizzazione e codifica del testo di Baltico abbiamo sempre cercato, nell'orientare le nostre scelte, di aver presente le eventuali esigenze di una biblioteca digitale, ed anzi ci siamo sforzati di fare in modo che, seppure per una prima fase della vita del documento digitale si prevede che questo vada a far parte esclusivamente di un archivio testuale, si possano offrire, in piccolo, anche con gli esigui strumenti del dipartimento alcuni servizi avanzati tipicamente prerogativa delle biblioteche digitali.

In quest'ottica la prima decisione veramente importante ha riguardato la scelta della soluzione da adottare per la memorizzazione digitale del testo. Il sistema di codifica del testo per adattarsi al meglio alle nostre esigenze ed agli obiettivi prefissatici avrebbe dovuto avere delle ben precise caratteristiche: essere in primo luogo dotato di una adeguata potenza rappresentazionale, tale da poter raffigurare

[107] Cfr. ALBERTO SALARELLI, ANNA MARIA TAMMARO, *La biblioteca digitale*, Milano, Editrice Bibliografica, 2000, p. 107.

adeguatamente i diversi fenomeni testuali, ed allo stesso tempo essere espandibile ed integrabile a seconda di specifiche esigenze di analisi o riproduzione. Il nostro linguaggio di codifica avrebbe dovuto dare inequivocabili garanzie di portabilità e riusabilità del prodotto digitale nello spazio e nel tempo; doveva, quindi, far necessariamente riferimento ad uno standard universalmente riconosciuto, adottabile su qualsiasi piattaforma informatica e non soggetto a rapida obsolescenza. All'indipendenza da una particolare architettura hardware o software si affianca l'indipendenza logica da tipologie di elaborazione, il nostro sistema di rappresentazione oltre ad essere portatile e standard non avrebbe dovuto essere orientato ad un'applicazione specifica, quali stampa, *information retrival o altro*, piuttosto si doveva adattare ed essere utilizzabile a seconda delle diverse esigenze. Anche nell'ottica della biblioteca digitale riveste notevole importanza la capacità dello standard di codifica di farsi portatore di un alto grado di meta-informatività, capace di rappresentare adeguatamente le relazioni interdocumentali ed intradocumentali; i metadati sono "bit che parlano di altri bit", ossia insieme di dati ed informazioni che descrivono un oggetto digitale, e costituiscono l'equivalente elettronico della scheda per la catalogazione e soggettazione dei documenti nella biblioteca tradizionale, ma a differenza delle schede catalografiche i metadati, come meglio vedremo nelle prossime pagine, veicolano informazioni che vanno al di là della semplice individuazione ed indicizzazione del documento ma ineriscono alla stessa natura dell'oggetto digitale e si ricollegano a problematiche quali la preservazione nel tempo e l'integrazione in reti telematiche.

Alla luce di quanto detto ed in virtù di quanto già evidenziato nei capitoli precedenti, la nostra scelta sul linguaggio da adottare per la codifica di Baltico non poteva che cadere su quello che ormai è univocamente indicato come lo standard per eccellenza nell'interscambio di documenti, l'XML giustamente designato come l'ASCII del ventunesimo secolo. L'adozione di un linguaggio di codifica di tipo dichiarativo ci ha peraltro permesso quella indipendenza da tipologie di

elaborazione, di cui abbiamo detto in precedenza; come dimostreremo concretamente nel resto della trattazione, partendo da un unico sorgente XML siamo stati in grado di ottenere output per diverse esigenze di fruizione o analisi. Appurata l'adozione di XML, rimaneva da scegliere la DTD, ossia il vocabolario XML da utilizzare, in tal senso la scelta è stata pressoché obbligata, non potendosi non orientare verso quello che in ambito umanistico è ormai considerato lo standard per la rappresentazione informatica dei testi letterari, ovvero lo schema di codifica definito dalla *Text Encoding Initiative.*

III.2. TEI

La *Text Encoding Initiative* è un progetto di ricerca internazionale, volto allo sviluppo di un modello di codifica standard finalizzato alla rappresentazione dell'informazione testuale ed alla gestione dei dati umanistico-letterari in formato elettronico. Il progetto Tei ha preso l'avvio nel 1988[108] ed è promosso, organizzato e sostenuto dalle tre più importanti associazioni nel campo dell'informatica umanistica e della linguistica computazionale: l'Association for Computers and the Humanities (ACH)[109], l'Association for Computational Linguistics (ACL)[110], e l'Association for Literary and Linguistic Computing (ALLC)[111]; il principale supporto finanziario al progetto è fornito dall'U.S. National Endowment for the Humanities (NEH)[112], dal XIII Direttorato della Commissione Europea (Information Society Directorate-General CEC/DG-XIII)[113], dalla Andrew W. Mellon Foundation[114] e dal Social Science and Humanities Research Council of Canada[115].

Sin dal suo avvio, obiettivo principale della *Text Encoding Initiative* è stato la definizione di uno standard libero per la memorizzazione dei testi letterari finalizzato all'interscambio di documenti di natura umanistico-letteraria tra piattaforme informatiche differenti, non dipendente da una particolare applicazione software ed adatto alla rappresentazione di tutte le categorie di testi e caratteristiche testuali oggetto di studio.

Per il conseguimento di questi obiettivi l'orientamento è teso verso una codifica di tipo dichiarativo; si adottò lo *Standard Generalized Markup Language* (SGML), per la definizione del linguaggio di *markup* dei documenti. Le prime specifiche

[108] L'idea che spinse alla realizzazione del progetto TEI vide la luce durante un convegno tenutosi nel 1987 al "Vassar College" di Poughkeepsie a New York. Nel corso del convegno furono redatti i cosiddetti "Poughkeepsie Principles", ossia i principi base cui ci si sarebbe dovuti attenere nella ralizzazione di uno schema di codifica dei testi letterari per la ricerca umanistica.
[109] <http://www.ach.org/>.
[110]<http://www.cs.columbia.edu/~acl/home.html>.
[111] <http://www.kcl.ac.uk/humanities/cch/allc/>.
[112] <http://www.neh.fed.us/>.
[113] <http://europa.eu.int/comm/dgs/information_society/index_en.htm>.
[114] <http://www.mellon.org/>.
[115] <http://www.sshrc.ca/>.

provvisorie sono state pubblicate nel 1991 con il titolo *Guidelines for Electronic Text Encoding and Interchange, TEI P1*. Successive revisioni del linguaggio si sono avute nel 1992, *TEI P2*, nel quale la struttura della DTD è stata notevolmente rimaneggiata e nel 1994, *TEI P3*. Sull'onda del successo ottenuto dal linguaggio TEI, adottato in numerosi progetti accademici a livello internazionale, nel dicembre del 2000 i membri della *Text Encoding Initiative* hanno deciso di rendere permanente il progetto, costituendo il *TEI Consortium*[116], una organizzazione internazionale senza scopo di lucro fondata al fine di sostenere e mantenere lo sviluppo e la diffusione dello standard TEI. Il TEI Consortium ha i suoi uffici esecutivi a Bergen in Norvegia ed è ospitato da quattro università nel mondo: University of Bergen, Brown University, Oxford University, e University of Virginia. Il Consorzio è diretto da un Consiglio di Direzione e la sua attività tecnico-scientifica è controllata da un Consiglio elettivo. Le attività sono svolte tipicamente da piccoli gruppi internazionali di esperti, coordinati da due curatori, uno in Nord America e uno in Europa. Nel giugno del 2002 il TEI Consortium ha provveduto alla pubblicazione di una nuova versione dello schema di codifica, *TEI P4,* contraddistinto dalla piena conformità con *XML*, fattore questo che ha prepotentemente proiettato la TEI, sulla scorta del notevole successo di XML e delle tecnologie ad esso correlate, verso nuovi importanti sviluppi nell'ambito delle biblioteche digitali e del World Wide Web stesso.

III.2.1. Lo schema di codifica TEI

TEI è fondamentalmente un vocabolario XML (originariamente un'applicazione SGML) definito per mezzo di una DTD; le indicazioni sull'uso degli elementi definiti nella DTD sono contenute in documenti ufficiali, detti *Guidelines,* reperibili liberamente sul sito del Consorzio TEI[117]. I principi base che hanno ispirato gli estensori del linguaggio sono quelli di una codifica di tipo dichiarativo mirante alla

[116] <http://www.tei-c.org>.
[117] L'indirizzo da cui è possibile prelevare o consultare una copia delle direttive per la codifica è: <http://www.tei-c.org/P4X/>. Ne esiste anche un'edizione a stampa, per i tipi della University of Virginia Press (<http://www.upress.virginia.edu/books/tei.html>).

descrizione di strutture logico-funzionali astratte del documento piuttosto che del suo aspetto fisico, in accordo in ciò con i fondamenti teorici dell'SGML, prima, e della sua recente evoluzione, l'XML, poi. La DTD TEI ripropone una sostanziale traduzione-trasposizione dei formalismi strutturali e funzionali convenzionalmente vigenti nei testi letterari nell'ambito dell'organizzazione strutturale degli elementi del linguaggio di marcatura.

Nota giustamente Fabio Ciotti: "Il modello descrittivo dei testi che sottende la TEI è basato su una formalizzazione delle convenzioni nella produzione di documenti testuali che sono state definite a partire dalla diffusione della stampa, e che sono state codificate fino al punto di divenire un vero e proprio schema di argomentazione (la divisone in capitoli, parti, paragrafi, etc., è un tipico esempio di questo fenomeno di determinazione dialettica tra modelli del pensiero e schemi imposti dagli strumenti di produzione intellettuale)"[118].

L'adozione di uno schema di codifica di tipo dichiarativo focalizzato sull'identificazione dei rapporti strutturali e funzionali presenti nel testo comporta, torno a citare Ciotti: "un apporto assolutamente soggettivo dello studioso-codificatore, che necessita di interpretare la funzione delle varie componenti strutturali tipografiche, o manoscritte, per essere in grado di impiegare in modo adeguato i marcatori per gli elementi previsti nella DTD della TEI"[119].

Si può ben capire, quindi, la necessità di una preparazione di tipo filologico umanistico del codificatore; la TEI è un linguaggio informatico di codifica dei testi letterari fatto da umanisti per umanisti, che comunque, come vedremo nel prosieguo della trattazione, può estrinsecare notevoli potenzialità anche in ambiti applicativi non strettamente "umanistici".

---

[118] FABIO CIOTTI, *Breve introduzione alla Text Encoding Initiative*, in *Biblioteca Italiana* <http://www.bibliotecaitaliana.it/tei_intro.asp> 1 dicembre 2003, (1 febbraio 2004).

[119] Vedi nota precedente.

Sono oltre 500 gli elementi definiti dalla DTD TEI tanto che molte caratteristiche strutturali di un testo possono essere agevolmente sottoposte a differenti tipi di codifica. Lo schema TEI è caratterizzato da una forte modularità e prevede ampie possibilità di personalizzazione e di estensioni al fine di adattarsi a ogni esigenza di codifica testuale. Nella pratica la DTD è costituita da una serie di, per così dire, sotto DTD, dette *DTD Fragments*, che possono essere modulate e combinate fra di loro per ottenere uno schema di codifica che si attagli al meglio alle caratteristiche del testo da codificare. I DTD Fragments definiscono differenti gruppi di tag, detti *Tag sets,* che vengono combinati fra di loro per produrre uno schema adatto alla tipologia del testo da sottoporre a codifica.

Le Guidelines identificano tre gruppi di *Tag sets* :

- Core *Tag set*: contiene il nucleo minimo di elementi comuni ad ogni tipo di testo, questo Tag set deve essere necessariamente incluso in ogni DTD.
- Base *Tag set:* contiene diversi *Tag set* specifici per vari tipi di documenti fondamentali: prosa, poesia, dramma, parlato, lessicografico, terminologico. Questi tag set vanno inclusi nella DTD a seconda della tipologia dell'opera da codificare: lirica, prosa, dramma etc..
- Additional DTD *Tag set:* insiemi di elementi per la rappresentazione di caratteristiche utili ad evidenziare determinate peculiarità del testo a partire da particolari prospettive analitiche o applicazioni specializzate quali: codifica di apparati di varianti, rappresentazione di strutture ipertestuali, codifica di fonti primarie ed altro.

TEI consente agli sviluppatori di personalizzare, ove se ne presenti la necessità, le DTD standard, inserendo nuovi tag ed entità ovvero modificando quelli esistenti grazie al cosiddetto *auxiliary tag set.* Al fine di aiutare gli utenti nella costruzione di versioni personalizzate della DTD nel sito del TEI Consortium si trova una

applicazione Web, chiamata scherzosamente *Pizza Chef*[120] con cui è possibile senza grande sforzo "cucinarsi", per adottare l'ironica definizione usata nel sito dell'applicazione, la propria DTD.

Con l'obiettivo di favorire la diffusione della TEI la commissione editoriale del progetto ha prodotto una versione ridotta dello schema di codifica battezzandola *TEI-Lite.*

TEI-Lite è una DTD, che contiene soltanto un sottoinsieme degli elementi (circa 170) dell'intero sistema TEI, pur mantenendo la piena compatibilità e conformità con esso.

TEI-Lite è praticamente una versione semplificata dell'intero schema di codifica definito dalla TEI, creato per soddisfare le esigenze più comuni nella codifica dei testi, "per rispondere al 90% delle esigenze del 90% della comunità di utenti della TEI"[121] e facilitare la realizzazione di testi in formato elettronico compatibili con l'intero schema, senza richiedere lo studio di tutta la DTD. Al momento TEI-Lite è il sott'insieme di TEI più diffuso, è adottato in numerosi progetti di codifica testuale e nella creazione di archivi documentali. Esiste un manuale d'uso chiaro ed agevole per lo studio della TEI-Lite disponibile anche nella traduzione italiana curata da Fabio Ciotti[122].

Per il nostro progetto si è deciso di adottare proprio la DTD TEI-Lite in quanto rispondente alle nostre necessità, essendo il nostro esperimento di codifica rivolto principalmente a quelle che poterebbero essere le esigenze di costituzione e gestione di un corpus testuale. Tuttavia, in virtù della natura incrementale del markup TEI/XML anche questo primo stadio base di rappresentazione del testo costituisce un primo passo per la realizzazione di codifiche ulteriori orientate a finalità di ricerca specifiche.

---

[120] <http://www.tei-c.org/pizza.html>.

[121] Vedi nota 118.

[122] BURNARD LOU, SPERBERG-MCQUEEN C. M., *TEI Lite: introduzione alla codifica dei testi*, trad. it. Fabio Ciotti, Guendalina Demontis, Giuseppe Gigliozzi, Massimo Guerrieri, Andrea Loreti in *TEI Website* <http://www.tei-c.org/Lite/teiu5_it.htm>
(ed. orig. *TEI U5: Encoding for Interchange: an introduction to the TEI*, <http://www.tei-c.org/Lite/teiu5_en.tei>)
gennaio 1998.

III. 3. ESPERIENZE DI CODIFICA IN ITALIA

Nelle pratiche di codifica si è cercato di rimanere coerenti con le altre esperienze italiane analoghe, in particolar modo con i testi del progetto TIL e del portale WEB *GriseldaOnLine*.

III.3.1. TIL

Testi Italiani in Linea (TIL)[123] è una biblioteca digitale sperimentale realizzata nell'ambito di un progetto di ricerca, iniziato nel 1998 e conclusosi nel 2000, coordinato dal CRILet[124] e finanziato dal Ministero dell'Istruzione, dell'Università e della Ricerca Scientifica (MURST). La biblioteca digitale di TIL contiene una collezione di opere della tradizione letteraria italiana dalle origini all'epoca contemporanea. I testi sono codificati in formato SGML o XML in base allo schema di codifica TEI. La consultazione on-line delle opere si basa sul software di *document management*, operante lato server, Dynaweb il quale produce dinamicamente una versione HTML dei documenti XML e un indice dei contenuti per ciascun testo, consente di effettuare ricerche full-text e contestuali e di creare concordanze dinamiche con collegamenti ipertestuali al testo. Oltre al *corpus* della biblioteca sul sito del progetto è stata pubblicata una ricca documentazione relativa al progetto TIL, fra cui spicca un buon manuale per la codifica dei testi realizzato da Fabio Ciotti. A differenza di quanto da noi fatto, i testi TIL adottano una versione personalizzata della intera DTD TEI; ove possibile, abbiamo cercato soprattutto in riferimento ad alcune impostazioni riguardanti i nomi di attributo, dove maggior libertà è lasciata dalla DTD al codificatore, di conformarci alle scelte fatte dagli studiosi romani.

[123] <http://til.scu.uniroma1.it/>.
[124] Centro ricerche Informatica e Letteratura, Dipartimento di studi Filologici Linguistici e Letterari di Roma “La Sapienza”, <http://crilet.scu.uniroma1.it/>.

III.3.2. GriseldaOnLine

Analogamente a quanto avvenuto con TIL nella nostra codifica abbiamo anche tenuto conto, nell'orientare le nostre scelte, del lavoro svolto per la sezione di informatica umanistica del sito *GriseldaOnLine*[125].

Frutto della collaborazione tra il Dipartimento di Italianistica dell'Università di Bologna e la casa editrice *Gedit*, *GriseldaOnLine* è una rivista sperimentale di letteratura dedicata alla scuola, alla formazione didattica e ai modelli informatici applicati alle scienze umane. Il sito dedica all'informatica umanistica una intera sezione del portale; qui si trovano un manuale di informatica umanistica ed un piccolo archivio digitale di classici della letteratura italiana codificati in formato XML conformi alla DTD Tei-Lite, i quali abbiamo tenuto in considerazione durante il nostro lavoro, al fine di mantenere una certa uniformità nelle pratiche di codifica, almeno, con le iniziative sorte in ambito nazionale.

Nel prosieguo della trattazione, allorché entreremo nello specifico della discussione sui criteri di codifica, illustreremo nel dettaglio dove ci siamo rifatti ai due progetti e dove invece abbiamo preferito discostarcene.

III.3.3. CIBIT

Sebbene non abbia per noi direttamente costituito un modello cui fare riferimento, a causa della lunga indisponibilità dei testi del progetto proprio nel periodo in cui ci dedicavamo alla codifica, merita in questa sede di essere menzionato il progetto CIBIT (Centro Interuniversitario Biblioteca Italiana Telematica), una tra le prime iniziative del mondo accademico italiano di digitalizzazione di testi letterari finalizzata alla realizzazione di una biblioteca digitale.

Nato da un accordo tra undici Università[126] il CIBIT, leggiamo nella pagina di presentazione del progetto, "ha lo scopo di promuovere la collaborazione scientifica e la costituzione e il potenziamento di servizi telematici in comune nel

[125] <http://www.griseldaonline.it/>.
[126] L'Aquila, Cassino, Ferrara, Genova, Messina, Napoli, Padova, Pavia, Pisa, Roma "La Sapienza", Torino, Trento, Udine e Venezia.

campo della documentazione primaria e secondaria relativa al patrimonio testuale (linguistico, letterario, storico, filosofico, scientifico, religioso, politico, giuridico, economico, artistico, musicale, ecc.) della tradizione culturale italiana". Il CIBIT ha concretizzato questi obiettivi soprattutto con la costituzione della "Biblioteca italiana telematica", una biblioteca digitale di circa 1.500 testi della tradizione culturale italiana dal Medioevo al Novecento. In origine la biblioteca era basata su DBT, un software di analisi testuale sviluppato presso l'Istituto di Linguistica computazionale del CNR di Pisa, il quale era legato a un formato di codifica dei documenti proprietario. Ormai da qualche anno l'accesso pubblico ai testi del CIBIT è sospeso in quanto anche la Biblioteca Italiana Telematica ha deciso di orientarsi verso lo standard XML: si sta, quindi, a quanto si legge sulle pagine del sito, curando una conversione del corpus testuale nel nuovo formato. Intanto negli ultimi mesi, promosso dal CIBIT, ha visto la luce un progetto nato dalla fusione delle esperienze di TIL e di Biblioteca Italiana Telematica. Nel nuovo progetto, dal nome di *Biblioteca Italiana*[127], confluiranno i patrimoni testuali delle due iniziative, l'archivio sarà costituito da testi in formato XML/TEI; per la fruizione dei testi si farà riferimento al già citato, software Dynaweb, che permetterà sia la lettura, sia l'effettuazione di complesse ricerche testuali on-line; inoltre, i testi liberi da diritto di autore potranno essere scaricati dall'utente sul proprio computer in diversi formati tra i quali: Adobe PDF, Microsoft Reader e OeBPS (Open eBook Publication Structure): ciò al fine di garantire la fruibilità dei testi sulle più disparate piattaforme informatiche (computer, palmari, etc.), ed usufruire dei vantaggi tipici di questi formati, quali una buona ergonomia di lettura ovvero la produzione di output cartacei di qualità (PDF). Condividiamo ampiamente questa scelta relativamente alla distribuzione dei testi, come vedremo in seguito, anche nel nostro lavoro, abbiamo cercato, seppur in piccolo con le nostre esigue risorse, di seguire un'impostazione analoga.

[127] <www.bibliotecaitaliana.it>.

III.4. DIGITALIZZAZIONE

La prima fase della codifica elettronica dei testi consiste nella digitalizzazione delle fonti. Con la digitalizzazione si traspone il testo affidato al supporto materiale dal mondo degli atomi a quello dei bit. Nel caso che del testo da codificare esista una edizione a stampa moderna il codificatore è agevolato nella sua attività di copista digitale da strumenti quali lo scanner[128] ed i software di riconoscimento ottico dei caratteri (OCR)[129]. Dovendo invece trattare manoscritti o antichi testi a stampa, il codificatore, novello amanuense elettronico, deve necessariamente trascrivere manualmente il testo al computer. Per la prima digitalizzazione di Baltico, trattandosi di una edizione moderna ci siamo potuti giovare degli strumenti poco sopra detti.

Nel dettaglio abbiamo proceduto così: dapprima abbiamo acquisito con lo scanner tutte le circa duecento pagine del testo, le scansioni sono state effettuate alla risoluzione di trecento Dpi[130] in scala di grigi, i file di immagine delle pagine risultanti dalle scansioni sono stati salvati sul computer nel formato TIFF[131]. La scelta di una risoluzione di trecento punti per pollice in fase di acquisizione è stata fatta in quanto tale risoluzione rappresenta il minimo per ottenere discreti risultati con L'OCR, probabilmente se avessimo avuto a disposizione una periferica più veloce del nostro vecchio scanner piano con interfaccia parallela, come ad esempio i nuovi dispositivi con interfaccia Usb, ci saremmo orientati per una risoluzione intorno ai seicento punti per pollice. Abbiamo acquisito in scala di grigi in quanto, per esperienza personale, abbiamo potuto constatare che i software di riconoscimento ottico dei caratteri danno migliori risultati con questa profondità di

[128] Vedi nota n°103.
[129] Vedi nota n°104.
[130] Il dpi, dall'inglese dots per inch, è un'unità di misura della risoluzione che indica il numero di punti per pollice da cui è composta un'immagine.
[131] Il TIFF (Tagged-Image File Format) è un formato di file molto utilizzato per la memorizzazione di immagini di tipo bitmap, tale formato utilizza un sistema basato su tag per la memorizzazione delle caratteristiche dell'immagine, supporta qualsiasi profondità cromatica ed è utilizzato soprattutto per lo scambio di file grafici fra piattaforme ed applicazioni diverse. Il TIFF adotta un metodo di compressione di tipo lossless ossia senza perdita di dati.

colore rispetto al bianco e nero; nel nostro caso non avrebbe avuto senso acquisire a colori in quanto le scansioni da noi effettuate erano finalizzate esclusivamente alle attività di OCR e non all'attestazione delle condizioni e delle caratteristiche della fonte materiale, lavorando infatti su di un testo moderno tali elementi sono irrilevanti.

Una volta terminata la scansione delle pagine abbiamo sottoposto i file grafici ottenuti al software di riconoscimento ottico dei caratteri, nel nostro caso Omnipage Pro 11.0 della Scansoft; per lungo tempo il grande problema dei programmi di OCR è stato la scarsa accuratezza nel riconoscimento del testo ed in particolar modo delle lettere accentate e dei segni di interpunzione, ormai da qualche anno però il livello di precisione di questi strumenti è diventato decisamente elevato ed infatti il software da noi adoperato, peraltro uno dei migliori della categoria, soltanto in pochi casi non ha riconosciuto efficacemente alcuni caratteri.

Dopo aver effettuato l'OCR delle pagine ed avere corretto gli eventuali errori compiuti dal software abbiamo salvato i file ottenuti in formato puro testo (.txt), rimuovendo così tutte le informazioni di formattazione, non necessarie; inoltre, tutti i trattini di rimando a capo sono stati eliminati, ricongiungendo di conseguenza tutte le parole sillabate a fine riga[132].

Una volta compiute queste operazioni preliminari si è passati alla fase successiva del nostro lavoro: la codifica XML/TEI del testo.

[132] Tale operazione è stata eseguita automaticamente grazie all'ausilio del programma OCR.

III.5. CODIFICA

Dopo la digitalizzazione della fonte cartacea ha potuto prendere il via la codifica del testo.

III.5.1. I livelli di codifica

La prima decisione da prendere ha riguardato cosa codificare; sulla scia di quanto già fatto sia nel progetto TIL sia da *GriseldaOnLine*, il processo di trascrizione e codifica si è limitato esclusivamente al contenuto testuale della fonte, trascurandone l'aspetto materiale, inoltre tutti i materiali paratestuali non d'autore o non relativi al testo in sé (indice dei contenuti, note, etc.) sono stati tralasciati, gli unici fenomeni materiali della fonte mantenuto nella codifica sono stati i salti pagina occorrenti nell'edizione fonte.

A tal proposito nell'ambito del progetto TIL è stato elaborato un criterio di classificazione delle fonti e dei livelli di codifica, cui queste possono essere sottoposte, in seguito adottato anche dal progetto Biblioteca Italiana (BibIt).

Il comitato editoriale di TIL ha originariamente deciso di suddividere tutte le possibili fonti di un'opera in due classi:

1. fonti primarie:
    - manoscritti
    - incunaboli
    - edizioni a stampa antiche
    - edizioni a stampa notevoli (es. prime edizioni, etc.)
    - edizioni diplomatiche a stampa
    - edizioni a stampa anastatiche
1. fonti secondarie

- edizioni a stampa moderne
- edizioni critiche moderne

Per le fonti primarie sono considerati rilevanti sia il contenuto testuale, sia i fenomeni materiali riscontrati sul documento originale, in tal caso inoltre si è deciso di affiancare alla trascrizione della fonte anche una digitalizzazione della stessa in formato grafico; la trascrizione e codifica di questo tipo di fonti deve pertanto approssimare quanto più possibile il livello di edizione diplomatico-interpretativa.

Per le fonti secondarie invece il processo di trascrizione e codifica si limita esclusivamente al contenuto testuale, trascurando l'aspetto materiale della fonte. Nella pratica questo comporta che tutti i materiali paratestuali non riconducibili direttamente all'autore o comunque estranei al testo in sé, per come questo è attestato nella tradizione (introduzione, prefazione, indice dei contenuti, note, etc.), presenti sull'edizione fonte sono tralasciati; inoltre ogni fenomeno materiale occorrente sulle pagine viene omesso, eventuali fenomeni di evidenziazione vengono codificati in modo funzionale; l'unico aspetto materiale della fonte che va mantenuto sono i salti pagina presenti nell'edizione fonte, in quanto possono essere utili ai fini di riferimento e citazione del testo.

In relazione alla natura incrementale del vasto markup TEI, che permette diversi tipi di codifica sulla base di esigenze interpretative e funzionali diverse, ed in riferimento alla segmentazione delle fonti gli studiosi di TIL hanno individuato cinque livelli di codifica, cui ciascun testo può esser sottoposto:

- livello 1: codifica della struttura editoriale del testo, di un limitato gruppo di fenomeni editoriali intralineari e linguistici;
- livello 2: codifica di un insieme di caratteristiche strutturali intralineari e linguistiche, codifica di semplici caratteristiche filologiche, eventuale introduzione di riferimenti incrociati e collegamenti ipertestuali; codifica avanzata dei metadati;

- livello 3: codifica di fenomeni testuali complessi in vista di applicazione di analisi avanzate (struttura semantica, narrativa, retorica, morfosintattica, etc.);
- livello 4: trascrizione diplomatica di una fonte primaria;
- livello 5: edizione critica di un opera.

I livelli 3, 4 e 5 non vanno necessariamente considerati come successivi su scala temporale o di complessità, ma piuttosto come livelli paralleli di articolazione del processo di codifica. È inoltre evidente che i livelli 4 e 5 sono applicabili esclusivamente a una o più fonti primarie.

Nella scala di classificazione elaborata dagli studiosi romani, il nostro lavoro può sostanzialmente essere ricondotto al primo livello di codifica con qualche "sconfinamento" nel secondo livello.

III.5.2. Il set di caratteri

Preliminare alla codifica del testo è stata la scelta del set di caratteri da utilizzare per la memorizzazione del contenuto; infatti, è bene ricordare che i file XML sono praticamente dei file di testo. Di default XML adotta il *charset* UTF-8 che, come sappiamo nelle prime 256 posizioni corrisponde perfettamente con il code set ISO 8859-1 o ISO Latin 1; il testo di Baltico generalmente non fa uso di caratteri particolari che avrebbero potuto creare problemi in fase di validazione[133] o trasferimento del documento; si sarebbe potuto quindi ignorare il problema dei caratteri, d'altra parte sappiamo che soltanto il set ISO 646 IRV (coincidente con il vecchio set ASCII a sette bit) è interpretato correttamente in qualsiasi piattaforma informatica e da qualunque applicazione; non a caso TEI prescrive l'adozione di questo *charset*[134] per i testi destinati all'interscambio, mentre per tutti i caratteri

[133] La validazione è il processo mediante il quale si verifica la conformità sintattica della codifica del documento XML con i vincoli imposti nella DTD. Oltre a controllare il corretto utilizzo degli elementi il parser (il software che esegue la validazione) verifica che nell'istanza XML non siano presenti caratteri non supportati dal set dichiarato in apertura di documento.

[134] Le prime 128 posizioni di UTF come di buona parte delle tavole caratteri esistenti corrispondono con i caratteri definiti nella tavola ISO 646 IRV.

non inclusi in questo *code set* stabilisce l'uso delle entità carattere definite negli insiemi pubblici rilasciati dalla ISO, tali entità incluse nella DTD TEI sono, ad esempio, utilizzate anche nel linguaggio HTML.

Abbiamo ritenuto lungimirante, al fine di favorire la più ampia interoperabilità possibile del documento codificato, pur adottando UTF-8, adoperare delle entità per tutti quei caratteri non compresi nelle prime 128 posizioni di questo *charset,* ossia per i caratteri eccedenti l'ISO 646 IRV, ovvero tutte le lettere accentate ed alcuni simboli grafici quali le virgolette. Tuttavia a differenza di quanto prescritto nelle *guidelines* della *Text Encoding Initiative* invece di adoperare le entità di carattere definite da ISO, come ad esempio *"è"*, abbiamo preferito adottare le rispettive entità numeriche, come ad esempio *"è"*, le quali sono supportate nativamente da XML; ciò è stato fatto principalmente per due motivi: in primo luogo per svincolarsi dalla necessità di associare sempre il documento XML alla relativa DTD, infatti le entità di carattere per poter essere utilizzate devono essere specificate nella DTD, utilizzare entità numeriche invece non impone questo vincolo. Personalmente riteniamo che in numerose circostanze possa rivelarsi vantaggioso il distribuire una istanza XML soltanto come documento *ben formato,* una volta verificatane la validità sintattica, infatti soprattutto per documenti di grosse dimensioni, quale appunto un'opera letteraria, i tempi di validazione possono allungarsi andando ad aggiungersi ai tempi di elaborazione di altre attività sul documento XML. Si pensi ad esempio alla trasformazione dinamica in HTML di un file XML per mezzo di XSLT, al tempo di elaborazione del foglio di stile si aggiunge quello per la validazione sintattica del documento, attività questa, a nostro modesto parere, pleonastica se ci si è già accertati in partenza della validità del documento. Il secondo motivo che ci ha indotto a servirci di entità numeriche, è stato l'aver constatato che in numerose circostanze si rivelano maggiormente "gradite" nei processi di elaborazione, soprattutto con alcuni software per il trattamento dei materiali XML.

Per la sostituzione di tutte le lettere accentate ed altri caratteri non inclusi nella tavola ASCII a sette bit, ci siamo serviti del software Search and Replace prodotto dalla Funduc Software Inc., il quale, con un processo automatico, si è occupato di rimpiazzare tutti i caratteri in questione con le rispettive entità numeriche.

III.5.3. Gli strumenti per la codifica

Prima di partire con la codifica effettiva del testo è stato indispensabile selezionare ed approntare gli strumenti per codificare "materialmente" il testo, ossia per applicare i tag ai file provenienti dalla digitalizzazione della fonte cartacea.

Dopo un'accurata valutazione la nostra scelta è caduta su due programmi[135] Altova XmlSpy 5[136] e NoteTab 4.9[137].

XmlSpy 5 è uno dei più completi e versatili editor XML al momento in commercio; tale software è stato adoperato principalmente per la revisione del testo codificato, la validazione sintattica dei file XML e, nella fase più avanzata del lavoro, per la preparazione dei fogli di stile XSLT; a causa degli alti costi di licenza del programma in questione ci siamo serviti della versione "trial"[138] del software esclusivamente per il periodo di prova di trenta giorni.

NoteTab è invece un editor di testo avanzato, una sorta di evoluzione del blocco note di Windows, che fornisce numerosi strumenti per l'editing e la marcatura dei documenti "text based" come appunto i file XML. Una delle funzioni di spicco di questo software è la possibilità di creare delle proprie librerie di tag, denominate clipbook, e barre di pulsanti personalizzate con cui applicare la marcatura ai file di testo. Prima di procedere alla codifica di Baltico abbiamo pertanto provveduto a creare una nostra clipbook, e la relativa barra pulsanti, contenente i tag TEI che prevedevamo di adoperare.

---

[135] Si da per scontato che per tutti i software citati nella trattazione si faccia riferimento alla versione Windows. Ricordo che per tutte le attività descritte nelle pagine di questa relazione è stato adoperato un PC con sistema operativo Microsoft Windows 98 SE.

[136] <www.altova.com>.

[137] <www.notetab.com>.

[138] Col termine trial si indicano quelle versioni di un programma distribuite, come dice il termine (trial trad. prova), al fine di permettere all'utente di provare e valutare le potenzialità del software prima di procedere all'acquisto. I programmi trial in genere hanno delle limitazioni rispetto alla versione completa e spesso cessano di funzionare dopo un periodo di prova, in genere, di trenta giorni.

NoteTab, in virtù di queste sue potenzialità, ha costituito lo strumento principale nella marcatura di base del testo; un altro grosso vantaggio di NoteTab è costituito dal fatto che nella sua versione "Light" questo software è completamente gratuito.

III.5.4. Struttura dei documenti TEI

Una volta definita tutta questa serie di particolari preliminari si è potuti partire con la codifica effettiva del testo.

Ogni documento TEI, che ricordiamo è anche un documento XML valido, è caratterizzato dall'elemento radice <*TEI*.2> dal quale discendono due elementi figli <*teiHeader*> e <*text*>, i quali dividono praticamente in due parti il documento: il *TEI header* contenente i metadati ossia le informazioni bibliografiche ed editoriali relative al documento TEI ed alla sua fonte, ed il *TEI text* in cui trova posto il testo vero e proprio.

Il *TEI header,* il quale costituisce il frontespizio elettronico della pubblicazione digitale, consta di quattro parti: <*fileDesc*>, <*encodingDesc*>, <*profileDesc*>, <*revisionDesc*>, solo la prima è obbligatoria e deve quindi necessariamente essere presente, le altre tre sono opzionali.

L'elemento *text* si divide in tre[139] elementi:

- <*front*> (opzionale): contiene tutti i materiali di tipo avantestuale che tipicamente precedono il corpo del testo nelle edizioni a stampa, dalla pagina del titolo, al frontespizio, ad eventuali introduzioni o prefazioni.
- <*body*> (obbligatorio): comprende il contenuto testuale vero e proprio, al suo interno si susseguono, continuando a scendere nella struttura gerarchica dell'albero del documento, ulteriori divisioni: capitoli, paragrafi etc..

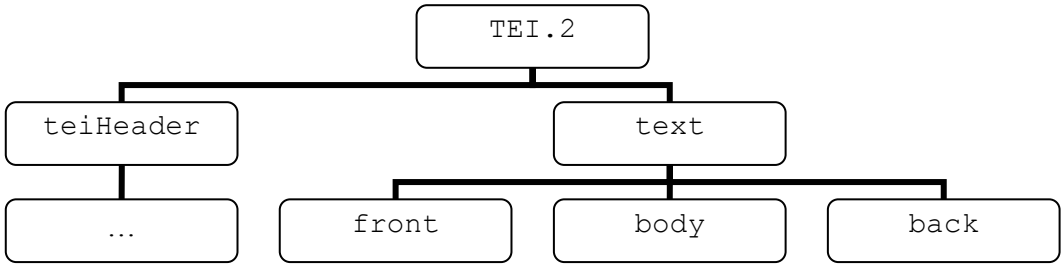
[139] In questo esempio si è preso in considerazione un testo unitario per testi compositi come un'antologia, una collezione di opere è previsto un ulteriore elemento figlio di <text> : <group> il quale è da impiegarsi nel caso di una serie di testi facenti parte della medesima pubblicazione, ognuno dotato della propria individualità (ognuno con un proprio front, body e back).

- <*back*> (opzionale): racchiude tutti gli annessi ed appendici che possono seguire la parte principale del testo, quali ad esempio: postfazioni, indici, glossari e materiali peritestuali in genere.

Schematicamente un testo unitario conforme alla DTD TEI presenterà una struttura di questo tipo:

```
<TEI.2>
  <teiHeader>
[Frontespizio elettronico (Metadati)]
  </teiHeader>
  <text>
     <front>
     [Materiali dell'avantesto]
     </front>
     <body>
     [Testo]
     </body>
     <back>
     [Materiali che seguono il testo]
     </back>
  </text>
</TEI.2>
```

Ecco invece come può essere rappresentato graficamente l'albero del documento TEI:


TEI.2
teiHeader
text
...
front
body
back

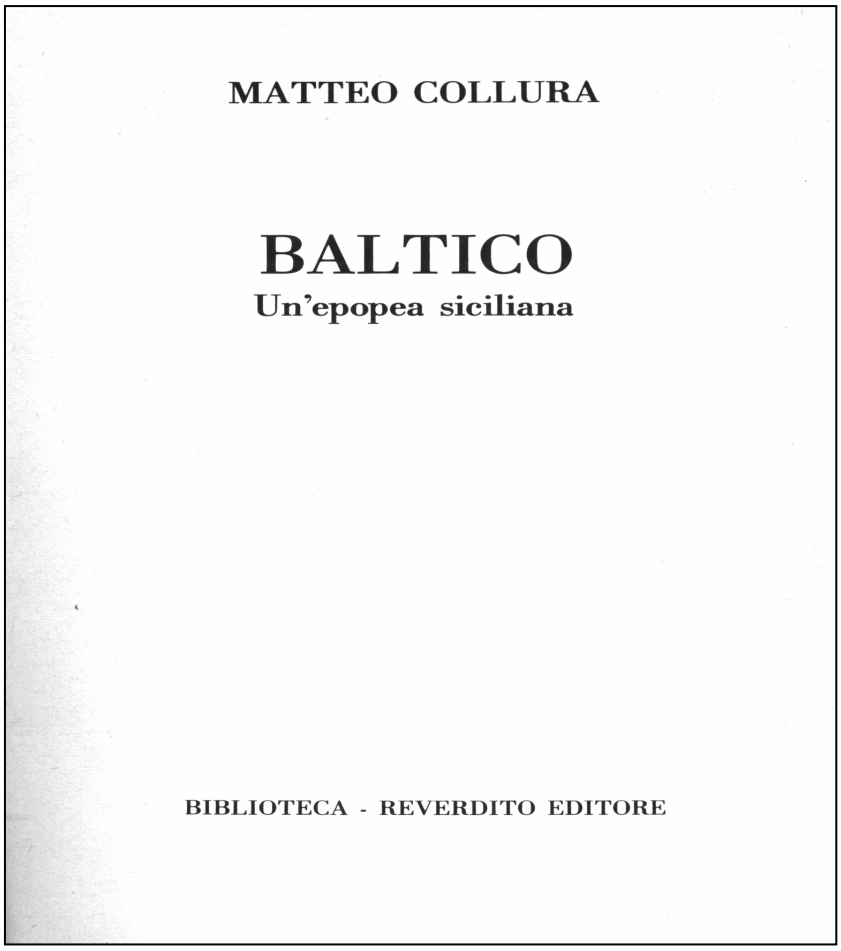
III.6. LA CODIFICA DI BALTICO

Sebbene il primo elemento di un documento TEI sia il <*teiHeader*>, entrando nel vivo della descrizione della codifica di Baltico, preferiamo iniziare l'esposizione illustrando le attività compiute a partire dal secondo elemento il <*text*>*,* il quale ospita i contenuti testuali dell'opera; infatti, una volta viste le modalità di codifica del testo sarà più semplice comprendere il contenuto dei metadati del <*teiHeader*>*.*

III.6.1. <front>

Il primo elemento figlio di <*text*> è <*front*>, all'interno di questo tag sono contenuti tutti i materiali di tipo avantestuale (testate, frontespizio, prefazioni, dediche, etc.), che si trovano prima dell'inizio vero e proprio del testo. Come detto il nostro lavoro si è limitato esclusivamente al contenuto testuale della fonte, tutti i materiali paratestuali non d'autore sono stati tralasciati, nello specifico la trascrizione e codifica di Baltico hanno avuto inizio dalla pagina cinque del testo fonte, contenente il frontespizio interno dell'opera, prima di questa pagina il volume contiene l'indicazione della collana ed una foto dell'autore Matteo Collura, queste pagine insieme alla pagina sei contenente l'indicazione del copyright sono state omesse nella codifica.

Riportiamo di seguito un'immagine che mostra come appare nel testo fonte la pagina del frontespizio interno da cui ha preso avvio la codifica e subito dopo quella stessa pagina codificata.




```
<titlePage>
 <docAuthor>
 <name type="persona.real">MATTEO COLLURA</name>
 </docAuthor>
 <docTitle>
 <titlePart type="main">BALTICO</titlePart>
 <titlePart type="sub">Un'epopea siciliana</titlePart>
 </docTitle>
 <docImprint>
 BIBLIOTECA - <publisher>REVERDITO EDITORE</publisher>
 </docImprint>
 </titlePage>
```

L’analisi parallela sia della rappresentazione grafica della fonte, sia della rispettiva codifica ci permette subito di fare due importanti considerazioni.

La codifica TEI/XML, essendo di tipo dichiarativo, è focalizzata sulla descrizione delle strutture funzionali del testo e non sul suo aspetto, che pure in questa pagina, presa ad esempio, è molto vario. I tag, che fra poco illustreremo nel dettaglio, esplicitano ruolo e funzione delle porzioni di testo non la loro formattazione, che semmai sarà applicata in un momento successivo per mezzo di adeguati strumenti. Il grande pregio della codifica dichiarativa è di dare significato al testo soprattutto a beneficio di agenti software; pensiamo, ad esempio, ad un motore di ricerca che venga interrogato per trovare tutti i libri dal titolo Baltico: potrà sicuramente individuare con grande precisione quei testi, in cui il titolo è identificato per mezzo di un’etichetta che indica “questo è un titolo”, come <*docTitle*>, a differenza invece di un’etichetta che dica “questa è una sequenza di caratteri in corpo 20 grassetto Times”.

La seconda considerazione riguarda il fatto che la codifica segue perfettamente l’ordine del testo fonte. Passando all’analisi del testo codificato vediamo che l'elemento <*titlePage*>, primo figlio di <*front*>*,* racchiude una serie di altri elementi che illustrano nel dettaglio la pagina del frontespizio. Il primo di questi è <*docAuthor*> il quale contiene il nome dell’autore, codificato a sua volta per mezzo del tag <*name*> accompagnato dall’attributo *persona*.*real,* vedremo meglio nel prosieguo della trattazione le caratteristiche di questo elemento*.*

La parte del frontespizio dedicata al titolo dell’opera è contenuta nell’elemento <*docTitle*>*,* al cui interno è codificata per mezzo di uno o più elementi <*titlePart*>; l’attributo *type* di questo tag viene adoperato per distinguere funzionalmente le diverse parti del titolo, con *main* si indica il titolo principale “Baltico”, con *sub* il sottotitolo “Un'epopea siciliana”. Per il valore di questi attributi ci siamo rifatti ai suggerimenti forniti nelle *guidelines* TEI, i testi TIL preferiscono al valore *main* consigliato dal manuale TEI un più italiano *princ*, mentre in *GriseldaOnLine* si riscontrano ambedue gli usi. È opportuna una precisazione, in molti casi la DTD

lascia ampia libertà al codificatore nella scelta del valore da dare agli attributi, ad esempio nel caso di *type* di <*titlePart*> la DTD si presenta così: type CDATA "main" , questa porzione di codice della DTD specifica che l'attributo *type* può avere come valore una qualsiasi sequenza di caratteri consentiti (CDATA) e che il valore di default dell'elemento, nel caso in cui l'attributo venga omesso è *main*; tuttavia nella maggior parte dei casi gli attributi vengono specificati in questa forma: nome_attributo CDATA #IMPLIED , stabilendo che l'attributo è facoltativo e non ha un valore di default; tutto ciò comporta che spesso per indicare la stessa cosa si ricorra a definizioni diverse nei vari progetti di codifica, per questo motivo, in mancanza di linee guida generali, al fine di mantenere una certa uniformità e coerenza con le pratiche di codifica relative almeno al territorio nazionale, abbiamo cerato di rifarci agli altri lavori analoghi esistenti in Italia. L'elemento <*docImprint*>, che chiude il nostro <*titlePage*>, contiene le informazioni relative all'editore, alla data ed al luogo di pubblicazione dell'opera così come sono presentate nel frontespizio; al suo interno vanno usati gli elementi <*docDate*>, <*publisher*> e <*pubPlace*> per codificare rispettivamente la data di pubblicazione, l'editore ed il luogo di pubblicazione. Nel caso di Baltico il frontespizio contiene il nome della collana e l'editore, mentre il nome di quest'ultimo è opportunamente codificato per mezzo del tag <*publisher*>, la DTD non prevede un elemento specifico per la collana[140] all'interno del tag <*docImprint*>, tuttavia consente di inserire un qualsiasi contenuto testuale (#PCDATA), pertanto abbiamo inserito il nome della collana, "Biblioteca", senza applicarvi alcuna codifica specifica.

Subito dopo la pagina del frontespizio e prima dell'inizio del contenuto della narrazione, il testo fonte presenta una dedica e due epigrafi, verosimilmente riconducibili all'autore dell'opera, pertanto sono state incluse all'interno del tag front essendo materiali dell'avantesto. Queste tre partizioni sono state codificate per mezzo di blocchi di tipo <*div*> accompagnati da attributi *type* che ne

[140] Tale informazione è comunque presente nel teiHeader.

specificano la funzione, i valori di questi attributi si rifanno al progetto TIL. Di seguito riportiamo il codice della dedica e delle due epigrafi:

```
<div type="ded" rend="italic">
 <p>A Bartolomeo Collura, mio padre.</p>
 </div>
```

Il valore *ded* dell'attributo *type* specifica che si tratta di una dedica, il secondo attributo *rend* indica che il testo della dedica è in corsivo. r*end* è un attributo globale, che può essere associato a quasi ogni elemento, il quale serve a definire alcune informazioni di tipo stilistico, in questa circostanza precisa che il testo nell'edizione fonte è in corsivo. Nonostante la natura prettamente dichiarativa dello schema TEI in taluni casi può essere utile fornire alcune informazioni relative alla formattazione del testo. Nel corso della nostra codifica questo attributo è stato usato solo poche volte col valore *italic*, per indicare dove il testo era in corsivo, corsivo che per altro abbiamo ritenuto riconducibile alla volontà dell'autore. Si noti che il testo della dedica è racchiuso dall'elemento <*p*>, in quanto il tag <*div*> serve solo a delimitare le macro-partizioni del testo, al suo interno non è consentito inserire testo, questo invece deve essere posto dentro il tag <*p*> il quale serve a delimitare le micro-partizioni del testo ossia i paragrafi.

Subito dopo la dedica incontriamo le due epigrafi:

```
<div type="ep">
 <p>Contagiati dal delirio delle escavazioni, subito accompagnato
dalla comparsa di affaristi scrocconi, si scoprirono impensate doti di
imprenditori; e sventrando valli e colline sognarono di arricchire,
mentre copioso colava lo zolfo e si ampliavano i cimiteri. Due secoli
di picconate cambiarono la faccia della terra. Subito si appales&#242;
il disastro, ma in quel turbinio di fortune immaginate nessuno vi fece
caso. Corsero ai ripari quando gi&#224; il vento screpolava gli spalti
delle zolfare e le erbacce cominciavano a nasconderne le bocche. Fu
come se un'ostinata bonaccia si fosse posata su un mare che era stato
in tempesta. Non lontano dai ruderi, oziosi, aspettarono sussidi e
```

```
pensioni; e polvere e silenzio sedimentarono sulla loro assurda
epopea.</p>
 </div>
 <div type="ep">
 <q type="citazione">
 Zufolava mentre andava al lavoro e parlava spesso di un futuro di
benessere e di abbondanza.
 <bibl>
 <author>
 <name type="persona.real">SHERWOOD ANDERSON</name>,</author>
 <title rend="italic">Un povero bianco.</title>
 </bibl>
 </q>
 </div>
```

La prima epigrafe non presenta particolari problemi, si noti soltanto l'uso delle entità numeriche al posto delle lettere accentate.

La seconda epigrafe invece è costituita da una citazione per questo motivo è racchiusa dal tag <*q*> con valore citazione, il riferimento bibliografico è codificato mediante gli opportuni tag. L'elemento <*q*> è usato per marcare citazioni e manifestazioni di discorso o di pensiero non espresse direttamente dalla voce narrante, come vedremo meglio in seguito, <*q*> viene adoperato sopratutto per la rappresentazione del discorso dei personaggi o di chi parla. La DTD TEI prevede l'elemento <*quote*> per la marcatura delle citazioni come quella della nostra epigrafe, ed infatti tale tag viene adottato in TIL, mentre la DTD TEI-Lite non lo include, pertanto come indicato nel manuale TEI-Lite abbiamo utilizzato il tag <*q*> anche per le citazioni associandovi il valore *citazione* per mezzo dell'attributo *type*.

III.7. LA CODIFICA DEL CORPO DEL DOCUMENTO (<body>)

L'elemento <*body*>, secondo e più importante figlio di <*text*>, contiene l'intero corpo di un singolo testo unitario, con l'esclusione di ogni elemento preliminare o di appendice, contenuti invece all'interno dei tag <*front*> e <*back*>.

All'interno dell'elemento <*body*> ricorrono ovviamente una serie di suddivisioni, necessarie a definire la struttura interna del testo indagato; tali suddivisioni, evidenziate usualmente nella forma materiale del documento originale, mediante convenzionali pratiche tipografiche: spaziature, pagine bianche, stili di carattere; ricalcano le tradizionali partizioni strutturali che hanno comunemente assunto dei nomi ben precisi in relazione al genere cui il testo appartiene: "capitolo", "paragrafo", "sezione", "parte", per la prosa; "poema", "canto", per la lirica; "atto", "scena", per il dramma.

Al livello più basso l'articolazione interna di un testo in prosa è delimitata dai paragrafi, nella codifica questa suddivisione è esplicitata usando il marcatore <*p*>, il corpo di un testo in prosa può essere costituito soltanto da una serie di paragrafi, tuttavia abitualmente tali paragrafi sogliono essere raggruppati insieme in capitoli, sezioni, sottosezioni, etc..

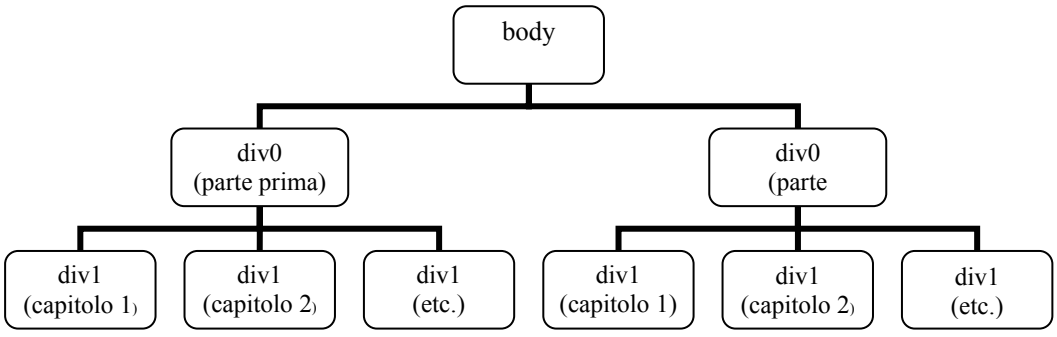
Lo schema TEI adotta i tag <*div#*> per rendere nella codifica queste segmentazioni di più alto livello del contenuto testuale. Gli elementi <*div#*> appartengono ad una famiglia di marcatori, che va da <*div0*>[141] sino a <*div7*> per mezzo dei quali vengono rappresentate gerarchicamente le partizioni strutturali dell'opera; ad esempio in un'opera divisa in capitoli, sezioni e sottosezioni adotteremo il tag <*div0*> per delimitare i capitoli, <*div1*> per le sezioni e <*div2*> per le sottosezioni. Accanto agli elementi <*div#*> numerati esiste un elemento

[141] L'uso di <div0> è facoltativo, nel caso in cui questo marcatore non venga adoperato le sezioni di più alto livello del testo vengono codificate per mezzo del tag <div1>.

<*div*> non numerato, il quale può ricorrere all'interno di altri elementi <*div*>, senza limiti alla profondità di annidamento. Questo marcatore, che abbiamo già incontrato in occasione delle epigrafi e della dedica presenti nell'avantesto, viene utilizzato per marcare quelle sezioni di cui non si vuole o non si può rendere esplicitamente la posizione gerarchica nell'articolazione strutturale del documento, oppure per rappresentare, alternativamente al sistema dei <*div#*> numerati, quei testi in cui siano presenti più di otto livelli nell'impianto organizzativo dell'opera.

Lo schema di Baltico è abbastanza semplice, il libro consta di due parti che raggruppano rispettivamente diciassette capitoli la prima, ventiquattro, o meglio ventitre più la conclusione, la seconda. Nella codifica abbiamo reso questa organizzazione dell'opera adoperando il tag <*div0*>, come elemento divisore di livello più alto, per identificare le due parti, mentre i singoli capitoli, al secondo livello nella scala gerarchica dell'albero del documento, sono stati racchiusi all'interno di marcatori di tipo <*div1*>.

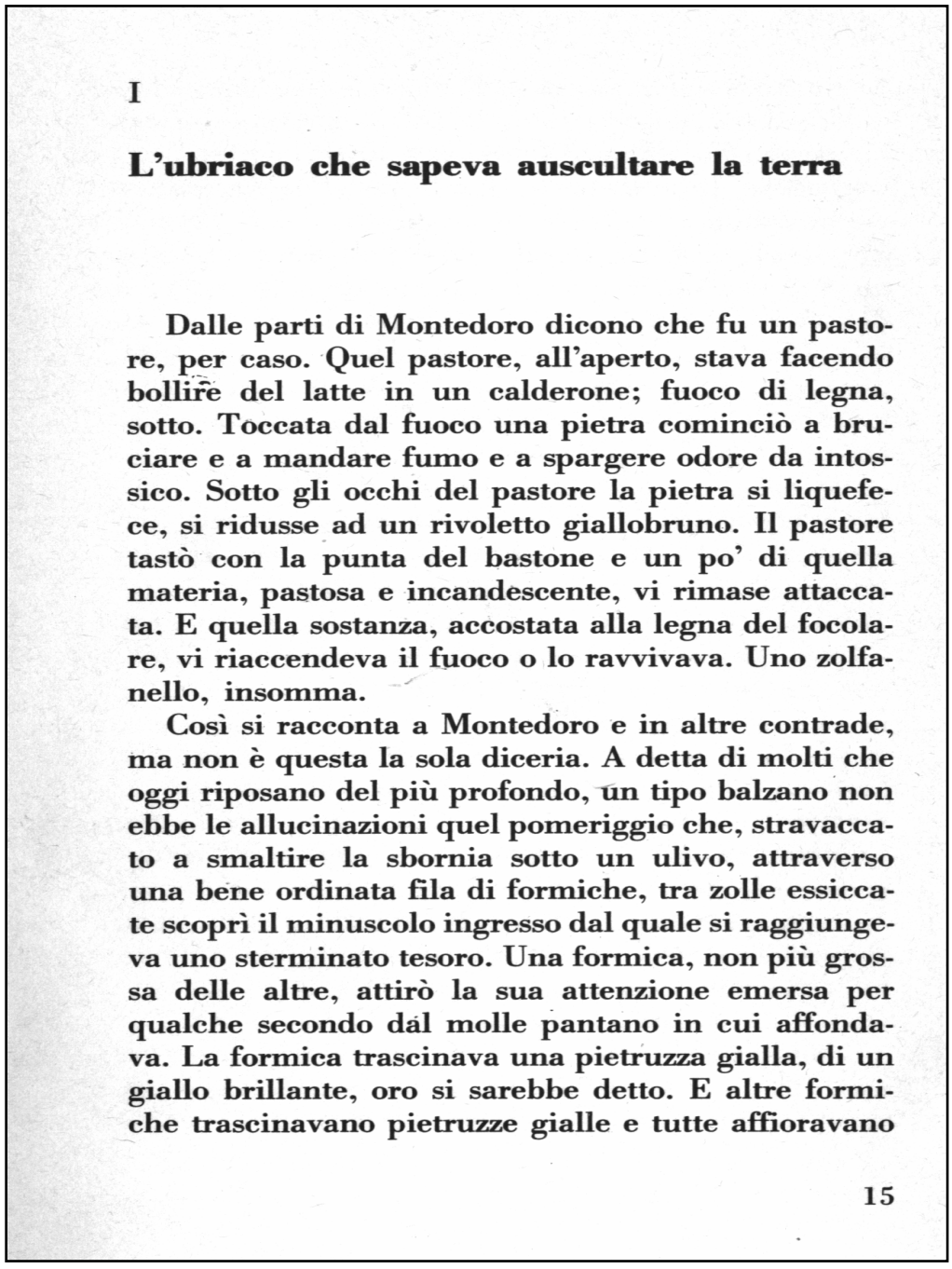
Ecco graficamente come si presenta la struttura del <*body*> di Baltico:



III.7.1. <div1>

Come già fatto, per discutere delle modalità di codifica dei singoli capitoli riportiamo la riproduzione grafica di una pagina del libro e di seguito la versione codificata di quella stessa pagina.

Ecco come si presenta nel testo fonte pagina 15, prima pagina del primo capitolo della prima parte dell'opera:



I

L'ubriaco che sapeva auscultare la terra

Dalle parti di Montedoro dicono che fu un pastore, per caso. Quel pastore, all'aperto, stava facendo bollire del latte in un calderone; fuoco di legna, sotto. Toccata dal fuoco una pietra cominciò a bruciare e a mandare fumo e a spargere odore da intossico. Sotto gli occhi del pastore la pietra si liquefece, si ridusse ad un rivoletto giallobruno. Il pastore tastò con la punta del bastone e un po' di quella materia, pastosa e incandescente, vi rimase attaccata. E quella sostanza, accostata alla legna del focolare, vi riaccendeva il fuoco o lo ravvivava. Uno zolfanello, insomma.

Così si racconta a Montedoro e in altre contrade, ma non è questa la sola diceria. A detta di molti che oggi riposano del più profondo, un tipo balzano non ebbe le allucinazioni quel pomeriggio che, stravaccato a smaltire la sbornia sotto un ulivo, attraverso una bene ordinata fila di formiche, tra zolle essiccate scoprì il minuscolo ingresso dal quale si raggiungeva uno sterminato tesoro. Una formica, non più grossa delle altre, attirò la sua attenzione emersa per qualche secondo dal molle pantano in cui affondava. La formica trascinava una pietruzza gialla, di un giallo brillante, oro si sarebbe detto. E altre formiche trascinavano pietruzze gialle e tutte affioravano

15

Ecco la stessa pagina nella codifica XML/TEI:

```
<pb n="15"/>
<div1 type="cap" n="1" id="I.1">
<head type="ord">I</head>
```

<head type="tem">L'ubriaco che sapeva auscultare la terra</head>

Dalle parti di <rs type="luogo.real">Montedoro</rs> dicono che fu un pastore, per caso. Quel pastore, all'aperto, stava facendo bollire del latte in un calderone; fuoco di legna, sotto. Toccata dal fuoco una pietra cominciò a bruciare e a mandare fumo e a spargere odore da intossico. Sotto gli occhi del pastore la pietra si liquefece, si ridusse ad un rivoletto giallobruno. Il pastore tastò con la punta del bastone e un po' di quella materia, pastosa e incandescente, vi rimase attaccata. E quella sostanza, accostata alla legna del focolare, vi riaccendeva il fuoco o lo ravvivava. Uno zolfanello, insomma.

Così si racconta a <rs type="luogo.real">Montedoro</rs> e in altre contrade, ma non è questa la sola diceria. A detta di molti che oggi riposano del più profondo, un tipo balzano non ebbe le allucinazioni quel pomeriggio che, stravaccato a smaltire la sbornia sotto un ulivo, attraverso una bene ordinata fila di formiche, tra zolle essiccate scoprì il minuscolo ingresso dal quale si raggiungeva uno sterminato tesoro. Una formica, non più grossa delle altre, attirò la sua attenzione emersa per qualche secondo dal molle pantano in cui affondava. La formica trascinava una pietruzza gialla, di un giallo brillante, oro si sarebbe detto. E

```
altre formiche trascinavano pietruzze gialle e tutte
affioravano
[…]
</p>
[…]
</div1>
```

Nella codifica ogni capitolo viene aperto dal marcatore <*div1*> associato a tre attributi: *type* designa la tipologia della partizione, il valore *cap*, abbreviazione comunemente usata sia in TIL sia in *GriseldaOnLine*, indica che si tratta di un capitolo. L'attributo *n* specifica un breve nome mnemonico o un numero per la divisione, nel nostro caso indica il numero di capitolo all'interno della parte. L'attributo *id,* infine, fornisce un identificatore univoco per la divisione, tale attributo non può presentarsi col medesimo valore in nessun altro elemento, pena l'invalidità sintattica del documento; *id* serve ad identificare in modo unico ed inequivocabile un elemento e può essere usato per i riferimenti incrociati o per altri collegamenti. Nel caso di Baltico il valore di *id* in <*div1*> è dato dal numero romano della parte cui il capitolo appartiene, separato con un punto dal numero arabo indicante il numero di capitolo all'interno della parte, come vedremo in seguito, nella produzione dei formati di output, l'attributo *id* è stato molto utile per la realizzazione di sistemi di collegamenti e riferimenti di vario tipo.

III.7.2. <head>

Subito dopo il tag <*div1*>, che può essere considerato come una sorta di contenitore del capitolo, inizia la codifica del testo vero e proprio. Come possiamo vedere nella riproduzione grafica della pagina, ogni capitolo si apre con i titoli in grassetto. Nella codifica TEI titoli ed intestazioni dei blocchi di tipo <*div#*> vengono marcati grazie all'elemento <*head*>, mediante l'attributo *type* si indica il tipo di titolo, i valori fra cui si è scelto, già adottati sia in ambito TIL sia da *GriseldaOnLine,* sono:

- **conv** (*convenzionale*) per i titoli convenzionali come "Capitolo Primo", "Parte I";
- **ord** (*ordinale*) per i titoli limitati a soli numeri arabi o romani;
- **tem** (*tematico*) per i titoli tematici in genere costituiti da pochi sintagmi;
- **desc** (*descrittivo*) per i titoli descrittivi più lunghi.

Nel nostro testo i capitoli vengono introdotti da due titoli uno di tipo ordinale in numeri romani ed uno di tipo tematico, per tale motivo abbiamo adottato due volte il marcatore <*head*> una volta con valore *ord* e la seconda con valore *tem* per codificare le intestazioni del capitolo.

III. 7.3. Paragrafi

I paragrafi costituiscono la più piccola unità strutturale di un testo in prosa. Nella codifica i paragrafi sono delimitati dall'elemento <*p*>; nella pratica tipografica moderna l'inizio di un nuovo paragrafo è tipicamente contraddistinto da un rientro a destra nella prima riga, proprio sulla base dei rientri nel testo fonte, come si può vedere, abbiamo marcato i singoli paragrafi nel codice.

III.7.4. Salti di pagina

Nella pagina di esempio riportata sopra, immediatamente prima del tag di apertura del blocco <*div1*> incontriamo l'elemento `<pb n="15"/>`; <*pb*> (page break – interruzione di pagina) è l'elemento deputato a segnalare nella codifica i salti di pagina presenti nel testo fonte delimitando i confini tra le pagine in un sistema di riferimento standard. L'elemento <*pb*> appartiene alla classe dei cosiddetti elementi "*milestone*", in italiano "pietra miliare", questi elementi non marcano come gli altri delle porzioni più o meno estese di testo, bensì servono ad identificare dei punti di riferimento all'interno del contenuto testuale, come si sarà notato <*pb*> è infatti un tag autonomo che non prevede un tag di chiusura. Nella codifica dei salti pagina ci siamo attenuti fedelmente a quanto suggerito nelle *guidelines* TEI, in ciò discostandoci sia dalla metodologia adottata da

GriseldaOnLine sia, probabilmente[142], da TIL; i <*pb*> sono stati da noi inseriti in cima alla pagina cui fanno riferimento, o, per essere più precisi, all'inizio di quella porzione di testo codificato corrispondente con il contenuto di una pagina nella fonte di riferimento; mediante l'attributo *n* è stato indicato in forma normalizzata il numero associato nella fonte alla pagina che segue il punto di inserimento; *GriseldaOnLine*, invece, ha inserito il tag per codificare i *page break* di seguito all'ultima parola che compare nella pagina, nella sostanza, la differenza con il metodo da noi adottato, che ricordiamo è quello consigliato dal manuale ufficiale TEI, sono minime, i <*pb*> nel testo codificato vengono a cadere quasi sempre nello stesso punto con ambedue le metodologie, tuttavia la pratica di codifica adottata da *GriseldaOnLine* presenta un ovvio inconveniente per la prima ed ultima pagina del testo.[143]

Una precisazione, allorché un salto pagina nell'edizione di riferimento spezza una parola sillabandola, abbiamo posto il tag <*pb*> dopo questa parola, assumendo che essa appartenga alla pagina in cui inizia.

Si sarà notato che nel testo codificato ricorrono anche degli elementi, che marcano alcune parole, come ad esempio `<rs type="luogo.real">Montedoro</rs>`, di questo come degli altri elementi intralineari parleremo nel prossimo paragrafo.

III.7.5. Elementi intralineari

All'interno dei vari elementi strutturali (<*div#*>, <*p*> etc.) la codifica TEI prevede tutta una serie di elementi, conosciuti come "elementi intralineari", che di norma si applicano a singoli termini oppure a sequenze di parole o frasi, i quali vengono

[142] Su questo punto il manuale per la codifica dei testi TIL non è molto chiaro, sembrerebbe tuttavia di poter dedurre, da alcuni riferimenti indiretti nel testo, che anche gli studiosi romani adottino un sistema di codifica dei salti pagina analogo a quello di *GriseldaOnLine.*

[143] Il problema riguarda in particolar modo la numerazione dei <*pb*>, infatti, con il metodo adoperato in *GriseldaOnLine,* il valore dell'attributo *n* indica in forma normalizzata il numero associato alla pagina che segue il punto d'inserimento. Con questo sistema, se abbiamo compreso bene quanto fatto dagli studiosi bolognesi, si rischia che il valore della prima pagina, che ovviamente non è preceduta da nessun'altra pagina, e viceversa il valore dell'ultima che altresì non è seguita da altre pagine, non venga mai registrato.

principalmente usati per marcare i nomi di persona e luogo, le date, i discorsi diretti nei testi in prosa e diversi altri fenomeni di evidenziazione.

III.7.6. Nomi ed espressioni referenziali

Leggiamo nel manuale Tei-Lite "Una *espressione referenziale* è un'espressione che si riferisce ad una persona, un luogo, un oggetto, etc."; lo schema TEI prevede due elementi per codificare queste espressioni <*rs*> e <*name*>, <*rs*> è in genere usato per nomi o espressioni referenziali generiche, <*name*> contiene invece nomi propri o espressioni sostantivali.

III.7.7. Nomi propri

Nella codifica di Baltico tutti i nomi propri sono stati racchiusi all'interno del tag <*name*>. Riportiamo a titolo d'esempio alcuni nomi incontrati nel romanzo di Collura oggetto della nostra analisi:

```
<name type="persona.real" key="Henry Temple Palmerston">Henry Temple
Palmerston</name>
<name type="persona.real" key="Giuseppe Garibaldi">Garibaldi</name>
<name type="persona.fit" key="Bartolomeo Ardito">Bartolomeo
Ardito</name>
<name type="persona.fit" key="Speranza Ardito">Speranza Ardito</name>
```

Come si può rilevare mediante l'attributo *type* si è specificato se il nome faccia riferimento ad una persona reale oppure ad un personaggio di fantasia. I valori possibili nel nostro testo per l'attributo *type* sono stati pertanto "*persona*.*real*" per i personaggi reali, "*persona*.*fit*" per i personaggi di fantasia. La distinzione ***.real** ***.fit**, tra soggetti reali oppure fittizi, non è stata adoperata esclusivamente per i nomi propri ma per tutte le espressioni referenziali codificate nel testo.

Siamo in primo luogo debitori ai codificatori dei testi elettronici del portale *GriseldaOnLine*, che come noi hanno adottato la TEI-Lite, nella scelta di questo sistema di denominazione dell'attributo *type* di tali marcatori; in verità una simile distinzione è presente anche nei testi TIL, tuttavia essendo stato, per questi ultimi,

adottato l'intero schema TEI, si sono talvolta utilizzati dei tag diversi nella marcatura delle espressioni referenziali, quali ad esempio <*persName*> e <*placeName*> non previsti dalla TEI-Lite.

All'elemento <*name*> si accompagna anche un altro attributo, *key* grazie al quale si fornisce una sequenza di caratteri che identifica in modo univoco un personaggio nel contesto dell'opera.

III.7.8. Nomi di luogo ed altre espressioni referenziali

Tutti i nomi di luogo sono stati codificati mediante il tag <*rs*>; si ricorderà, ad esempio, <rs type="luogo.real">Montedoro</rs> che abbiamo incontrato nella porzione di codice riportata nelle pagine precedenti; come si vede si è mantenuta la distinzione *.real *.fit già adottata per i nomi propri. Oltre che per i toponimi con <*rs*> si sono marcati anche altri termini giudicati rilevanti, i valori dell'attributo *type* nella codifica di Baltico sono*: luogo.real; luogo.fit; famiglia.fit; soprannome.fit; istituzione.real; societa.real* (manca l'accento perché altrimenti alcuni parser non considerano ben formato il file); *zolfara.real*. Si noti che in alcuni casi mediante i valori di <*rs*> si è fornita una specificazione ulteriore di quelli che genericamente avrebbero potuto essere codificati come nomi o luoghi, in particolare abbiamo preferito adottare *zolfara.real* invece di *luogo.real*, in quanto il tema dell'estrazione dello zolfo in Sicilia è uno degli argomenti principali dell'opera, abbiamo pertanto optato per dedicare una particolare attenzione a questo elemento in vista di eventuali esigenze di analisi e di interpretazione del testo.

Questi elementi appena descritti, come d'altronde quasi tutti gli elementi TEI, sono portatori principalmente di quella che potremmo definire meta-informatività interna, grazie a questi marcatori si specifica meglio il valore e la funzione del contenuto testuale dell'opera, si dà, in pratica, significato al testo elettronico cosicché applicazioni informatiche possano agevolare studiosi e lettori in genere del testo nell'analisi e nella fruizione dello stesso, si pensi ad esempio ai

programmi di trattamento automatico del linguaggio naturale oppure ai già citati software di infromation retrival.

III.7.9. Riferimenti temporali

Tra gli elementi del testo che lo schema di codifica TEI prevede siano sottoposti ad un'opportuna marcatura vi sono date ed orari; i tag designati all'identificazione dei riferimenti temporali sono <*date*> e <*time*>. Mediante l'attributo *value* viene indicato il valore normalizzato della data in un formato standard, ad esempio l'espressione "nove di luglio del duemila" verrà così codificata `<date value="09-07-2000">9 di luglio del duemila</date>`, le *Guidelines* TEI consigliano di adottare lo standard ISO 8601:2000 basato sullo schema **aaaa-mm-gg** (anno-mese-giorno) nella rappresentazione dei valori dell'attributo, tuttavia tale sistema è ben poco utilizzato in Italia, perciò abbiamo ritenuto opportuno conformarci alla forma canonica **gg-mm-aaaa** (giorno-mese-anno) abitualmente adoperato nei nostri lidi ed adottato da tutti gli altri progetti di codifica italiani. Per quanto riguarda le date parziali, ad esempio 1988, nove luglio, abbiamo proceduto, al fine di evitare fraintendimenti, sostituendo la parte mancante del riferimento temporale con delle "X", le date riportate poco sopra nella codifica si presenterebbero ad esempio così:

```
<date value="xx-xx-1988">1988</date>,
<date value="09-07-xxxx">9 luglio</date>.
```

Nella codifica di Baltico non è servito mai il tag <*time*>, il suo uso, comunque, è analogo a quello di <*date*>.

III.7.10. Discorso diretto e citazioni

Uno dei tag per la marcatura di fenomeni intralineari, che più spesso ricorre nella codifica TEI di Baltico, è <*q*>. Questo elemento, cito testualmente il manuale TEI-Lite, "contiene una citazione, manifesta o meno - una rappresentazione di discorso o pensiero marcata come se fosse espressa da qualcun altro (sia essa realmente citata o meno); in narrativa, le parole sono di solito quelle di un personaggio o di

chi parla; nei dizionari, <*q*> può essere usato per indicare esempi, reali o inventati, dell'uso di un termine."[144] Principalmente <*q*> è utilizzato per codificare il discorso diretto, prima di esaminare nei dettagli le caratteristiche di questo elemento riportiamo a titolo di esempio, alcuni paragrafi del testo codificato del romanzo ricchi di discorso diretto:

```
<p>
<q type="diretto" who="Sebastiano Azzalora">&#171;Chi ti ha
avvelenato?&#187; lo accolse l'amico.</q>
</p>
<p>
<q type="diretto" who="Raimondo Battaglia">&#171;I pensieri non mi
fanno dormire. E dai pensieri nascono le idee. Ne ho una che far&#224;
parlare, qui intorno&#187;</q>. <name type="persona.fit" key="Raimondo
Battaglia">Raimondo Battaglia</name> aveva gli occhi infebbrati e le
mascelle tremanti.
</p>
<p>
<q type="diretto" who="Sebastiano Azzalora">&#171;Se non si dorme
s'impazzisce&#187;</q>. <name type="persona.fit" key="Sebastiano
Azzalora">Sebastiano Azzalora</name> riprese a zappare.
</p>
<p>
<q type="diretto" who="Raimondo Battaglia">&#171;In questa terra
c'&#232; un tesoro: basta scavare. Ma non per fare quello che stai
facendo tu. Se riusciamo a mettere le mani sullo zolfo diventeremo
ricchi&#187;</q>. Urlava, non parlava, <name type="persona.fit"
key="Raimondo Battaglia">Raimondo Battaglia</name>.
```

[144] LOU BURNARD, C. M. SPERBERG-MCQUEEN, *TEI Lite: introduzione alla codifica dei testi*, trad. it. Fabio Ciotti, Guendalina Demontis, Giuseppe Gigliozzi, Massimo Guerrieri, Andrea Loreti in *TEI Website* <http://www.tei-c.org/Lite/teiu5_it.htm>
(ed. orig. *TEI U5: Encoding for Interchange: an introduction to the TEI*, <http://www.tei-c.org/Lite/teiu5_en.tei >).
gennaio 1998, (20 febbraio 2004).

```
</p>
<p>
<q type="diretto" who="Raimondo Battaglia">&#171;Qui sotto c'&#232;
zolfo, e laggi&#249; c'&#232; zolfo, e laggi&#249;, e ovunque riesci a
vedere. Basta avere i mezzi per arrivarci&#187;</q>.
</p>
```

Come si sarà notato ogni battuta dei due personaggi Raimondo Battaglia e Sebastiano Azzalora è racchiusa dal tag <*q*>, normalmente nei testi a stampa il discorso diretto è introdotto e delimitato da particolari caratteri come le virgolette o i trattini lunghi, nel testo del nostro esempio il discorso diretto è marcato dalle doppie virgolette ad angolo, o meglio dalle rispettive entità numeriche (« corrisponde al carattere **[«]** mentre » al carattere **[»]**) che come si può vedere sono inserite all'interno dell'elemento <*q*>.

Due attributi, *type* e *who,* specificano il valore di questo marcatore; *type* viene in genere usato per indicare se il brano sia parlato o pensato, se rappresenti un discorso diretto oppure, ad esempio, un monologo interiore. Nella nostra codifica abbiamo dato a questo attributo il valore "*diretto*" oppure "*pensato*" a seconda delle esigenze poste dalla narrazione. Col secondo attributo *who* si identifica colui che pronuncia il discorso, ove possibile il valore di *who* corrisponde con la chiave associata al nome del personaggio nell'attributo key del tag <*name*>. Spesso tuttavia il parlante non è identificato precisamente con il nome durante la narrazione ma piuttosto attraverso un titolo ovvero una elemento distintivo fisico o morale, in questo caso si è preferito qualificarlo nel tag per mezzo di queste caratteristiche, ad esempio: uomo dai rattoppi multicolori, vecchio monaco, sicari, ubriacone etc..

In taluni casi però viene a mancare anche questa minima caratterizzazione del personaggio, o altrimenti si fa troppo labile; spesso, inoltre, viene riportata come discorso diretto la voce popolare, in tutti questi casi si è adoperato un valore

generico "personaggio/i indefinito/i", "popolo indefinito" per l'identificazione del parlante mediante l'attributo *who*.

Come già accennato in precedenza il tag <*q*>, accompagnato dall'attributo *type* con valore citazione

(<q type="citazione">…</q>*),* è servito anche per delimitare le citazioni presenti nel testo. Si è deciso nella codifica, mediante l'attributo *rend*, di registrare, ove necessario, in che modo è stampata la citazione nella fonte materiale, i valori usati nell'attributo per far ciò sono stati: *bloc* per citazioni messe in risalto in un blocco di testo graficamente autonomo ed *italic* per le citazioni in corsivo.

Alcune citazioni in Baltico riportano dei brevi brani di testo in versi, come ad esempio la citazione presente a pagina trenta del libro, che riportiamo di seguito:

```
<pb n="30"/><p>
<q type="diretto" who="Serafino Collica">&#171;Pu&#242;
sbagliarsi <name type="persona.real" key="Dante
Alighieri">Dante</name>?&#187;</q> disse quasi a se stesso. Poi,
assaporando lo sgorgare dei versi. scand&#236;
<q type="diretto" who="Serafino Collica">
<q type="citazione" rend="bloc">
      <lg part="N" type="terzina">
      <l>&#171;E la bella <rs type="luogo.real">Trinacria</rs>,
che caliga</l>
      <l>tra <rs type="luogo.real">Pachino</rs> e <rs
type="luogo.real">Peloro</rs> sopra 'l golfo</l>
      <l>che riceve da Euro maggior briga,</l>
      </lg>
      <lg part="Y" type="terzina">
      <l>non per Tifeo ma per nascente solfo,</l>
      <l>attesi avrebbe li suoi regi ancora...</l>
      </lg>
      </q>
```

```
&#200; tutto scritto, basta saper leggere&#187;</q>.
</p>
```

Il personaggio Serafino Collica pronuncia i celebri versi[145] dell'ottavo canto del Paradiso, i quali, all'interno della citazione, sono stati marcati nella codifica per mezzo dei tag dello schema TEI specifici per la lirica.

L'elemento <*lg*> contiene un gruppo di versi che costituiscono un'unità formale, come ad esempio una stanza, un refrain etc.; l'attributo *part* specifica se l'unità strofica contenuta all'interno di <*lg*> è completa o meno, tornando al nostro esempio vediamo che la prima terzina essendo riportata nella sua interezza presenta l'attributo *part* con valore *N*[146], della seconda terzina invece sono citati soltanto i primi due versi pertanto l'attributo *part* in questo caso ha valore *Y*. Con l'attributo *type* si indica il tipo di blocco strofico marcato da <*lg*>, nella circostanza trattandosi di una terzina dantesca abbiamo adottato il valore terzina. All'interno di <*lg*> i singoli versi sono delimitati dal tag <*l*>, analogamente ad <*lg*> anche <*l*> può essere associato all'attributo *part* per specificare la completezza metrica o meno del verso.

Un altro caso particolare di citazione è rappresentato dal capitolo ventidue della parte seconda di Baltico dal titolo "Dove fedelmente si trascrive una lettera trovata per caso". Come già si evince dal titolo, in questo capitolo si riporta una lettera che un emigrato in America invia ai propri cognati in Sicilia. L'intero capitolo può essere considerato una lunga citazione e ad avvalorare questa ipotesi vi è il fatto che il testo del capitolo è completamente in corsivo. Assai difficoltosa è stata la decisione su come rendere nella codifica questo capitolo, alla fine abbiamo scelto di rifarci alle indicazioni contenute nel manuale del progetto BibIt ed abbiamo così trascritto nel codice il capitolo:

```
<pb n="200"/>
<div1 type="cap" n="22" id="II.22">
<head type="ord">XXII</head>
```

[145] Per precisione si tratta dei versi da 67 a 71.
[146] I due valori N e Y indicano rispettivamente che l'unita strofica è completa ovvero incompleta.

<head type="tem">Dove fedelmente si trascrive una lettera trovata per caso</head>
<q type="epistola" >
<text>
<body>
<opener rend="italic">
<salute>Cari cognati <name type="persona.fit" key="Sebastiano">Sebastiano</name> e <name type="persona.fit" key="Raimondo">Raimondo</name>.</salute>
</opener>

Ho ricevuto la vostra lettera e piango ancora per le cose tristi che mi avete scritto. Sappiate, però, che la vostra adorata sorella <name type="persona.fit" key="Rosetta">Rosetta</name> sta bene e vi manda tanti cari saluti.
[. . .]

<closer rend="italic">
<salute>Aspettando vostre notizie vi mando un forte abbraccio e tanti fraterni saluti anche da parte di vostra sorella <name type="persona.fit" key="Rosetta">Rosetta</name> che vi prega di questo: quando capita che qualcuno di
<rs type="luogo.real">Grotte</rs> parte per
l'<rs type="luogo.real">America</rs>, mandatele un fiore di zolfo. Sceglietela voi una bella pietra brillante, perché, testa sbadata, non ha più trovata quella che le avevo regalato io tanti anni fa. Decidetevi a venire, che
l'<rs type="luogo.real">America</rs> è grande e c'è posto per tutti. Ancora fraterni saluti dal vostro caro cognato
</salute><signed>

```
<name type="persona.fit" key="Pinzello Rosario">Pinzello
Rosario</name></signed>.
</closer>
</body>
</text>
</q>
</div1>
```

Per comodità abbiamo riportato solamente la parte iniziale e quella finale del capitolo.

Come si può vedere il testo del capitolo riportante l'epistola è interamente racchiuso all'interno dell'elemento <*q type="epistola"*> , in questo soltanto ci siamo, necessariamente, discostati dalle indicazioni del manuale BibIt, il quale invece prescrive l'utilizzo dell'elemento <*quote*> che, come abbiamo già avuto modo di verificare, non è incluso nella DTD TEI-Lite.

Essendo la citazione costituita da un testo completo, nello specifico un'epistola, sulla scorta di quanto indica il manuale del progetto Biblioteca Italiana (BibIt), essa è stata codificata all'interno di <*q*> dentro un elemento <*text*>, seguito da <*body*>. Per la marcatura del testo della lettera si sono usati i tag specifici per la codifica delle epistole, le formule di apertura e chiusura delle lettere sono contenute dentro gli elementi <*opener*> e <*closer*>, i quali racchiudono alcuni elementi tipici che a loro volta sono codificati mediante ben determinati tag: <*dateline*> contiene le indicazioni di tempo e luogo di una lettera, <*byline*> contiene invece le indicazioni di responsabilità di una lettera, <*salute*> è usato per le formule di saluto e le dediche, mentre <*signed*> viene adoperato per la firma dell'autore in calce a una epistola.

III.8. I MATERIALI CHE SEGUONO IL TESTO (<back>)

Quasi tutti i libri di Matteo Collura si concludono con una nota, contenente indicazioni sulla genesi dell'opera ed alcune considerazioni finali sulla stessa; non fa eccezione Baltico che pure termina con una breve nota-commento dell'autore.

Così come il romanzo si chiude con la nota dello scrittore, anche il nostro file XML finisce con la codifica della nota finale, la quale non facendo ovviamente parte del contenuto dell'opera ma piuttosto degli annessi del libro è stata inserita all'interno del tag <*back*>, l'ultimo elemento del <*text*> TEI, destinato ad ospitare tutti i materiali peritestuali, annessi o d'appendice, che seguono la parte principale del testo.

Di seguito riportiamo per intero il testo codificato della nota finale, unico elemento degli annessi del documento sottoposto a codifica[147]:

```
<back>
 <pb n="211"/>
 <div type="nota" id="nota">
 <head type="conv">NOTA</head>
<p>All'inizio avevo pensato ad un racconto che raccogliesse le cose
che mio padre, grottese, mi narrava sugli zolfatari del suo paese. In
parte mi sono attenuto al programma (perci&#242; questa storia va
considerata immaginaria), ma ho aggiunto qualcos'altro e, con mia
stessa sorpresa, ne &#232; venuta fuori una sorta di epopea degli
zolfatari siciliani. E non poteva non essere cos&#236;, dato che un
paio di secoli di storia siciliana sono stati, se non proprio
condizionati, caratterizzati almeno dalla situazione di monopolio
obiettivo che la <rs type="luogo.real">Sicilia</rs>ha avuto nel campo
dell'estrazione zolfifera. La <rs type="luogo.real">Sicilia</rs>,
```

[147] Si ricorderà a tal proposito, sulla scorta dei principi editoriali elaborati in ambito TIL, la nostra decisione di omettere tutti i materiali paratestuali non riconducibili direttamente all'autore, pertanto è stato escluso dalla codifica l'indice che nella fonte materiale si incontra subito dopo la nota conclusiva di Collura.

```
insomma, avrebbe potuto fare con lo zolfo quello che i paesi arabi
fanno con il petrolio. Avrebbe potuto e non l'ha fatto: ecco,
scrivendo degli zolfatari che mio padre conobbe, mi sono imbattuto in
uno dei nodi dell'intricato, irrimediabile dissesto economico della
<rs type="luogo.real">Sicilia</rs>. E il risultato complessivo non
poteva che dare il senso di un fallimento, l'eterno fallimento delle
speranze dei siciliani.</p>
 <signed>
 <abbr type="sigla" expan="Matteo Collura">m.c.</abbr>
 </signed>
 </div>
 </back>
```

Non vi sono indicazioni particolari relative a questa porzione di codice. Come si evince ricorrono gli stessi elementi precedentemente illustrati. Si noti, tuttavia, che il blocco è racchiuso dal marcatore <*div*> e non <*div1*> adottato per i singoli capitoli, in quanto così si è cercato di distinguere funzionalmente dai capitoli questo punto nella struttura dell'opera, rimarca ciò l'attributo type="nota" , che specifica la differente natura di questa porzione di testo.

III.9. IL FRONTESPIZIO ELETTRONICO

Ogni documento TEI è sostanzialmente diviso in due parti dai figli dell'elemento radice <*TEI*.2>, <*teiHeader*> e <*text*>; dopo aver parlato del *TEI text* contenente la trascrizione codificata del testo vero e proprio, ci accingiamo ad illustrare la testata TEI la quale raccoglie tutte le informazioni per la descrizione bibliografica del testo elettronico e della fonte materiale di riferimento.

Il <*teiHeader*> ospita le informazioni necessarie alla documentazione del testo elettronico: i metadati. Il lettore forse ricorderà la metaforica definizione di metadati come "bit che parlano di altri bit" ovvero "dati su altri dati", da noi data nelle pagine precedenti; mediante i metadati del <*teiHeader*> si risponde a vari ordini di esigenze: la documentazione bibliografica del testo digitale e della sua fonte materiale, la preservazione nel tempo del materiale codificato, l'integrazione in corpora documentali e la catalogazione ed indicizzazione del testo, la certificazione della originalità, del grado di precisione e della provenienza del documento digitale.

Il <*teiHeader*> è diviso in quattro sezioni dai suoi elementi figli:

- ***<fileDesc>***: contiene la descrizione bibliografica del documento elettronico e della sua fonte cartacea;
- ***<encodingDesc>***: illustra le metodologie impiegate nella codifica;
- ***<profildreDesc>***: racchiude una serie di informazioni accessorie che caratterizzano il testo;
- ***<revisionDesc>***: riassume la storia delle revisioni e delle modifiche che il documento elettronico ha subito.

Gli elementi della testata TEI presentano una, per così dire, desinenza indicativa della loro funzione generale; i marcatori i cui nomi finiscono in *Desc* (per *description*), come quelli appena incontrati, hanno natura descrittiva, invece gli elementi i cui nomi si concludono in *Stmt* (per *statement*) contengono in genere un

gruppo di tag che marcano informazioni strutturate, mentre gli elementi i cui nomi terminano in *Decl* (per *declaration*) includono informazioni sulle pratiche di codifica specifiche messe in atto nel documento.

III.9.1. Descrizione del file

Il primo elemento ed unico obbligatorio della testata TEI è <*fileDesc*>. Questo marcatore racchiude informazioni analoghe a quelle contenute nel frontespizio di un testo a stampa e può essere, per certi versi, considerato una sorta di frontespizio elettronico del documento digitale. <*fileDesc*> si articola in una serie di sette elementi figli, che forniscono una descrizione bibliografica completa del file e della sua fonte: <*titleStmt*>, <*editionStmt*>, <*extent*>, <*publicationStmt*>, <*seriesStmt*>, <*notesStmt*>, <*sourceDesc*>; di questi sotto-elementi solo tre sono obbligatori: <*titleStmt*>, <*publicationStmt*> e <*sourceDesc*>.

III.9.2. Dichiarazione del titolo

Il primo elemento <*titleStmt*> contiene il titolo del documento elettronico. Mostriamo di seguito come si è proceduto per questo elemento nella codifica di Baltico:

```
<teiHeader>
  <fileDesc>
   <titleStmt>
    <title>Baltico. Un'epopea siciliana: edizione elettronica</title>
    <author><name type="forename">Matteo</name> <name
type="surname">Collura</name></author>
    <respStmt>
      <resp>Digitalizzazione e codifica elettronica a cura di</resp>
       <name>Salvatore Salzillo</name>
       <resp>Supervisione</resp>
       <name>Prof. Lucrezia Lorenzini</name>
    </respStmt>
```

```
</titleStmt>
```

All'interno dell'elemento come si può vedere le informazioni vengono strutturate mediante un gruppo di ulteriori tag. Il marcatore <*title*> riporta il titolo dell'opera; è prassi comune nella denominazione dei documenti TEI, nel caso siano la trascrizione digitale di preesistenti fonti materiali (testi a stampa, manoscritti etc.), riportare il titolo del testo fonte accompagnato dall'indicazione: edizione elettronica ovvero versione elettronica. Non è ad oggi ancora ben chiaro quando sia opportuno parlare di versione e quando di edizione elettronica di un testo digitale, nella pratica i due termini vengono sostanzialmente usati come sinonimi, forse con un po' di presunzione, abbiamo preferito optare per la prima dizione.
Il tag <*author*> racchiude il nome dell'autore dell'opera, si sarà notato che abbiamo distinto mediante il tag <*name*> il cognome dell'autore dal nome di battesimo. I metadati della testata TEI rivestono una notevole importanza nell'indicizzazione bibliografica dei testi digitali, in particolare il cognome dell'autore costituisce spesso una chiave d'accesso all'informazione trasmessa dalla fonte, si pensi alle schede catalografiche delle biblioteche, ordinate prima in base al cognome e poi al nome dello scrittore, si comprenderà bene quindi quanto sia importante questa distinzione tra nome e cognome nell'header TEI. Per operare nella marcatura questo distinguo tra cognome e nome siamo dovuti ricorrere, per così dire, ad un espediente; sia il progetto TIL sia *GriseldaOnLine*[148] adottano nella testata del file il tag <*persname*>, presente esclusivamente nello schema TEI completo, il quale mediante i due elementi figli <*surname*> e <*forename*> consente di distinguere con precisione il nome dal cognome; non essendo questi

[148] Nominalmente i testi di *GriseldaOnLine* sono aderenti alla DTD TEI-Lite tuttavia gli studiosi del portale dell'Università di Bologna, hanno preferito distribuire i propri testi come files "ben formati" delle singole parti di cui sono composti (header, front, body, back); in taluni casi, come in questo per l'appunto, tuttavia i codificatori sono "sconfinati" nella sintassi TEI integrale, ciò comporta che se i files fossero ricostituiti nella loro interezza e sottoposti a validazione sintattica risulterebbero invalidi. In ogni caso si può dire che a parte pochi particolari le opere di *GriseldaOnLine* sono generalmente conformi alla TEI-Lite.

marcatori presenti nella DTD TEI-Lite abbiamo usato il tag <*name*>[149], in precedenza già illustrato, sul modello dei suddetti elementi, specificandone il contenuto grazie all'attributo *type* con valore "*surname*" per il cognome e "*forename*" per il nome.

Mediante l'elemento <*respStmt*>, recitano le guidelines TEI, si fornisce una dichiarazione di responsabilità relativa al responsabile del contenuto intellettuale di un testo, un'edizione, una registrazione, una collana, se gli elementi specifici per autori, curatori etc., non sono sufficienti o non sono adatti. Nella pratica comune questo marcatore viene adoperato principalmente per registrare tutti coloro che hanno contribuito alla redazione dell'opera nella sua versione elettronica. Il ruolo ed il nome dei responsabili dell'edizione elettronica vengono indicati per mezzo della coppia di tag <*resp*> e <*name*>, il primo contiene un'espressione che descrive per esteso la natura della responsabilità, il secondo il nome del responsabile. Nella testata di Baltico si sono registrati due nomi, quello di Salvatore Salzillo, che si è occupato della digitalizzazione e codifica del testo, e quello della Prof. Lucrezia Lorenzini, che ha avuto cura di sovrintendere a tutte le attività.

III.9.3. Dichiarazione dell'edizione

Il secondo elemento figlio di <*fileDesc*>, <*editionStmt*> raggruppa le informazioni relative ad una data edizione di un testo, anche questo tag è normalmente usato per la dichiarazione di edizione relativamente all'edizione elettronica di un opera. Tornando al nostro esempio riportiamo di seguito la corrispondente porzione di codice:

```
<editionStmt>
  <edition>Prima edizione elettronica
  <date value="22-07-2004">22-07-2004</date>
  </edition>
```

[149] Nella codifica è questo l'unico caso in cui il tag <*name*> è adoperato secondo questa modalità.

```
</editionStmt>
```

Come si può vedere numero e data di edizione sono codificati con il tag <*edition*>, si noti pure che la data è normalmente racchiusa all'interno del marcatore <*date*>.

III.9.4. Dichiarazione della dimensione

Proseguendo nell'analisi della testata TEI incontriamo l'elemento <*extent*>, il quale è usato per descrivere le dimensioni approssimative di un testo elettronico in un'opportuna unità di misura. Nel nostro caso:

```
<extent>ca. 372 kb</extent>
```

abbiamo adottato il kilobyte come unità di misura. La dichiarazione della dimensione approssimativa del file inerisce, per così dire, al supporto "materiale" del testo digitale ed ha il suo corrispettivo nell'indicazione del numero delle pagine di un libro, che tipicamente si trova nelle schede catalografiche delle biblioteche.

III.9.5. Dichiarazione della pubblicazione

L'elemento obbligatorio <*publicationStmt*> raggruppa tutte le informazioni relative alla pubblicazione e distribuzione del testo in versione elettronica. Riportiamo subito il codice prodotto per il testo di Collura per poi passare di seguito a commentare i singoli marcatori, in cui si articola il contenuto di questo elemento:

```
<publicationStmt>
 <publisher>Universit&#225; degli Studi di Messina, Facolt&#225; di
Lettere e Filosofia, Dipartimento di Filologia e Linguistica, Cattedra
di Letteratura e Filologia Siciliana</publisher>
   <pubPlace>Messina</pubPlace>
   <availability status="restricted">
   <p>Questo testo &#232; attualmente soggetto a diritto d'autore, la
fruizione nella sua interezza &#232; possibile soltanto presso i
locali del Dipartimento di Filologia e Linguistica, Cattedra di
Letteratura e Filologia Siciliana dell'Universit&#225; degli Studi di
Messina o diversamente previo accordo con il titolare dei diritti. La
```

```
distribuzione di questo file &#232; vietata. Versioni limitate di
pochi capitoli di questo testo vengono distribuite liberamente a scopo
didattico. Ogni copia del sorgente XML del testo deve essere corredato
del file di firma digitale Baltico.xml.sig per mezzo di cui &#232;
possibile verificare l'integrit&#224; e la provenienza del documento.
Il file &#232; firmato con la chiave pubblica del codificatore
<name>Salvatore Salzillo</name>. Validit&#224; ed autenticit&#224;
della chiave possono essere verificate, anche telefonicamente, presso
la Cattedra di Letteratura e Filologia Siciliana, Dipartimento di
Filologia e Linguistica dell'Universit&#225; degli Studi di Messina.
Per la firma elettronica si &#232; usato il software PGP.</p>
   <p> Di seguito si riporta la chiave pubblica di <name>Salvatore
Salzillo</name>:
 <![CDATA[
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 8.0.3 - not licensed for commercial use: www.pgp.com

mQGiBD+7TYARBADVS3ZkAx5V015WBTcL79hRm6aylcLLmlkZ7DFs9/hC9F0FrVwC
Cn47N7BJ2cyCod7TUIrkT/7tXmJ1PlVrzRKa9LHsi8yZHKPLjNQSB9IkiBl776ZW
[. . .]
XzUYITwgY/zuCV1ECLXiix9GTQCgpVYcNQmjlm28LscTLoon3BGHqXc=
=NaBh
-----END PGP PUBLIC KEY BLOCK-----
]]></p>
 </availability>
  <date value="xx-07-2004">07-2004</date>
 </publicationStmt>
```

Come si può notare il contenuto di <*publicationStmt*> si sviluppa in una serie di sotto elementi, che identificano precisamente le diverse informazioni; il primo di questi è <*publisher*>, come si evince anche dal nome, scopo di questo tag è marcare il nome del soggetto, persona od organizzazione, responsabile della

pubblicazione del testo elettronico; nel nostro caso responsabile della pubblicazione è la Cattedra di Filologia e Letteratura Siciliana dell'Università degli Studi di Messina, di cui è titolare la Prof. Lucrezia Lorenzini, responsabile e promotrice del progetto DigiSic.

Il tag <*pubPlace*> contiene il luogo dell'edizione elettronica del documento.

Il successivo marcatore <*availability*> racchiude tutte le informazioni sulla disponibilità del testo. Il romanzo di Matteo Collura, edito nel 1988, è ancora soggetto al diritto d'autore, pertanto una sua distribuzione libera non è possibile, tutte le limitazioni e le possibilità di fruizione elettronica del testo sono illustrate, come si può ben vedere, dentro il tag <*availability*>. In accordo con il titolare dei diritti si è deciso di consentire la distribuzione per scopi didattici di una versione limitata a pochi capitoli del testo codificato di Baltico.

Un'esigenza della distribuzione dei testi digitali è la certificazione della provenienza e dell'integrità del file. Abbiamo adottato una soluzione, per così dire, "artigianale" per cercare di raggiungere questo obiettivo; servendoci del noto software di crittografia PGP[150], disponibile gratuitamente per uso privato ed accademico, abbiamo corredato di una firma digitale il file XML di Baltico, in modo tale da consentire la verifica dell'integrità e della provenienza del file . Il software PGP si basa sui principi della crittografia asimmetrica di cui abbiamo parlato nei capitoli precedenti, il file Baltico.xml.sig, con cui specifichiamo deve sempre essere corredato il file XML di Baltico, contiene la firma digitale prodotta con la chiave privata del codificatore Salvatore Salzillo e il codice hash del file del testo, in virtù di ciò, se il lettore ricorda quanto già detto nei paragrafi dedicati alla firma digitale, nell'ipotesi in cui il file, cui fa riferimento la firma, dovesse subire una seppur minima modifica, la firma digitale risulterebbe non valida, inoltre essendo la chiave privata usata per la firma elettronica nella disponibilità esclusiva del

[150] Creato da Philip Zimmermann, un ingegnere software con oltre 20 anni di esperienza, specializzato in crittografia, sicurezza dei dati, networking e sistemi real-time embedded, PGP (acronimo di Pretty Good Privacy, ossia: "Una privacy niente male") è un programma di crittografia a chiave pubblica, diventato ormai uno standard (tanto che l'insegnamento del suo utilizzo è previsto nel master per la sicurity dell'Università di Milano), che permette di scambiarsi informazioni (anche riservate) in modo estremamente sicuro e veloce.

codificatore, è ovvio che, nel caso in cui la firma coincida con la sua chiave pubblica (la quale è stata inserita anche all'interno del testo codificato) si potrà avere con buon grado di sicurezza certezza della provenienza del testo. La validità della firma digitale, indichiamo in <*availability*>, può essere accertata anche telefonicamente mediante la verifica del *fingerprint*. Quando si genera una coppia di chiavi, questa ha un proprio *fingerprint*[151] ossia una "impronta digitale" che è sempre univoca, sebbene ipoteticamente qualcuno potrebbe sostituire la chiave pubblica del codificatore con la propria, non potrebbe praticamente in alcun modo riprodurne anche il *fingerprint*, grazie a questo sistema è possibile essere certi della provenienza di una chiave. Il *fingerprint* della chiave del codificatore Salvatore Salzillo usata per la firma digitale del file è: EF8C 8AF1 0362 6519 7928 369F 82D4 4849 EA1E A688.

L'adozione del sistema di distribuzione contraddistinto dalla firma digitale dei file, non ha come obiettivo la blindatura delle opere, ma quello di rispondere all'esigenza della certificazione della provenienza e dell'integrità dei testi elettronici; è palese che chiunque potrebbe distribuire i file eliminando i riferimenti alla chiave pubblica e la firma digitale, è d'altra parte vero, sebbene un sistema simile in ambito umanistico rappresenti una novità nel panorama italiano, che se divenisse comune e diffusa questa pratica di distribuzione, il lettore sarebbe portato a diffidare dei testi sprovvisti della firma digitale e pertanto il sistema potrebbe estrinsecare tutta la sua potenza ed efficienza. Quella da noi proposta è una soluzione empirica a basso costo, cha fa uso, adattandoli allo scopo, di strumenti nati non con in mente specificamente le esigenze della distribuzione dei testi digitali; riteniamo che la tecnologia della firma digitale possa fornire importanti risposte anche alle nuove realtà della distribuzione informatica dei testi in ambito umanistico e delle biblioteche digitali, sarà pertanto opportuno che il dibattito scientifico dedichi spazio anche a questi argomenti al fine di trovare soluzioni comuni ed adatte alle nuove necessità poste, anche al mondo delle

[151] Il fingerprint è sostanzialmente un codice di hash della chiave.

"humanae litterae" dall'evoluzione tecnologica. Per conto nostro umilmente poniamo sul tavolo delle proposte questo semplice progetto di distribuzione, che abbiamo appena descritto, basato su un software gratuito[152] estremamente comune e diffuso (PGP).

Proseguendo nell'analisi incontriamo il marcatore <*date*>, con cui si conclude il nostro <*publicationStmt*>, il quale contiene la data di pubblicazione del documento elettronico.

III.9.6. Dichiarazione della fonte

Immediatamente dopo <*publicationStmt*> la DTD TEI prevede due elementi facoltativi

<*seriesStmt*> e <*notesStmt*>, i quali servono a registrare informazioni sull'eventuale collana elettronica cui appartiene una pubblicazione e ad inserire eventuali note; nel nostro lavoro non abbiamo avuto la necessità di adoperare questi elementi.

Chiude il <*fileDesc*> l'elemento obbligatorio <*sourceDesc*>; questo marcatore contiene una descrizione dettagliata della fonte o delle fonti materiali impiegate per realizzare la versione elettronica.

Riportiamo di seguito il codice realizzato per Baltico:

```
<sourceDesc>
  <biblFull>
  <titleStmt>
  <title type="main">Baltico</title>
  <title type="sub">Un epopea siciliana</title>
  <author>Matteo Collura</author>
  </titleStmt>
  <editionStmt>
  <edition>prima edizione</edition>
  </editionStmt>
```

[152] Il software è gratuito soltanto per uso personale ed accademico.

```
  <extent>219 p. 21 cm.</extent>
  <publicationStmt>
  <publisher>Luigi Reverdito Editore</publisher>
  <pubPlace>Trento</pubPlace>
  <date value="xx-05-1988">1988</date>
  <idno type="ISBN">88-342-0229-5</idno>
  </publicationStmt>
  <seriesStmt>
  <title>Biblioteca</title>
  </seriesStmt>
  </biblFull>
</sourceDesc>
```

come si può vedere il riferimento bibliografico al testo fonte è organizzato mediante una descrizione articolata in elementi gerarchicamente ordinati racchiusi dentro il marcatore <*biblFull*>[153], la cui struttura ricalca quella della prima parte della testata TEI; avendo già incontrato in quella sede buona parte dei tag qui presenti ci limitiamo adesso ad illustrare soltanto la funzione di quelli non ancora visti o adottati diversamente. All'interno di <*titleStmt*> i due elementi <*title*> caratterizzati dall'attributo *type* con valore *main* e *sub*, distinguono il titolo dal sottotitolo. Il tag <*extent*>, facendo in questo caso riferimento ad un testo cartaceo, fornisce, sul modello delle schede catalografiche tradizionali, una descrizione fisica dell'opera originale: dimensioni in centimetri e numero delle pagine. <*idno*> contiene un numero, standardizzato o meno, usato per identificare un'unità bibliografica, in questo caso il codice ISBN del libro fonte; l'attributo *type* specifica lo schema o lo standard cui l'elemento fa riferimento.

Il tag <*seriesStmt*>, con cui si conclude la descrizione bibliografica della fonte, racchiude il titolo della collana di appartenenza dell'opera originale.

[153] Le guidelines prevedono che le informazioni sulla fonte materiale possano anche essere riportate mediante un riferimento bibliografico non rigorosamente strutturato, utile soprattutto per i testi più antichi o non derivanti da edizioni a stampa, per mezzo del tag <bibl>.

III.9.7. Descrizione della codifica

Dopo l'elemento <*fileDesc*> la testata TEI prevede l'elemento <*encodingDesc*>, il cui fine è la descrizione delle metodologie e dei principi editoriali impiegati nella trascrizione e codifica del testo. Sebbene l'elemento sia facoltativo il suo uso è fortemente consigliato in quanto illustrando i principi e i metodi che hanno sovrinteso alla trascrizione e codifica elettronica del testo, che come abbiamo visto spesso dipendono notevolmente dalle scelte e dall'arbitrio del codificatore, favorisce l'interoperabilità e consente di interpretare le prassi adottate nella codifica anche a soggetti lontani nel tempo e nello spazio.

Riportiamo di seguito il codice dell'<*encodingDesc*> da noi prodotto per la testata della versione elettronica di Baltico:

```
<encodingDesc>
 <projectDesc>
   <p>Q uesto testo &#232; stato digitalizzato e codificato per il
progetto DigiSic al fine di fornire un'esemplificazione di testo
letterario sottoposto a codifica XML-TEI</p>
 </projectDesc>
 <editorialDecl>
   <p>Il testo elettronico &#232; stato corretto e riportato al testo
cartaceo dell'edizione di riferimento. Tutti i simboli di segnalazione
del
discorso diretto e di citazione sono stati convertiti sotto forma di
entit&#224; numeriche XML. Tutti i trattini di rimando a capo sono
stati eliminati.
I valori standard delle date sono forniti in formato GG-MM-AAAA.</p>
 </editorialDecl>
 <refsDecl>
   <p>L'attributo ID di ogni DIV1 contiene il riferimento canonico per
ogni divisione nella forma XX.YY dove XX &#232; il numero della parte
in numeri romani e YY indica il capitolo in numeri arabi, mentre
```

```
l'attributo N indica soltanto il numero del capitolo all'interno della
parte in cui compare. L'attributo N di ogni div0 specifica il numero
della parte.
I salti di pagina sono codificati mediante elementi PB accompagnati da
un attributo n il cui valore corrisponde al numero della pagina nel
testo cartaceo di riferimento.</p>
 </refsDecl>
 <classDecl>
 <taxonomy id="CDD">
   <bibl>Classificazione Decimale Dewey</bibl>
  </taxonomy>
  </classDecl>
 </encodingDesc>
```

La descrizione della codifica si apre con l'esposizione dettagliata, racchiusa dentro il tag <*projectDesc*>, del fine e dello scopo per cui il testo elettronico è stato codificato.

La spiegazione dei principi e delle pratiche editoriali seguite nella codifica del testo è contenuta invece dentro il marcatore <*editorialDecl*>; il manuale TEI indica che gli argomenti[154] su cui dovrebbe vertere la dichiarazione editoriale sono: le modalità di correzione del testo, il livello di regolarizzazione apportata rispetto alla fonte, il trattamento riservato alle virgolette dell'originale, il trattamento dedicato alla sillabazione di fine riga presente nell'originale, quali eventuali informazioni analitiche o interpretative sono state aggiunte al testo.

L'elemento <*refsDecl*> viene usato per documentare il funzionamento di ogni schema di riferimento standard introdotto nella codifica; nel nostro caso <*refsDecl*> mediante una breve descrizione in prosa illustra il sistema di strutturazione dei blocchi <div#> adottato nella ripartizione del testo; spiega

[154] La DTD TEI completa prevede una serie di elementi specifici per ognuno di questi argomenti.

inoltre il metodo di numerazione dei salti pagina e la loro relazione con l'originale cartaceo.

L'ultimo[155] elemento contenuto nel nostro <*encodingDesc*> è <*classDecl*> il quale raccoglie le definizioni o le fonti di ogni schema di classificazione descrittiva usata in altre parti della testata. In particolare <*taxonomy*> serve a specificare la tipologia usata per la classificazione del testo, sia implicitamente, mediante una citazione bibliografica, sia esplicitamente, con una tassonomia strutturata; l'attributo *id* fornisce un identificatore univoco all'elemento mediante il quale, come vedremo nel prosieguo della trattazione, si permette ad altri elementi di far riferimento al sistema di classificazione definito in <*taxonomy*>.

III.9.8. Descrizione del profilo

L'elemento <*profileDesc*>, che nell'header TEI compare subito dopo <*encodingDesc*>, ha il compito di registrare una serie di informazioni non-bibliografiche che caratterizzano, sotto vari aspetti descrittivi, un testo. Esso dispone di tre componenti opzionali: <*creation*> contenente informazioni relative alla creazione di un testo, <*langUsage*>, il quale descrive le lingue e i dialetti presenti all'interno di un testo, <*textClass*> che racchiude informazioni inerenti la natura o gli argomenti di un documento, nei termini di uno schema di classificazione standard.

Si riporta il codice:

```
<profileDesc>
 <creation>
 <date value="xx-xx-1988">1988</date>
 </creation>
 <langUsage>
 <language id="IT">Italiano</language>
 </langUsage>
```

[155] In realtà vi sono anche altri elementi figli previsti per <encodingDesc>, in questa sede tuttavia ci limiteremo a descrivere solamente quelli da noi adottati nel corso della codifica.

```
<textClass>
<keywords scheme="CDD">
<term>NARRATIVA ITALIANA. 1945 - 1999</term>
</keywords>
<classCode scheme="CDD">853.914</classCode>
</textClass>
</profileDesc>
```

Il primo elemento, non essendovi nel nostro caso nulla di particolare da annotare, si limita a riportare, secondo una prassi consolidata in tutti i progetti di codifica italiani, l'anno di redazione del testo originale. Molto semplice è anche il contenuto del secondo elemento <*langUsage*> che non va oltre la dichiarazione dell'unico linguaggio presente nel testo, l'italiano. Il valore dell'attributo *id* del marcatore <*language*>, dato dal codice a due lettere identificativo del linguaggio secondo lo standard ISO 639, fornisce un identificatore univoco alla dichiarazione di linguaggio cui si può far riferimento in qualsiasi elemento mediante l'attributo globale *lang* per specificare la lingua del testo contenuto nell'elemento.

L'ultimo elemento <*textClass*> classifica il testo sulla base della tassonomia definita nell'elemento <*classDecl*>, cui fa esplicitamente riferimento mediante l'attributo *scheme* dei suoi sotto-componenti <*keywords*> e <*classCode*>. La nostra scelta è caduta sulla **Classificazione Decimale Dewey**, sulla scorta di quanto fatto già da *GriseldaOnLine*[156], in quanto questo sistema di catalogazione gode di buona fortuna in Italia; dal 1958, infatti, la Bibliografia Nazionale Italiana adotta la **CDD** per ordinare le schede che, all'interno dei fascicoli mensili, descrivono le pubblicazioni; inoltre anche il Sistema Bibliotecario Nazionale adopera lo schema, ed appunto dal portale web[157] dell'organizzazione abbiamo ricavato le informazioni che abbiamo inserito nei due tag <*keywords*> e <*classCode*>. Il primo contiene una lista di parole chiave o di espressioni, che

[156] Il progetto TIL ha preferito adottare la Classificazione Decimale Universale.
[157] <www.sbn.it>.

identificano il tema o la natura del documento, in questo caso la classificazione CDD per esteso, il secondo marcatore invece racchiude il codice di catalogazione CDD.

III.9.9. Descrizione della revisione

L'ultimo elemento del <*teiHeader*> <*revisionDesc*> ha lo scopo di fornire informazioni relative alla storia delle modifiche e delle revisioni che il documento elettronico ha subito, ne riportiamo il codice:

```
<revisionDesc>
 <change>
 <date value="xx-xx-2003">2003</date>
 <respStmt>
 <resp>Digitalizzazione e correzione</resp>
 <name>Salvatore Salzillo</name>
 </respStmt>
 <item>digitalizzazione e correzione del testo in base alla edizione
di riferimento
<bibl>
 <author>Matteo Collura</author>
 <title>Baltico : un'epopea siciliana</title>
 <publisher>Luigi Reverdito Editore</publisher>
 <date>1988</date>
 </bibl>
 </item>
 </change>
 <change>
 <date value="xx-xx-2003">2003</date>
 <respStmt>
 <resp>Codifica e revisione</resp>
 <name>Salvatore Salzillo</name>
 </respStmt>
```

```
<item>codifica XML del testo</item>
</change>
</revisionDesc>
     </teiHeader>
```

Ogni cambiamento effettuato sul documento elettronico va documentato mediante gli elementi <*change*>, in cui vengono registrati la data (<*date*>), il nome ed il ruolo del responsabile della modifica (<*respStmt*>) ed una breve descrizione dell'intervento (<*item*>). Sebbene di veri e propri cambiamenti non si tratti, è prassi comune indicare nelle prime due fasi del "diario" delle revisioni, la digitalizzazione e la codifica del testo.

Si conclude con la descrizione di quest'ultimo elemento della testata TEI la parte della trattazione relativa alla codifica XML del testo, nei prossimi capitoli ci dedicheremo all'esposizione delle attività collaterali e complementari alla codifica: la produzione di output per scopi ed esigenze differenti, quali fruizione ed analisi del testo.

III.10. REVISIONE DELLA CODIFICA

Subito dopo la codifica si è proceduto alla revisione del lavoro svolto alla ricerca di errori sia nella trascrizione, sia nella marcatura del testo.

In primo luogo si è verificata la correttezza del contenuto testuale e la sua conformità con la fonte al fine di individuare eventuali errori (battiture accidentali, caratteri sbagliati o mancanti), introdotti nel testo dal programma di OCR e sfuggiti ad una prima analisi oppure frutto di digitazioni involontarie durante l'attività di codifica; si è, pertanto, avuto cura di rileggere tutto il testo "digitale", verificando la perfetta corrispondenza di questo con quello della fonte cartacea.

In secondo luogo si è provveduto alla ben più complicata revisione della marcatura TEI/XML. Della verifica del rispetto dei vincoli sintattici imposti nella DTD come sappiamo si occupa il parser XML; tuttavia, pur nella piena conformità con le regole della DTD può capitare di commettere degli errori nella codifica, si può sbagliare nell'aprire o chiudere un elemento non delimitando perfettamente il testo a cui dovrebbe far riferimento oppure si può semplicemente dimenticare di marcare un elemento o ancora più succedere che, per distrazione o stanchezza, si adoperino elementi o valori di attributo errati. Per cercare di ovviare a tutti gli sbagli, che può aver commesso nell'immissione del markup il novello amanuense elettronico, è opportuno dedicarsi subito dopo la codifica ad un attento e scrupoloso esame della stessa.

Per cercare di facilitarci almeno in parte il duro lavoro di correzione, abbiamo messo in atto alcuni, per così dire, trucchi. Un errore, in cui siamo incorsi più volte, è stato l'aver dimenticato di codificare con l'opportuno tag <*q*> il discorso diretto oppure di aver chiuso il marcatore prima della fine del discorso; per tal motivo, partendo dalla riflessione che ogni discorso diretto è nel nostro testo racchiuso sempre tra doppie virgolette ad angolo (**«**. . .**»**), corrispondenti nella codifica alle entità numeriche "«" e "»" , servendoci della funzione

"trova" dell'editor di testi[158] ci siamo accertati che prima di ogni "«" si trovasse un tag <*q*> e viceversa che dopo ogni "»" si incontrasse il rispettivo tag di chiusura </*q*>.

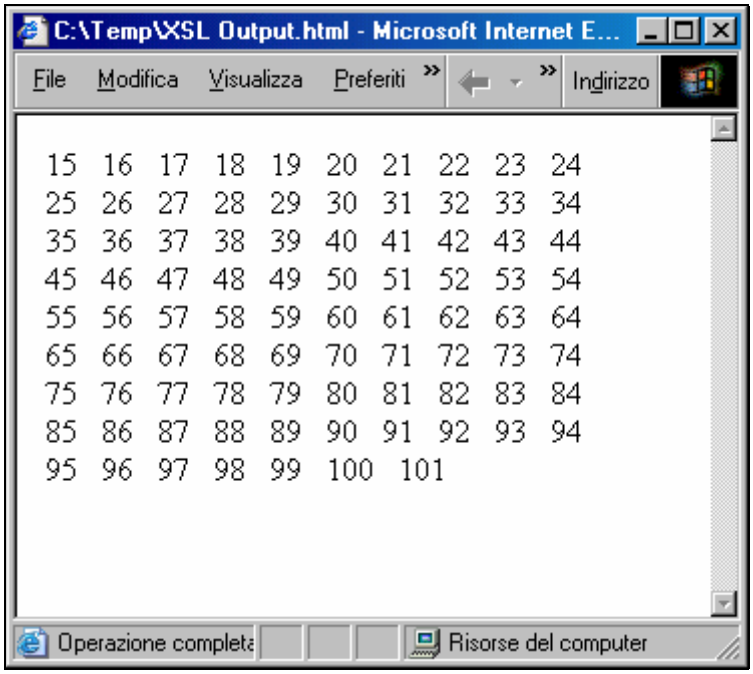
Un altro espediente si è adottato per il controllo dei page break: è naturale che questi elementi si devono susseguire nel file in ordine crescente, abbiamo perciò provveduto a creare un piccolo foglio di stile XSLT[159] che si occupasse di mandare in output in un file HTML il valore dell'attributo *n* di tutti i tag <*pb*> del documento. Riportiamo di seguito il codice del file XSLT in questione:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" version="4.0"/>
<xsl:template match="/">
      <xsl:for-each select="//pb">
            <xsl:choose>
                  <xsl:when test="position() mod 10 = 0">
                  &amp;#160;<xsl:value-of select="@n"/>&amp;#160;<br/>
                  </xsl:when>
                  <xsl:otherwise>
                  &amp;#160;<xsl:value-of select="@n"/>&amp;#160;
                  </xsl:otherwise>
            </xsl:choose>
      </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

[158] La funzione "trova" è comunemente presente in qualunque editor di testi ed ha la funzione, appunto, di individuare nel testo una determinata sequenza di caratteri specificata dall'utente.

[159] Come vedremo meglio in seguito XSLT può essere utilizzato non soltanto per la trasformazione dei files XML in nuovi formati di output più adatti alla fruizione degli stessi, ma anche per operazioni di manipolazione e gestione dei dati contenuti nelle istanze XML.

Ecco, invece, l'output prodotto dal nostro foglio di stile dopo essere stato applicato ad una parte del <*body*> visualizzato nel browser Microsoft Internet Explorer 6.0:



Come si può notare si è ottenuta una sequenza numerica corrispondente ai valori degli attributi *n* dei tag <*pb*>; è stato, quindi facile accorgersi di eventuali errori, numeri mancanti o ripetuti nella successione che avrebbero indicato inequivocabilmente un errore o il mancato inserimento di un marcatore di salto pagina. Tralasciamo di commentare il codice del foglio XSLT in quanto di questo linguaggio parleremo più ampiamente nel prosieguo della trattazione allorché affronteremo la parte relativa alla produzione dei diversi file di output proprio grazie a tale tecnologia; in questa sede diremo soltanto che il foglio effettua mediante l'espressione X-Path "//pb" una ricorsione su tutti gli elementi <*pb*> dell'istanza XML e si occupa di mandare in uscita il valore *n* degli elementi selezionati, per migliorare la leggibilità del file HTML risultante dalla trasformazione abbiamo inserito nella ricorsione una semplicissima elaborazione condizionale[160] che provvede a mandare a capo la sequenza numerica ogni dieci elementi.

[160] L'espressione test="position() mod 10 = 0" verifica che il nodo soggetto ad elaborazione sia il decimo nell'ordine degli elementi selezionati, infatti position() è una funzione X-Path che restituisce un numero che rappresenta la posizione ordinale del nodo entro un elenco di nodi; mentre mod è un

A prescindere comunque, dai sistemi adottati per cercare di automatizzare ed agevolare il lavoro, il grosso dell'attività di revisione è stato effettuato rileggendo con cura tutto il codice del file, prestando la massima attenzione ad accidentali errori o sviste.

operatore aritmetico che fornisce il resto di una divisione di un numero per un altro, in sostanza con position() mod 10 si divide per dieci il numero di posizione del nodo e quando il resto della divisione è zero, ovvero con tutti i multipli di dieci, eventualità verificata dall'espressione = 0 il resto dell'elaborazione condizionale si occupa di aggiungere nell'output un tag
 (che in HTML serve per specificare il ritorno a capo) subito dopo il valore *n* dell'elemento <pb> elaborato, cosa che invece non avviene in tutti gli altri casi (otherwise) ossia quando l'elemento <pb> non è un multiplo di dieci nell'ordine della sequenza degli elementi selezionati.

III.11. LA PRODUZIONE DEGLI OUTPUT

Sebbene XML sia fatto in modo da poter essere intellegibile anche agli esseri umani, è stato in primo luogo concepito per permettere un trattamento ottimale dei dati alle applicazioni informatiche. La codifica elettronica da noi operata è stata portata avanti sulla base del presupposto che il codice sarebbe poi stato sottoposto ad elaborazione automatica dei dati. Si ricorderà quanto gia detto nelle parti precedenti della trattazione sull'importanza della separazione del contenuto dalla sua rappresentazione; mediante la codifica TEI abbiamo sostanzialmente identificato e demarcato la struttura del testo secondo un ben determinato livello descrittivo. Per il trattamento dei dati ci siamo giovati degli strumenti offerti dall'**Extensible Stylesheet Language (XSL)** ed in particolar modo dal suo componente **Extensible Stylesheet Language Trasformations (XSLT)**. Abbiamo già parlato in termini teorici di XSL e della sua genesi nella prima parte di questa relazione, ci accingiamo adesso a mostrare nella pratica, sulla base del lavoro svolto per il romanzo di Collura, come questa tecnologia possa essere impiegata per la creazione di output destinati alla fruizione o addirittura all'analisi del testo.

III.11.1. L'output HTML dell'Header

Intraprendiamo la nostra esposizione con la descrizione della produzione di un output HTML per l'Header del nostro file TEI/XML. In appendice vengono forniti sia il codice completo del foglio di stile XSLT adoperato per la trasformazione, sia il codice HTML prodotto ed una sua rappresentazione[161] grafica. Ove non indicato diversamente si precisa che il processore XSLT da noi adottato è stato *Istant Saxon 6.5.2*.

[161] Html è stato ideato per esser fruito principalmente a video, pertanto in sede di stampa abbiamo cercato di fornire una riproduzione il più fedele possibile dell'output prodotto.

Sostanzialmente XSLT è un linguaggio di estrazione dei dati. Questo significa che l'applicazione, in base alle istruzioni del documento, estrae i dati di un altro documento, nel nostro caso il file XML di Baltico, e li posiziona all'interno di una nuova struttura, nello specifico quella del file HTML con i dati dell'HEADER TEI.

Passando all'analisi del codice salta subito all'occhio che un foglio di stile XSLT è anche un file XML ben formato. Il documento si apre con la consueta dichiarazione XML

`<?xml version="1.0" encoding="UTF-8"?>` cui segue l'elemento radice del foglio XSLT `<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">`, subito dopo incontriamo `<xsl:output method="html" encoding="iso-8859-1" version="4.01" doctype-public="-//W3C//DTD HTML 4.01 Transitional//EN"/>` con cui si specifica il formato dell'output, in questo caso HTML 4.01. tralasciando i due elementi

```
<xsl:strip-space elements="*"/>
<xsl:preserve-space elements="author"/>
```

entriamo nel vivo del foglio di stile; il cuore della sintassi XSLT è costituito dalle dichiarazioni di modello (*template*), mediante le quali si mettono in corrispondenza i nodi, ovvero gli elementi, del sorgente XML con le istruzioni definite nel foglio XSLT. Tornando all'esempio concreto vediamo che il primo modello definito è:

```
<xsl:template match="/">
 <html>
 <head>
 <title>
 <xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/> -
<xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
[FRONTESPIZIO]</title>
 </head>
 <body>
 <hr/>
```

```
 <h1>
 <xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
 <hr/>
 </h1>
 <xsl:apply-templates/>
 </body>
 </html>
</xsl:template>
```

Questo primo modello, mediante l'attributo *match*, mette in relazione il nodo radice[162] del file sottoposto ad elaborazione, nel nostro caso[163] <*TEI*.2>… </*TEI*.2>, con le istruzioni definite nel foglio di stile. Ecco nel concreto cosa avviene: il processore dopo aver trovato corrispondenza nel file sorgente con il nodo identificato nel modello, ossia <*TEI*.2>, manda in output le istruzioni del modello: in pratica comincia a creare un file HTML scrivendo gli opportuni tag: <*html*> <*head*> <*title*>, arrivato a questo punto l'espressione `<xsl:value-of select="TEI.2/teiHeader/fileDesc/titleStmt/author`[164]`"/>` ordina al processore di mandare in output, di scrivere, il contenuto dell'elemento identificato dall'espressione XPath (`TEI.2/teiHeader/fileDesc/titleStmt/author`), la quale può essere così tradotta: "il contenuto di <*author*> figlio di <*titleStmt*> il quale è figlio di <*fileDesc*> il quale è figlio di <*teiHeader*> il quale è figlio di <*TEI*.2>", il lettore forse ricorderà che questo elemento così si presenta nella testata di Baltico

`<author><name type="forename">Matteo</name>`

`<name type="surname">Collura</name></author>`, come vediamo il tag <*author*> ha due elementi figli <*name*>, tuttavia il comportamento di *value-of* prevede che venga mandato in output il valore di stringa (in parole povere il

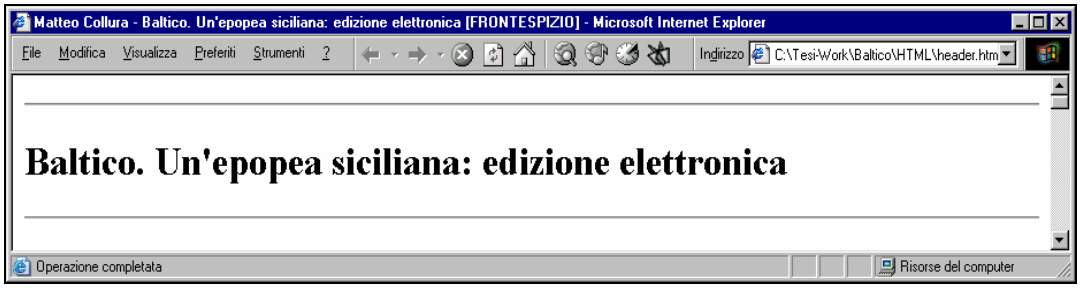
[162] In Xpath la radice di un documento XML è rappresentata dallo slash (/).
[163] Si sta ovviamente facendo riferimento al file XML di Baltico.
[164] Quello qui illustrato è un percorso di posizione di tipo assoluto, ossia viene descritto tutto il percorso da seguire lungo l'albero del documento a partire dall'elemento radice, il più delle volte tuttavia si farà riferimento a percorsi di posizione relativi al nodo contesto dell'elaborazione.

contenuto testuale) dell'elemento selezionato e dei suoi figli, in questo caso <*name*>. Procedendo nell'analisi del foglio di stile osserviamo che il modello prevede che si inserisca un trattino e poi, allo stesso modo di quanto già fatto con il nome dell'autore, si trasmetta il valore del tag <*title*> del <*titleStmt*>, contenente il titolo dell'opera. Dopo queste operazioni si prescrive al template di scrivere [FRONTESPIZIO], e di seguito </*title*> </*head*> <*body*> <*hr*/> <*h1*>, marcatori con cui si chiude la testata ed inizia il contenuto del file HTML, dopo si manda nuovamente in output il titolo dell'opera sempre grazie all'ausilio di espressioni *xsl:value-of*, dopodiché l'elaborazione prosegue scrivendo <*hr*/></*h1*>, finché non incontra `<xsl:apply-templates/>` ed allora si arresta temporaneamente; `<xsl:apply-templates/>` impone al processore di bloccarsi e chiamare i modelli per gli elementi figli dell'elemento selezionato, solo dopo aver effettuato queste operazioni il modello torna attivo ed esegue le restanti operazioni, nello specifico scrivere i tag di chiusura del file HTML </*body*> </*html*>. Ecco l'output prodotto dal modello appena illustrato fino al suo arresto al momento in cui incontra `<xsl:apply-templates/>`:

```
<html>
 <head>
 <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 <title>Matteo Collura - Baltico. Un'epopea siciliana: edizione
elettronica [FRONTESPIZIO]</title>
  </head>
  <body>
  <hr>
  <h1>Baltico. Un'epopea siciliana: edizione elettronica</h1>
  <hr>
```

Come si può vedere all'interno del tag <*title*> sono stati inseriti i valori dei tag <*author*> e <*title*> della testata TEI, lo stesso è stato fatto anche per il marcatore

<*h1*> che apre il corpo del file HTML. Di seguito mostriamo come appare nel browser Internet Explorer 6.0 questo output HTML:



Proseguendo nell'analisi del foglio di stile il secondo modello che incontriamo è:

```
<xsl:template match="teiHeader">
   <xsl:apply-templates/>
</xsl:template>
```

Il contenuto di questo template si limita a dire al processore di chiamare gli altri modelli per gli elementi figli del nodo corrente; si ricorderà che in effetti l'elemento <*teiHeader*> non è altro che un contenitore per tutta una serie di sotto elementi contenenti i dati veri e propri. Una impostazione analoga al modello adottato per il <*teiHeader*> la ritroviamo nel modello del suo primo figlio <*fileDesc*> ed a sua volta per il figlio di questi <*titleStmt*>: entrambi, infatti, sono elementi che, come sappiamo, racchiudono un gruppo di tag che marcano informazioni strutturate, pertanto i modelli che interverranno saranno quelli facenti riferimento ai loro sottocomponenti di livello più basso.

Una precisazione occorre fare in merito al comportamento di `<xsl:apply-templates/>`, questo elemento come abbiamo visto viene usato per invocare i modelli per gli elementi figli dell'elemento contesto, tuttavia come si potrà notare proseguendo nell'analisi del foglio XSLT, in taluni casi questo comando è presente anche in modelli i quali fanno riferimento ad elementi nel sorgente il cui contenuto è costituito esclusivamente da testo e che si trovano al livello più basso dell'albero del documento XML, ci si domanderà quindi il perché dell'adozione di `<xsl:apply-templates/>` in questi contesti. In realtà, oltre ai modelli esplicitamente dichiarati nel foglio di stile XSLT, il processore prevede dei modelli, per così dire impliciti,

definiti in un foglio di stile di default che interviene quando si incontra un elemento non messo in corrispondenza con un template[165] definito nel foglio di stile principale. Sulla base dei comportamenti di default, allorché il processore XSLT raggiunge il contenuto di un elemento per cui non è stato definito un modello, viene mandato in output il contenuto dell'elemento, in sostanza è come se fosse presente il comando `<xsl:value-of select="."/>`[166].

Riprendendo l'analisi del nostro foglio di stile incontriamo i modelli per gli elementi <*title*> ed <*author*> del <*titleStmt*> del <*fileDesc*>:

```
     <xsl:template match="fileDesc/titleStmt/title">
 <p><b>Titolo dell'Opera</b>:  <xsl:apply-templates/></p>
     </xsl:template>
     <xsl:template match="fileDesc/titleStmt/author">
 <p><b>Autore</b>:  <xsl:apply-templates/></p>
     </xsl:template>
```

come vediamo questi template per ognuno dei due elementi creano un paragrafo (delimitato dai tag <*p*>…</*p*>) poi scrivono del testo, rispettivamente "Titolo dell'opera" ed "Autore", racchiudendolo dentro tag <*b*>[167] in modo che venga reso in grassetto nell'output HTML ed infine invocano eventuali altri modelli mediante `<xsl:apply-templates/>`, tuttavia non essendovi modelli definiti per i figli di questi elementi interviene il foglio di stile di default che manda in output il contenuto del nodo corrente.

La medesima impostazione dei modelli appena illustrati la ritroviamo nei template definiti per gli altri elementi dell'Header TEI presenti nel resto del foglio di stile, di seguito, pertanto, descriveremo soltanto quei modelli che in qualche modo si discostano da questa organizzazione di base.

---

[165] Si tenga presente che nella trattazione i termini modello e template vengono utilizzati come sinonimi.

[166] In XPath il punto (.) è l'espressione che fa riferimento al nodo contesto corrente.

[167] In questo documento si è fatto ricorso in più casi dei marcatori <b> per il grassetto ed <i> per il corsivo, l'uso di questi tag nell'HTML 4 è deprecato dal W3c Consortium a favore dei fogli di stile CSS, tuttavia in questa sede abbiamo preferito adoperarli al fine di non gravare eccessivamente la leggibilità del sorgente XSLT con ulteriori elementi ben più prolissi. Nostro obiettivo primario è stato l'esplicazione dei meccanismi di trasformazione di XSL più che la produzione di un output particolarmente efficiente.

Hanno richiesto una programmazione leggermente diversa i template per le dichiarazioni di responsabilità, che nel sorgente XML sono racchiuse dentro il tag <*respStmt*>. Si rammenterà che dentro questo marcatore il ruolo ed il nome dei responsabili dell'edizione elettronica vengono indicati per mezzo della coppia di tag <*resp*> e <*name*>, il primo contenente un'espressione che descrive per esteso la natura della responsabilità, il secondo il nome del responsabile. Nell'Header del file XML di Baltico queste coppie di elementi ricorrono due volte, la prima per registrare la responsabilità della digitalizzazione e codifica nella persona di Salvatore Salzillo, la seconda invece per la dichiarazione di responsabilità della supervisione nella figura della prof. Lucrezia Lorenzini. Nell'output HTML desideravamo che ognuna delle due dichiarazioni di responsabilità (ossia il contenuto dei due elementi <*resp*> e <*name*>) fosse inserita dentro un paragrafo distinto, tuttavia non essendo, dentro <*respStmt*>, racchiuse le coppie di tag <*resp*> e <*name*> dentro un ulteriore elemento contenitore[168], siamo dovuti ricorrere, per così dire, ad un espediente per l'ottenimento di questo tipo di struttura HTML. Riportiamo di seguito il codice dei tre modelli in questione:

```
<xsl:template match="fileDesc/titleStmt/respStmt">
      <xsl:apply-templates/>
</xsl:template>
<xsl:template match="fileDesc/titleStmt/respStmt/resp">
      <xsl:text disable-output-escaping="yes"> &lt;p&gt;</xsl:text>
     <b><xsl:apply-templates/></b>:  
</xsl:template>
<xsl:template match="fileDesc/titleStmt/respStmt/name">
      <xsl:apply-templates/>
     <xsl:text disable-output-escaping="yes"> &lt;/p&gt;</xsl:text>
</xsl:template>
```

[168] Applicare la consueta sintassi `<p><xsl:apply-templates/></p>` al modello di <*respStmt*> avrebbe fatto si che nel file HTML di output si producesse un unico paragrafo con il contenuto delle due dichiarazioni di responsabilità, non essendo queste distinte da un ulteriore elemento contenitore, che avrebbe potuto quindi permettere la creazione di un modello per le singole dichiarazioni di responsabilità.

Il modello del primo elemento <*respStmt*>[169] si limita ad invocare i modelli degli elementi figli; il secondo modello, per l'elemento <*resp*>, prima manda in output la sequenza di caratteri <*p*>, che sappiamo corrispondere al tag HTML di apertura paragrafo, grazie all'espressione:

```
<xsl:text disable-output-escaping="yes">&lt;p&gt;</xsl:text>
```[170]

successivamente mediante `<xsl:apply-templates/>` scrive nell'output il valore dell'elemento contesto (ossia <*resp*>) racchiudendolo dentro tag <*b*> … </*b*> e facendolo seguire dal segno di due punti ed uno spazio bianco.

Il template dell'elemento <*name*> invece subito dopo aver trasmesso nell'output il proprio valore scrive il tag HTML di chiusura paragrafo </p> sempre grazie all'espressione:

```
<xsl:text disable-output-escaping="yes"></p></xsl:text>.
```

Grazie alla tecnica appena illustrata siamo riusciti ad aprire il paragrafo HTML prima di scrivere il valore del tag <*resp*> ed a chiuderlo subito dopo che nell'output si è inserito il valore di <*name*>, ottenendo così il risultato da noi voluto di divisione in paragrafi separati delle dichiarazioni di responsabilità[171]. Mostriamo di seguito la porzione di codice HTML prodotto dai modelli appena illustrati:

```
<p><b>Digitalizzazione e codifica elettronica a cura di</b>:  
```[172]

```
Salvatore Salzillo</p>
```

---

[169] Nella sintassi XPath per l'invocazione di questi modelli abbiamo adoperato un percorso di posizione che parte da <fileDesc> al fine di evitare ambiguità nella programmazione, infatti la DTD TEI prevede, anche in contesti diversi da questi al momento da noi descritti, la presenza di tag dal nome <respStmt> <resp> e <name>. Nella pratica con tali percorsi di posizione si è specificato che i modelli in questione devono essere applicati esclusivamente agli elementi <respStmt> (e ai suoi figli <resp> e <name>) figli di <titleStmt> e di <fileDesc>.

[170] L'elemento <xsl:text> viene usato per inserire testo nell'output. L'attributo disable-output-escaping determina se il valore debba essere sottoposto o meno ad "escape" nell'output, ossia se le entità XML debbano essere sostituite con le rispettive entità del formato di output o meno. Nel nostro caso le due entità XML "<" ed ">" nel file HTML non vengono rimpiazzate dalle corrispettive entità HTML (che per altro hanno lo stesso nome) ma vengono mandate in output così come sono, ovvero come i due caratteri "<" e ">".

[171] È bene precisare che affinché questi modelli producano il risultato da noi illustrato è indispensabile che nel sorgente XML le dichiarazioni di responsabilità si presentino sempre come sequenze di elementi <resp> e <name>, ordine che non è espressamente specificato nella DTD TEI e che quindi è compito del codificatore eventualmente rispettare.

[172] Si noti come non essendo stato impostato per l'entità numerica XML ** ** lo sganciamento dell'output (come invece fatto per produrre i tag di paragrafo) questa venga sostituita nell'output dalla rispettiva entità HTML ** **.

```
<p><b>Supervisione</b>:   prof. Lucrezia Lorenzini </p>
```

Precisiamo che abbiamo adottato il comando `<xsl:text disable-output-escaping="yes">` per creare i paragrafi esclusivamente per mantenere la "buona formazione"[173] XML del nostro foglio di stile XSLT.

Anche l'elemento <*availability*> ha richiesto una cura particolare nella programmazione del suo template. Il lettore forse ricorderà la nostra decisione di adoperare le tecniche di firma digitale per l'attestazione della provenienza dell'originalità e dell'integrità dei sorgenti XML di opere codificate elettronicamente e rammenterà probabilmente la decisione di inserire la chiave pubblica del codificatore all'interno del file di Baltico, che appunto è stata posta nell'Header all'interno del tag <*availability*>. Avendo noi preferito non inserire nell'output HTML dell'Header, sotto forma di testo, il codice della chiave pubblica (cosa comunque possibile) così come fatto nel file XML, ma piuttosto come collegamento ad un file esterno, abbiamo creato due modelli, il cui codice riportiamo di seguito, uno per l'elemento <*availability*> ed uno per il suo ultimo paragrafo:

```
<xsl:template match="publicationStmt/availability">
      <p><b>Copyright</b>:  <xsl:apply-templates/><br/>
            <a href="chiave_codificatore.asc">Download chiave pubblica
      per la verifica della firma digitale</a>
       </p>
</xsl:template>
<xsl:template match="availability/p[position() = last()]"/>
```

Il primo modello, sulla scorta del comportamento degli altri template, invoca mediante `<xsl:apply-templates/>` i modelli per i figli dell'elemento contesto <*availability*>, che nello specifico nel nostro file XML è costituito da due paragrafi (<*p*>), il primo contenente le dichiarazioni di disponibilità del documento ed il secondo la chiave pubblica del codificatore. Per il primo paragrafo il processore

[173] Si ricordi che un file XSLT è anche una istanza XML e pertanto deve essere almeno "ben formata" prima ancora che "valida" (i concetti di buona formazione e validità XML sono stati chiariti nella prima parte di questa relazione). Se avessimo scritto semplicemente nel primo modello e nel secondo, avremmo compromesso la buona formazione del file in quanto avremmo avuto dei marcatori orfani o del tag di chiusura o di quello di apertura.

non trova corrispondenza con alcun modello pertanto sulla base del foglio di stile di default manda in output il contenuto del primo elemento <*p*>, tuttavia per il secondo <*p*> è stato precisato un modello: `<xsl:template match="availability/p[position() = last()]"/>`, questo template trova corrispondenza con l'ultimo paragrafo (<*p*>) di <*availability*>, come si può notare non ha alcun contenuto, infatti questo template indica al processore di non compiere alcuna azione allorché l'elaborazione giunge all'elemento selezionato, in pratica ordina di ignorarlo. L'espressione XPath "`availability/p[position() = last()]`" selezionerà sempre l'ultimo paragrafo di <*availability*>, indipendentemente dal numero di paragrafi presenti nell'elemento, pertanto il codificatore per escludere dall'elaborazione il paragrafo con la sua chiave pubblica, dovrà soltanto aver cura di porla nell'ultimo paragrafo[174].

Un template analogo a questo appena illustrato è quello posto alla fine del documento con cui si fa in modo che l'elaborazione si limiti al TEI Header e non proceda nel resto del file XML: `<xsl:template match="text"/>`.

Nella trasformazione dei documenti XML il linguaggio XSLT dà anche la possibilità di compiere delle elaborazioni condizionali sui dati, ossia consente di stabilire che determinate operazioni vengano compiute soltanto al verificarsi di specifiche condizioni. Anche nel foglio di stile che stiamo esaminando è stata inserita una semplice elaborazione condizionale, infatti il modello per tutti i figli dell'elemento <*bibl*>[175], contenente la citazione bibliografica del testo fonte all'interno del diario di codifica (<*revisionDesc*>), esegue comandi differenti sulla base del verificarsi di certe condizioni, di seguito ne riportiamo il codice:

```
<xsl:template match="revisionDesc/change/item/bibl/*">
   <xsl:choose>
```

[174] La coppia di modelli definiti per <availability> richiede che dentro l'elemento siano presenti almeno due paragrafi, infatti se vi fosse un solo <p> questo sarebbe sia il primo sia l'ultimo, e quindi attiverebbe il template che blocca l'elaborazione; pertanto ove il codificatore decidesse di non adottare la firma digitale del documento oppure di racchiudere tutto dentro un unico paragrafo, dovrebbe provvedere o ad inserire un secondo paragrafo, anche vuoto, oppure ad eliminare il template per l'ultimo <p> dal foglio di stile.

[175] Tale elemento contiene una citazione bibliografica non strutturata, i cui componenti possono, o meno, essere marcati esplicitamente.

```
      <xsl:when test="position() = 1">:  <xsl:apply-
templates/>,</xsl:when>
      <xsl:when test="position() != last()">
 <xsl:apply-templates/>,</xsl:when>
      <xsl:otherwise>  <xsl:apply-templates/>.</xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

Come si può rilevare l'espressione XPath seleziona tutti gli elementi[176] figli di <bibl>, il template mediante il comando <xsl:choose> sulla base del verificarsi di condizioni diverse prescrive al processore di eseguire azioni differenti; nello specifico quando il nodo elaborato è il primo (la condizione è data dall'espressione `test="position() = 1"`) il processore scrive nell'output il carattere due punti e successivamente due spazi bianchi[177], dopo aver mandato in output il contenuto dell'elemento (`<xsl:apply-templates/>`) scrive una virgola. Allorché invece l'elemento selezionato **non è** l'ultimo (`test="position() != last()"`), analogamente al comportamento precedente scrive due spazi bianchi, in questo caso non preceduti dai due punti, ed in fine dopo il contenuto dell'elemento pone una virgola. L'ultimo comportamento (`<xsl:otherwise>`) stabilisce che in tutti gli altri casi, ed in pratica soltanto quando l'elemento selezionato è l'ultimo, si mandino in output due spazi bianchi ed il contenuto dell'elemento seguito dal punto. Nel file XML di Baltico così si presenta la citazione bibliografica del testo di riferimento contenuta nell'elemento deputato alla descrizione della revisione:

```
<bibl>
  <author>Matteo Collura</author>
  <title>Baltico : un'epopea siciliana</title>
  <publisher>Luigi Reverdito Editore</publisher>
  <date>1988</date>
```

[176] L'asterisco [*] è un carattere jolly nella sintassi XPath che consente la selezione di tutti i nodi di elemento nel contesto corrente.
[177] Il secondo spazio bianco è definito mediante un'entità numerica XML che poi nell'output sarà sostituita dalla corrispondente entità HTML.

```
</bibl>
```

Ecco, invece, come tale citazione viene riprodotta nel browser dopo la trasformazione con XSLT:

digitalizzazione e correzione del testo in base alla edizione di riferimento[178]: Matteo Collura, Baltico: un'epopea siciliana, Luigi Reverdito Editore, 1988.

Sebbene la finalità del foglio di stile appena descritto sia prevalentemente la dimostrazione delle potenzialità di XSLT nel trattamento dei testi, piuttosto che un suo uso pratico, ci teniamo a sottolineare come con pochi eventuali aggiustamenti[179] sarebbe possibile applicare il nostro foglio di stile ad innumerevoli testi codificati secondo le regole della TEI-Lite, è questo infatti uno dei pregi principali della codifica dichiarativa dei testi. Concludendo possiamo dire che in queste pagine abbiamo dato una prima dimostrazione pratica di alcune delle potenzialità che si hanno nella rappresentazione di dati identificati funzionalmente mediante una marcatura di tipo dichiarativo. Come si ricorderà al momento della codifica XML-TEI abbiamo racchiuso le varie porzioni di testo entro tag che hanno specificato cosa identificassero, cosa fossero, entro una data struttura funzionale i diversi blocchi testuali. Potremmo dire che con la codifica abbiamo espresso cosa un tag contenesse e solo ora grazie ad XSLT stabiliamo come quel contenuto vada rappresentato; ma la grande novità sta nel fatto che il contenuto non è più vincolato ad un'unica rappresentazione è bensì pronto ad infinite raffigurazioni atte a soddisfare le esigenze più diverse; nelle pagine che seguiranno vedremo come partendo sempre dallo stesso nucleo di dati, pur nel nostro piccolo, siamo stati in grado di produrre output in formati rappresentazionali per gli scopi e gli usi più vari.

[178] L'output prodotto dal modello appena descritto inizia effettivamente dopo la parola riferimento.
[179] Il foglio appena illustrato è stato programmato sulla base del file XML di Baltico, per essere veramente universale sarebbe necessario prevedere dei templates per quegli elementi che pur previsti dalla DTD TEI-Lite non sono stati adoperati nella codifica del libro di Collura.

Appendice 1

FOGLIO DI STILE XSLT PER LA PRODUZIONE DI UN OUTPUT HTML A PARTIRE DAI DATI DELLA TESTATA (HEADER) TEI

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="iso-8859-1" version="4.01"
doctype-public="-//W3C//DTD HTML 4.01 Transitional//EN"/>
  <xsl:strip-space elements="*"/>
  <xsl:preserve-space elements="author"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/>
          -
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
          [FRONTESPIZIO]
        </title>
      </head>
      <body>
        <hr/>
        <h1>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/></h1>
        <hr/>
        <xsl:apply-templates/>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="teiHeader"><xsl:apply-
templates/></xsl:template>
  <xsl:template match="filedesc"><xsl:apply-templates/></xsl:template>
  <xsl:template match="titleStmt"><xsl:apply-
templates/></xsl:template>
  <xsl:template match="fileDesc/titleStmt/title">
    <p><b>Titolo dell'Opera</b> :  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="fileDesc/titleStmt/author">
    <p><b>Autore</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="fileDesc/titleStmt/respStmt">
      <xsl:apply-templates/>
  </xsl:template>
  <!-- questa sintassi è necessaria per mantenere il file ben formato
-->
  <xsl:template match="fileDesc/titleStmt/respStmt/resp">
    <xsl:text disable-output-escaping="yes">&lt;p&gt;</xsl:text>
    <b><xsl:apply-templates/></b>:  
  </xsl:template>
  <xsl:template match="fileDesc/titleStmt/respStmt/name">
    <xsl:apply-templates/>
    <xsl:text disable-output-escaping="yes">&lt;/p&gt;</xsl:text>
  </xsl:template>
```

```
  <xsl:template match="editionStmt">
    <p><b>Edizione</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="extent">
    <p><b>Dimensioni file</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="publicationStmt"><xsl:apply-
templates/></xsl:template>
  <xsl:template match="publicationStmt"><xsl:apply-
templates/></xsl:template>

<xsl:template match="publicationStmt/publisher">
    <p><b>Pubblicazione e Distribuzione</b>:  <xsl:apply-
templates/></p>
  </xsl:template>
  <xsl:template match="publicationStmt/pubPlace">
    <p><b>Luogo di pubblicazione</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="publicationStmt/availability">
    <p>
     <b>Copyright</b>:  <xsl:apply-templates/>
     <!-- inserire collegamento alla firma digitale -->
    </p>
  </xsl:template>
  <!--modello x evitare di mandare in output la firma digitale -->
  <xsl:template match="availability/p[position() = last()]"/>
  <xsl:template match="publicationStmt/date">
    <p><b>Data di pubblicazione</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="sourceDesc">
    <h2>Descrizione della fonte</h2>
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull">
    <p><b>Testo di Riferimento</b>:  <br/><xsl:apply-
templates/></p>
  </xsl:template>
  <xsl:template
match="sourceDesc/biblFull/titleStmt/title[@type='main']">
    <i>TITOLO</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template
match="sourceDesc/biblFull/titleStmt/title[@type='sub']">
    <i>Sottotitolo</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/titleStmt/author">
    <i>Autore</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/editionStmt">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/editionStmt/edition">
    <i>Edizione</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/extent">
    <i>Dimensioni</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/publicationStmt">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/publicationStmt/publisher">
```

```
    <i>Editore</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/publicationStmt/pubPlace">
    <i>Luogo di pubblicazione</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/publicationStmt/date">
    <i>Anno di pubblicazione</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="sourceDesc/biblFull/publicationStmt/idno">
    <i><xsl:value-of select="@type"/></i>:  <xsl:apply-
templates/><br/>
  </xsl:template>

  <xsl:template match="sourceDesc/biblFull/seriesStmt">
    <i>Collana</i>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="encodingDesc">
    <h2>Descrizione della codifica</h2>
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="projectDesc">
    <p><b>Descrizione del progetto</b>:  <xsl:apply-
templates/></p>
  </xsl:template>
  <xsl:template match="editorialDecl">
   <p><b>Descrizione dei principi di codifica</b>:  <xsl:apply-
templates/></p>
  </xsl:template>
  <xsl:template match="refsDecl">
    <p><b>Descrizione dei principi di codifica inerenti il sistema di
riferimento</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="classDecl">
    <p><b>Classificazione</b>:  <xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="profileDesc">
    <p><h2>Descrizione del profilo:</h2><xsl:apply-templates/></p>
  </xsl:template>
  <xsl:template match="profileDesc/creation">
    <b>Data di creazione</b>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="profileDesc/langUsage">
    <b>Lingua</b>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="profileDesc/textClass"><xsl:apply-
templates/></xsl:template>
  <xsl:template match="profileDesc/textClass/keywords">
    <b>Categoria</b>:  <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="profileDesc/textClass/classCode ">
    <b><xsl:value-of select="@scheme"/></b>:  <xsl:apply-
templates/><br/>
  </xsl:template>
  <xsl:template match="revisionDesc">
    <p>
      <h2>Descrizione della revisione</h2>
      <xsl:apply-templates/>
    </p>
  </xsl:template>
  <xsl:template match="revisionDesc/change/date">
    <b>Data</b>:  <xsl:apply-templates/><br/>
```

```
  </xsl:template>
  <xsl:template match="revisionDesc/change/respStmt">
    <xsl:apply-templates/><br/>
  </xsl:template>
  <xsl:template match="revisionDesc/change/respStmt/resp">
    <b><xsl:apply-templates/></b>:  
  </xsl:template>
  <xsl:template match="revisionDesc/change/respStmt/name">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="revisionDesc/change/item">
    <b>Descrizione</b>:  <xsl:apply-templates/><br/><br/>
  </xsl:template>

  <xsl:template match="revisionDesc/change/item/bibl/*">
    <!-- elaborazione condizionale per inserire i due punti ed il
punto finale -->
    <xsl:choose>
      <xsl:when test="position() = 1">
        :  <xsl:apply-templates/>,
      </xsl:when>
      <xsl:when test="position() != last()">
         <xsl:apply-templates/>,
      </xsl:when>
      <xsl:otherwise>
         <xsl:apply-templates/>.
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <xsl:template match="text"/>
</xsl:stylesheet>
```

Appendice 2

OUTPUT HTML PRODOTTO DAL FOGLIO XSLT OPERANTE SUI DATI DELLA TESTATA TEI

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
      <title>Matteo Collura - Baltico. Un'epopea siciliana: edizione
elettronica [FRONTESPIZIO]</title>
   </head>
   <body>
      <hr>
      <h1>Baltico. Un'epopea siciliana: edizione elettronica</h1>
      <hr>
      <p><b>Titolo dell'Opera</b>:  Baltico. Un'epopea siciliana:
edizione elettronica</p>
      <p><b>Autore</b>:  Matteo Collura
      </p>
<p><b>Digitalizzazione e codifica elettronica a cura di</b>:  
Salvatore Salzillo </p>
<p><b>Supervisione</b>:   Prof. Lucrezia Lorenzini </p>
<p><b>Edizione</b>:  Prima edizione elettronica 22-07-2004</p>
<p><b>Dimensioni file</b>:  ca. 372 kb </p>
<p><b>Pubblicazione e Distribuzione</b>:  Universit&aacute; degli
Studi di Messina, Facolt&aacute; di Lettere e Filosofia, Dipartimento
di Filologia e Linguistica, Cattedra di Letteratura e Filologia
Siciliana</p>
<p><b>Luogo di pubblicazione</b>:  Messina</p>
<p><b>Copyright</b>:  Questo testo &egrave; attualmente soggetto
a diritto d'autore, la fruizione nella sua interezza &egrave;
possibile soltanto presso i locali del Dipartimento di Filologia e
Linguistica, Cattedra di Letteratura e Filologia Siciliana
dell'Universit&aacute; degli Studi di Messina o diversamente previo
accordo con il titolare dei diritti. La distribuzione di questo file
&egrave; vietata. Versioni limitate  di pochi capitoli di questo testo
vengono distribuite liberamente a scopo didattico. Ogni copia del
sorgente XML del testo        deve essere corredato del file di firma
digitale Baltico.xml.sig per mezzo di cui &egrave; possibile
verificare l'integrit&agrave; e la provenienza del documento. Il file
&egrave; firmato con la chiave pubblica del codificatore Salvatore
Salzillo. Validit&agrave; ed autenticit&agrave; della chiave possono
essere verificate, anche telefonicamente, presso la Cattedra di
Letteratura e Filologia Siciliana, Dipartimento di Filologia e
Linguistica dell'Universit&aacute; degli Studi di Messina. Per la
firma elettronica si &egrave; usato il software PGP. </p>
<p><b>Data di pubblicazione</b>:  07-2004</p>
<h2>Descrizione della fonte</h2>
<p><b>Testo di Riferimento</b>: <br>
<i>TITOLO</i>:  Baltico<br>
<i>Sottotitolo</i>:  Un epopea siciliana<br>
<i>Autore</i>:  Matteo Collura<br>
<i>Edizione</i>:  prima edizione<br>
<i>Dimensioni</i>:  219 p. 21 cm.<br>
<i>Editore</i>:  Luigi Reverdito Editore<br>
```

Luogo di pubblicazione: Trento
Anno di pubblicazione: 1988
ISBN: 88-342-0229-5
Collana: Biblioteca
<h2>Descrizione della codifica</h2>
Descrizione del progetto: Questo testo è stato digitalizzato e codificato per il progetto DigiSic, al fine di fornire un'esemplificazione di testo letterario sottoposto a codifica elettronica
Descrizione dei principi di codifica: Il testo elettronico è stato corretto e riportato al testo cartaceo dell'edizione di riferimento. Tutti i simboli di segnalazione del discorso diretto e di citazione sono stati convertiti sotto forma di entità numeriche XML. Tutti i trattini di rimando a capo sono stati eliminati.
I valori standard delle date sono forniti in formato GG-MM-AAAA.
Descrizione dei principi di codifica inerenti il sistema di riferimento: L'attributo ID di ogni DIV1 contiene il riferimento canonico per ogni divisione nella forma XX.YY dove XX è il numero della parte in numeri romani e YY indica il capitolo in numeri arabi, mentre l'attributo N indica soltanto il numero del capitolo all'interno della parte in cui compare.
L'attributo N di ogni div0 specifica il numero della parte.
I salti di pagina sono codificati mediante elementi PB accompagnati da un attributo n il cui valore corrisponde al numero della pagina nel testo cartaceo di riferimento.
Classificazione: Classificazione Decimale Dewey

<h2>Descrizione del profilo:</h2>
Data di creazione: 1988
Lingua: Italiano
Categoria: NARRATIVA ITALIANA. 1945 -1999
CDD: 853.914
<h2>Descrizione della revisione</h2>
Data: 2003
Digitalizzazione e correzione: Salvatore Salzillo
Descrizione: digitalizzazione e correzione del testo in base alla edizione di riferimento: Matteo Collura, Baltico : un'epopea siciliana, Luigi Reverdito Editore, 1988.

Data: 2003
Codifica e revisione: Salvatore Salzillo
Descrizione: codifica XML del testo

</body></html>

Appendice 3

STAMPA DELL'OUTPUT HTML

Baltico. Un'epopea siciliana: edizione elettronica

Titolo dell'Opera: Baltico. Un'epopea siciliana: edizione elettronica

Autore: Matteo Collura

Digitalizzazione e codifica elettronica a cura di: Salvatore Salzillo

Supervisione: Prof. Lucrezia Lorenzini

Edizione: Prima edizione elettronica 22-07-2004

Dimensioni file: ca. 372 kb

Pubblicazione e Distribuzione: Universitá degli Studi di Messina, Facoltá di Lettere e Filosofia, Dipartimento di Filologia e Linguistica, Cattedra di Letteratura e Filologia Siciliana

Luogo di pubblicazione: Messina

Data di pubblicazione: 07-2004

Descrizione della fonte

Testo di Riferimento:
TITOLO: Baltico
Sottotitolo: Un epopea siciliana
Autore: Matteo Collura
Edizione: prima edizione
Dimensioni: 219 p. 21 cm.
Editore: Luigi Reverdito Editore
Luogo di pubblicazione: Trento
Anno di pubblicazione: 1988

ISBN: 88-342-0229-5
Collana: Biblioteca

Descrizione della codifica

Descrizione del progetto: Questo testo è stato digitalizzato e codificato per il progetto DigiSic, al fine di fornire un'esemplificazione di testo letterario sottoposto a codifica elettronica

Descrizione dei principi di codifica: Il testo elettronico è stato corretto e riportato al testo cartaceo dell'edizione di riferimento. Tutti i simboli di segnalazione del discorso diretto e di citazione sono stati convertiti sotto forma di entità numeriche XML. Tutti i trattini di rimando a capo sono stati eliminati. I valori standard delle date sono forniti in formato GG-MM-AAAA.

Descrizione dei principi di codifica inerenti il sistema di riferimento: L'attributo ID di ogni DIV1 contiene il riferimento canonico per ogni divisione nella forma XX.YY dove XX è il numero della parte in numeri romani e YY indica il capitolo in numeri arabi, mentre l'attributo N indica soltanto il numero del capitolo all'interno della parte in cui compare. L'attributo N di ogni div0 specifica il numero della parte. I salti di pagina sono codificati mediante elementi PB accompagnati da un attributo n il cui valore corrisponde al numero della pagina nel testo cartaceo di riferimento.

Classificazione: Classificazione Decimale Dewey

Descrizione del profilo:

Data di creazione: 1988
Lingua: Italiano
Categoria: NARRATIVA ITALIANA. 1945 -1999
CDD: 853.914

Descrizione della revisione

Data: 2003
Digitalizzazione e correzione: Salvatore Salzillo
Descrizione: digitalizzazione e correzione del testo in base alla edizione di riferimento : Matteo Collura, Baltico : un'epopea siciliana, Luigi Reverdito Editore, 1988.

Data: 2003
Codifica e revisione: Salvatore Salzillo
Descrizione: codifica XML del testo

III.12. OUTPUT HTML

Dopo l'analisi della produzione di un output HTML per i metadati dell'Header TEI, passiamo ora ad esaminare il foglio di stile XSLT mediante il quale abbiamo trasformato in HTML il sorgente XML dell'edizione elettronica del romanzo di Matteo Collura, oggetto del nostro lavoro. Anche in questo caso, in appendice forniamo il codice completo del foglio di stile XSLT adoperato per l'elaborazione.

Nella parte iniziale il codice ricalca da vicino quello del file, già visto, per la trasformazione dell'Header. Come di consueto il primo template che incontriamo è quello dell'elemento radice, che definisce i tag principali del nostro file HTML. Salta subito all'occhio la presenza nell'<*head*> HTML di una serie di istruzioni di formattazione incluse dentro il marcatore `<style type="text/css">`, si tratta dei comandi di un foglio di stile CSS incorporato dentro il file HTML[180]. Come già detto, CSS è un linguaggio di rappresentazione con cui associare delle norme di visualizzazione ai tag HTML (o XML).

Le istruzioni di formattazione CSS hanno una sintassi molto semplice, composta da due blocchi principali, a sinistra troviamo il *selettore*, il quale serve a definire a quale parte del documento verrà applicata la regola, a destra, racchiusa tra parentesi graffe, si trova invece il *blocco delle dichiarazioni* il quale contiene le norme di rappresentazione da associare all'elemento definito dal selettore.

Ad esempio se prendiamo la prima istruzione CSS del nostro foglio di stile:

```
p {   font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
      font-size: 12pt;
      text-align: justify;
      text-indent: 25pt;
      line-height: 2em;}
```

[180] Abitualmente si preferisce non inserire direttamente dentro il file HTML il foglio di stile CSS, ma piuttosto connetterlo ad esso mediante il collegamento ad un file esterno, in questa sede per esigenze di trattazione abbiamo preferito non adottare questa pratica.

vediamo a destra il selettore *p*, il quale associa le regole di riproduzione definite nel blocco delle dichiarazioni a tutti gli elementi <*p*> del documento HTML, in pratica al testo contenuto dentro i tag <*p*> … </*p*> nel file HTML. A sinistra entro parentesi graffe il blocco delle dichiarazioni definisce le istruzioni di formattazione sotto forma di coppie *proprietà:valore*; la proprietà, separata dal suo valore dai due punti, definisce una caratteristica dell'elemento da modificare; le varie coppie *proprietà:valore* sono separate tra di loro dal punto e virgola. Riprendendo il nostro esempio notiamo che la prima proprietà definita (font-family) riguarda il tipo di carattere con cui deve essere reso graficamente il testo, i suoi valori stabiliscono che esso dovrà essere riprodotto usando il carattere Verdana oppure, in caso di assenza del carattere in questione nel sistema in cui verrà visualizzato il documento, con il font Geneva, oppure Arial e così via. Le altre istruzioni stabiliscono che il carattere debba avere una dimensione di dodici punti, che il testo presenti un allineamento di tipo giustificato, che la prima riga di ogni paragrafo sia rientrata di venticinque punti e che l'interlinea abbia un valore di due *em*[181].

Continuando ad esaminare il CSS incorporato nel nostro file HTML, notiamo che un certo numero di selettori è preceduto da un punto ed ha nomi non corrispondenti con quelli abituali dei tag HTML, si tratta dei selettori di classe. I selettori di classe specificano lo stile di quegli elementi nel documento aventi un attributo *class*[182] corrispondente con il nome del selettore di classe, ad esempio il selettore di classe `.italic {font-style: italic;}` stabilisce che tutti gli elementi in cui si trovi un attributo class="italic" presentino il testo in corsivo, non limitandosi ad una determinata categoria di tag; potremo quindi avere: `<p class="italic">`, `<span class="italic">`, `<div class="italic">` etc. e tutti si atterranno alle istruzioni di visualizzazione definite dal selettore di classe italic `{font-style: italic;}`.

[181] L'em (em-height) è una unità di misura relativa ai font uguale all'altezza del box del carattere del font. Pertanto impostare il line-height a 2em corrisponde a stabilire una interlinea doppia rispetto alla dimensione del font.
[182] Così come ID, *class* è un attributo globale che può ricorrere in tutti gli elementi HTML.

Tornando al foglio di stile XSLT, subito dopo il modello per l'elemento radice incontriamo il template del <*teiHeader*>: `<xsl:template match="teiHeader"/>`; questo modello non ha alcun contenuto in modo tale da escludere dall'elaborazione, (come già visto con il foglio di stile per l'Header TEI) l'elemento in questione con tutti i suoi figli, infatti in questo caso non desideriamo un output anche per i metadati del documento, avendo noi deciso di inserirli in un file separato, illustrato nelle pagine precedenti.

Per agevolare il lettore nella consultazione dell'output HTML di Baltico abbiamo ritenuto utile porre all'inizio del file un indice con collegamenti ipertestuali ai capitoli; così come l'intero documento anche l'indice dei capitoli è stato generato dinamicamente grazie al foglio di stile XSLT. Per fare ciò ci siamo giovati principalmente della possibilità offerta dal linguaggio XSLT di applicare i modelli secondo determinate modalità. Mediante le modalità, che come vedremo fra poco, sono specificate usando l'attributo mode, non vengono solamente selezionati i nodi a cui si devono applicare le regole ma anche la modalità con cui devono essere trattati. In pratica la modalità costituisce un sottoinsieme di template, consentendo così di definire modelli diversi che utilizzano gli stessi elementi, anche se in modo differente. Sostanzialmente grazie alle modalità il documento viene percorso più volte dal processore, ed in ognuna di queste elaborazioni vengono applicate le differenti regole agli elementi secondo le modalità specificate nei diversi template.

Quanto ci apprestiamo ad illustrare di seguito costituisce una ulteriore riprova dell'utilità della codifica dichiarativa e della potenza di XSLT nel trattamento informatico dei testi; questi strumenti, infatti, consentono di manipolare e gestire i dati testuali secondo modalità ed approcci diversi, in risposta a differenti esigenze, anche all'interno di un medesimo documento. Nello specifico per la produzione dell'indice HTML di Baltico abbiamo operato nel modo seguente. Si ricorderà che nel sorgente XML del testo ogni capitolo è racchiuso dentro un elemento <*div1*> contraddistinto da alcuni attributi, ad esempio: `<div1 type="cap" n="2"`

id="I.2">, in particolare l'attributo *id* costituisce un identificatore che definisce in modo univoco ogni capitolo e che quindi ben si presta alla realizzazione di un sistema di ancore e riferimenti ipertestuali; presupposto su cui abbiamo fondato l'impostazione della programmazione di questa porzione del foglio di stile.
L'elaborazione effettiva del nostro indice parte dal nodo <*text*>, elemento che nella sintassi TEI, come sappiamo, ospita il contenuto testuale vero e proprio dell'opera:

```
<xsl:template match="text">
      <xsl:apply-templates mode="index"/>
      <xsl:apply-templates/>
</xsl:template>
```

come vediamo il codice di questo modello presenta due <xsl:apply-templates> il primo dei quali contraddistinto dall'attributo mode="index", grazie a questa sintassi abbiamo istruito il processore a percorrere per due volte il documento elaborando i dati secondo modalità differenti, una prima volta applicando i template caratterizzati dall'attributo *mode="index"*, ed una seconda volta, una volta conclusa la modalità "*index*", eseguendo invece i modelli privi di questo attributo. Così facendo, modelli diversi trovano corrispondenza con gli stessi nodi e producono differenti output. I modelli di tipo "*index*" creano la "copertina" del documento ed un indice dei capitoli; sostanzialmente, per quanto riguarda l'indice operano così: l'albero del documento viene percorso mandando in output soltanto i pochi elementi necessari alla costituzione dell'indice ed escludendo gli altri, in particolare, cuore della realizzazione del nostro indice è il modello

```
<xsl:template match="head" mode="index">
    <a href="#{../@id}"><xsl:value-of select="."/></a><br/>
    <xsl:apply-templates mode="index"/>
</xsl:template>
```

Questo template, durante l'elaborazione in modalità "index", seleziona tutti i titoli dei capitoli, che come sappiamo sono racchiusi dentro il tag <*head*>, e ne manda in output il valore racchiudendolo in un link HTML (`<a href="destinazione">…</a>`), il link punta ad un'ancora HTML (`<a name="nome destinazione">`), che viene generata durante l'elaborazione standard (ossia quella non in modalità "*index*") dal modello delle intestazioni di capitolo di tipo ordinale:

```
<xsl:template match="head[@type='ord']">
      <a name="{../@id}"/>
            <h2>
                  <xsl:apply-templates/>
            </h2>
</xsl:template>
```

Come vediamo sia il nome dell'ancora (che costituisce la destinazione del link), sia la destinazione del link vengono generati grazie ad un espressione XPath "../@id" che seleziona in modo relativo l'attributo *id* dell'elemento padre del nodo contesto e lo manda in output[183] sia nell'attributo name dell'ancora, producendo così il nome dell'ancora, sia nell'attributo href (attributo che indica la destinazione del collegamento ipertestuale) del link. Per aiutare a comprendere meglio riportiamo una spiegazione empirica del funzionamento del foglio. Prendiamo, ad esempio, nel sorgente XML il codice dell'inizio del secondo capitolo:

```
<pb n="18"/>
<div1 type="cap" n="2" id="I.2">
<head type="ord">II</head>
<head type="tem">Quando e come i contadini
incubarono il malanno</head>
<p>
```

[183] Quando negli attributi di un output HTML o XML si vuole inserire un valore estratto da un nodo del documento XML di origine, non si può usare l'elemento 'xsl:value-of'; al suo posto si usano delle parentesi graffe.

Evocazione di visionari o fatalità del caso, fatto è che, ad un certo momento, [. . .]

Allorché l'elemento `<head type="ord">II</head>` viene raggiunto dal modello dell'<*head*> in modalità *index* (`<xsl:template match="head" mode="index">`), viene prodotto questo codice HTML:

```
<a href="#I.2">II</a>
```

come vediamo è stato creato un collegamento ipertestuale, che punta ad un'ancora nel documento il cui nome è "I.2",l'ancora sarà generata nel prosieguo dell'elaborazione dal template:

```
<xsl:template match="head[@type='ord']">
      <a name="{../@id}"/>
      <h2>
            <xsl:apply-templates/>
      </h2>
</xsl:template>
```

Il modello in questione non appartiene più come possiamo notare alla modalità *index*, ma fa parte dell'elaborazione, per così dire, principale la quale si occupa della produzione dell'output del contenuto testuale vero e proprio del documento. Questo è l'output HTML creato dal modello:

```
<a name="I.2"></a>
<h2>II</h2>
```

Abbiamo creato dei riferimenti incrociati, in cui, il valore sia del link, sia dell'ancora è generato dinamicamente sulla base della posizione del nodo contesto nell'albero del documento, a partire dal valore dell'attributo *id* del nodo padre dell'elemento contesto[184], che nello specifico è I.2.

[184] Abbiamo optato per una sintassi relativa facente riferimento in modo generico al nodo padre (../@id) e non ad una sintassi assoluta come ad esempio div1/@id, per far si che i templates fossero efficaci con qualsiasi <*head*>, anche con quello, ad esempio della nota finale nel <back> del documento, dove l'elemento non è figlio di un nodo <*div1*> bensì semplicemente di un nodo <*div*>.

Dopo aver esaminato la generazione dinamica dell'indice poco altro resta da esaminare nel foglio di stile, buona parte dei template ricalcano nella sintassi quelli già visti per l'elaborazione dell'Header TEI e si limitano ad inserire dentro opportuni tag HTML le varie parti del testo.

Merita di essere menzionato il modello per il trattamento delle citazioni, che nel sorgente, come forse il lettore ricorderà, sono identificate dal marcatore *<q>* accompagnato dall'attributo `type='citazione'`. Nel testo di riferimento le citazioni presentano caratteristiche grafiche diverse, alcune sono rese in corsivo, altre invece sono messe in risalto mediante l'inserimento in un blocco di testo graficamente autonomo, altre ancora non sono contraddistinte da alcuna evidenziazione tipografica particolare. Nella codifica, grazie all'attributo *rend*, si è registrato, ove necessario, in che modo è riportata la citazione nella fonte materiale, i valori usati nell'attributo per far ciò sono stati: *bloc* per le citazioni dentro blocchi di testo graficamente autonomi ed *italic* per le citazioni in corsivo. Tale distinzione è stata naturalmente mantenuta anche nel template per il trattamento dei nodi di citazione il cui codice riportiamo di seguito:

```
<xsl:template match="q[@type='citazione']" priority="1">
            <xsl:choose>
                  <xsl:when test="@rend='bloc'">
                        <blockquote>
                              <xsl:apply-templates/>
                        </blockquote>
                  </xsl:when>
                  <xsl:when test="@rend='italic'">
                        <cite class="italic">
                              <xsl:apply-templates/>
                        </cite>
                  </xsl:when>
                  <xsl:otherwise>
                        <cite>
```

```
                        <xsl:apply-templates/>
                    </cite>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:template>
```

Come si vede adoperiamo anche in questo caso una elaborazione condizionale per rendere nell'output in modi differenti l'elemento, in base alle indicazioni sulle caratteristiche di evidenziazione grafica dello stesso registrate nell'attributo rend. Nel caso, in cui il testo della citazione marcata da <*q*> presenti un attributo rend='bloc', questa nell'output sarà racchiusa dentro l'opportuno tag <*blockquote*>, ove invece nel sorgente la citazione sia associata all'attributo rend='italic', sarà marcata grazie all'elemento `<cite class="italic">`[185], in tutti gli altri casi invece si adotterà semplicemente il marcatore <*cite*>. Si sarà notato che il modello appena descritto dispone di un attributo `priority="1"`; la sua funzione è quella di impostare una priorità di elaborazione per il modello a cui è associato nell'eventualità in cui più modelli (all'interno però di una medesima modalità) trovino corrispondenza con uno stesso elemento. Nel foglio XSLT immediatamente dopo il template per le citazioni incontriamo due modelli che con questo potrebbero collidere durante l'elaborazione:

```
<xsl:template match="*[@rend='bloc']">
                <div class="bloc">
            <xsl:apply-templates/>
                </div>
        </xsl:template>
```

[185] Di default nei browser il testo contenuto dentro l'elemento <*cite*> è reso graficamente in corsivo, tuttavia mediante il foglio di stile CSS incorporato all'inizio del file HTML abbiamo ridefinito le caratteristiche di visualizzazione di questo tag stabilendo che il testo non sia riprodotto in corsivo ma normalmente. Abbiamo proceduto così per distinguere graficamente le citazioni anche nell'output HTML, ove necessaria la rappresentazione in corsivo del testo è stata specificata nel tag <*cite*> per mezzo di una apposita classe CSS (*italic*) da noi opportunamente definita nel foglio di stile.

```
<xsl:template match="*[@rend='italic']">
      <span class="italic"><xsl:apply-templates/></span>
      </xsl:template>
```
[186]

Mediante l'impostazione di una priorità[187] elevata abbiamo fatto in modo che con i nodi `<q type='citazione'>` anche se accompagnati da un attributo *rend* intervenga sempre il primo modello, mentre in tutti gli altri casi troveranno corrispondenza i modelli generici definiti successivamente.

Il foglio di stile appena illustrato è stato in più casi alquanto personalizzato per l'elaborazione specifica dell'edizione XML del romanzo di Collura; tuttavia, si è sempre cercato durante la sua programmazione di aver cura di mantenerlo il più possibile generico ed universale[188], in modo da poterlo eventualmente in futuro adoperare anche con un corpus di testi dalle caratteristiche di codifica omogenee ed analoghe al nostro, nella speranza così di offrire un primo, minimo, contributo alla costituzione di un archivio digitale della Cattedra di Filologia e Letteratura Siciliana dell'Università di Messina.

Concludendo, vorremmo dire che nella produzione dell'output HTML si è cercato di "reinterpretare" graficamente il testo di Baltico adattandolo al nuovo medium, pur sforzandosi di mantenere una certa coerenza estetica con l'edizione di riferimento; è, comunque, bene tener presente che quella da noi proposta è soltanto una delle possibili forme di riproduzione del testo, come vedremo meglio nelle prossime pagine, i formati di rappresentazione possono venir modellati ed impostati in modo tale da attagliarsi alle esigenze più disparate.

[186] La funzione di questi modelli è quella di elaborare tutti i nodi cui sia associato un attributo *rend* con valore *italic* oppure *bloc*. Si ricordi, infatti, che *rend* è un attributo globale che può ricorrere in pressoché tutti gli elementi TEI.

[187] Le priorità di default dei modelli non superano mai il valore di 0.5.

[188] In tal senso si spiega la decisione di omettere dall'elaborazione automatica le dediche ed epigrafi iniziali. Dediche ed epigrafi si possono presentare con caratteristiche tipografiche e di disposizione assai diverse nei testi, poco proficuo e sensato riteniamo sarebbe stato il cercare di prevedere tutte le possibili forme in cui queste possono apparire in un testo, nell'impostazione di una elaborazione di tipo batch quale quella XSLT, considerando anche la loro spesso esigua dimensione nei testi. Pertanto si è ritenuto più utile inserirle "manualmente" nell'output.

Appendice 4

FOGLIO DI STILE XSLT PER LA PRODUZIONE DELL'OUTPUT HTML DI BALTICO

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="iso-8859-1" version="4.01"
doctype-public="-//W3C//DTD HTML 4.01 Transitional//EN"/>
  <xsl:strip-space elements="*"/>
  <xsl:preserve-space elements="author"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/> - <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
        </title>
        <style type="text/css">
p {
  font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
  font-size: 12pt;
  text-align: justify;
  text-indent: 25pt;
  line-height: 2em;
  margin: 0pt 0pt;
}

a { color: Black; font-size: 13pt;  text-align: left;  font-family:
"Times New Roman", Times, serif;  text-decoration: none;     line-
height: 1em; }

a:hover { color: #1E20FF; }

h1 { font-size: 17pt;  text-align: left;  font-weight: bold;  font-
family:"Times New Roman", Times, serif;  }

h2 {font-size: 15pt;  text-align: left;  font-family:"Times New
Roman", Times, serif; }

cite { font-style: normal; }

.italic {font-style: italic;}

.mc { font-size: 16pt;  font-family: "Times New Roman", Times, serif;
font-weight: bold;  text-align: center;  }

.bal {
  font-size: 25pt;  font-family: "Times New Roman", Times, serif;
font-weight: bold;  text-align: center; letter-spacing: 1em;}

.sub {
font-size: medium;  font-family: "Times New Roman", Times, serif;
text-align: center;}

.imp {
```

```
  font-size: medium;  font-family: "Times New Roman", Times, serif;
text-align: center;
}

.cap { margin-left: 10%;  margin-right: 10%;  }

.part {
font-size: medium;  font-family: "Times New Roman", Times, serif;
 font-style: italic;
 text-align: left;}

 .center { text-align: center; }

 .hr { width: 65%; }

 .bloc {font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
  font-size: 12pt;
  line-height: 2em;
  margin-left: 10%;  margin-right: 10%; }

 blockquote {
  font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
  font-size: 12pt;
  text-align: justify;
  line-height: 2em;
}

body { background: #C0D7DF;
    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size: 12pt;
    line-height: 2em;}
</style>
       </head>
       <body>
         <xsl:apply-templates/>
       </body>
     </html>
  </xsl:template>
  <xsl:template match="teiHeader"/>
  <xsl:template match="text">
    <xsl:apply-templates mode="index"/>
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="front" mode="index">
    <xsl:apply-templates mode="index"/>
  </xsl:template>
  <xsl:template match="titlePage" mode="index">
    <div class="center">
      <br/>
      <hr class="hr"/>
      <br/>
      <xsl:apply-templates mode="index"/>
      <br/>
      <br/>
      <hr class="hr"/>
      <br/>
      <br/>
      <span class="imp">INDICE</span>
    </div>
  </xsl:template>
  <xsl:template match="docAuthor" mode="index">
```

```
  <span class="mc">
    <xsl:value-of select="."/>
  </span>
  <p> </p>
  <p> </p>
</xsl:template>
<xsl:template match="docAuthor"/>
<xsl:template match="docTitle" mode="index">
  <xsl:apply-templates mode="index"/>
</xsl:template>
<xsl:template match="titlePart[@type='main']" mode="index">
  <span class="bal">
    <xsl:value-of select="."/>
  </span>
  <br/>
</xsl:template>
<xsl:template match="titlePart[@type='main']"/>
<xsl:template match="titlePart[@type='sub']" mode="index">
  <span class="sub">
    <xsl:value-of select="."/>
  </span>
  <br/>
  <p> </p>
  <p> </p>
</xsl:template>
<xsl:template match="titlePart[@type='sub']"/>
<xsl:template match="docImprint" mode="index">
  <span class="imp">
    <xsl:value-of select="."/>
  </span>
</xsl:template>
<xsl:template match="docImprint"/>
<xsl:template match="div[@type='ded']" priority="1">
  <p>***inserire dedica***</p>
</xsl:template>
<xsl:template match="div[@type='ep']" priority="1">
  <p>***inserire epigrafe***</p>
</xsl:template>
<!-- inzio mode index -->
<xsl:template match="body">
  <xsl:apply-templates/>
</xsl:template>
<xsl:template match="div0" mode="index">
  <div class="cap">
    <br/>
    <span class="part">
      <xsl:value-of select="p"/>
    </span>
    <br/>
    <xsl:apply-templates mode="index"/>
  </div>
</xsl:template>
<xsl:template match="div0/p" mode="index"/>
<xsl:template match="div1" mode="index">
  <br/>
  <xsl:apply-templates mode="index"/>
</xsl:template>
<xsl:template match="back/div" mode="index">
  <div class="cap">
    <br/>
    <xsl:apply-templates mode="index"/>
```

```
  </div>
</xsl:template>
<xsl:template match="head" mode="index">

    <xsl:value-of select="."/>
  </a>
    <br/>
    <xsl:apply-templates mode="index"/>

</xsl:template>
<xsl:template match="text()|@*" mode="index"/>
<!-- fine mode index -->
<xsl:template match="div0">
  <div class="center">
    <br/>
    <br/>
    <br/>
    <span class="imp">
      <xsl:value-of select="p"/>
    </span>
    <br/>
    <br/>
***<br/>
    <br/>
    <br/>
  </div>
  <xsl:apply-templates/>
</xsl:template>
<xsl:template match="div0/p"/>
<xsl:template match="div1">
  <xsl:comment>INIZIO CAPITOLO <xsl:value-of select="@id"/>
  </xsl:comment>
  <div class="cap">
    <xsl:apply-templates/>
  </div>
  <xsl:comment>FINE CAPITOLO <xsl:value-of select="@id"/>
  </xsl:comment>
</xsl:template>
<xsl:template match="head[@type='ord']">
  <a name="{../@id}"/>
  <h2>
    <xsl:apply-templates/>
  </h2>
</xsl:template>
<xsl:template match="head[@type='conv']">
  <a name="{../@id}"/>
  <h2>
    <xsl:apply-templates/>
  </h2>
</xsl:template>
<xsl:template match="head[@type='tem']">
  <h1>
    <xsl:apply-templates/>
  </h1>
  <hr/>
  <br/>
</xsl:template>
<xsl:template match="p">
  <p>
    <xsl:apply-templates/>
  </p>
```

```
  </xsl:template>
  <xsl:template match="div[@type='nota']">
    <div class="cap">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="q[@type='citazione']" priority="1">
    <xsl:choose>
      <xsl:when test="@rend='bloc'">
        <blockquote>
          <xsl:apply-templates/>
        </blockquote>
      </xsl:when>
      <xsl:when test="@rend='italic'">
        <cite class="italic">
          <xsl:apply-templates/>
        </cite>
      </xsl:when>
      <xsl:otherwise>
        <cite>
          <xsl:apply-templates/>
        </cite>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <xsl:template match="l">
    <xsl:apply-templates/>
    <br/>
  </xsl:template>
<xsl:template match="lb"><br/> </xsl:template>

<xsl:template match="*[@rend='bloc']">
       <div class="bloc">
    <xsl:apply-templates/>
       </div>
  </xsl:template>

<xsl:template match="*[@rend='italic']">
  <span class="italic"><xsl:apply-templates/></span>
  </xsl:template>

<xsl:template match="q[@type='epistola']">
  <div><xsl:apply-templates/></div>
  </xsl:template>

</xsl:stylesheet>
```

III.13. ACCESSIBILITÀ

Concludendo il capitolo precedente abbiamo affermato che i dati nella loro rappresentazione possono venire modellati ed impostati in modo tale da attagliarsi alle esigenze più disparate. In particolar modo, le possibilità di personalizzazione dell'output connesse con la codifica dichiarativa si rivelano particolarmente utili nella produzione di formati di riproduzione che, grazie a particolari accorgimenti, consentano un'agevole accessibilità ai contenuti anche a persone con disabilità di vario tipo.

Abbiamo pertanto ritenuto opportuno nell'ambito del nostro lavoro di codifica elettronica e riproduzione digitale del testo di Matteo Collura apprestare una piccola applicazione per la personalizzazione dell'output rivolta principalmente a quegli utenti afflitti da disabilità, in particolar modo di natura visiva. Quella che ci apprestiamo ad illustrare non è per nulla un'applicazione completa ed ampiamente efficace, ma piuttosto vuol essere una esemplificazione delle potenzialità della codifica dichiarativa XML e delle tecnologie connesse nel favorire la fruibilità dei testi anche a soggetti disabili. È, infatti, il tema dell'accessibilità assai vasto e complesso e non è stato per noi possibile in questa circostanza andare oltre un approccio estremamente superficiale ed incompleto; l'unica speranza da noi nutrita nell'elaborare l'elementare lavoro che mostreremo di seguito è quella di fornire un primo spunto per la riflessione in vista di un futuro approfondimento dell'argomento.

A vantaggio di possibili lettori con problemi di ipovisione o daltonismo abbiamo approntato un foglio di stile ed una piccola applicazione, con cui consentire all'utente di scegliere sulla base delle proprie esigenze o più semplicemente dei propri gusti personali alcune caratteristiche di visualizzazione, come la dimensione dei caratteri oppure il colore dello sfondo, dell'output HTML di Baltico; come di consueto in appendice viene riportato il foglio di stile utilizzato. Si sarà notato che

il codice è pressoché identico a quello del foglio per l'output HTML del romanzo; le principali differenze si trovano all'inizio del documento per la presenza di un nuovo comando, `<xsl:param>`, e dentro il foglio di stile CSS incorporato nel file HTML. Obiettivo di questo XSLT è quello di consentire la personalizzazione dell'output HTML; per il raggiungimento del nostro scopo ci siamo giovati della possibilità offerta da XSLT di influenzare dinamicamente l'output mediante parametri definiti all'esterno del foglio di stile. Come si può constatare esaminando il codice riportato in appendice, subito dopo la dichiarazione del tipo di output e prima delle varie dichiarazioni di modello, incontriamo tre tag `<xsl:param>`, i quali hanno la funzione di definire tre differenti parametri globali dal nome[189], rispettivamente, di *color*, *font* e *size*. La funzione dei parametri è quella di ricevere dati, dall'esterno del foglio di stile e del sorgente XML, grazie ai quali manipolare l'output. Nello specifico il compito dei tre parametri dichiarati nel nostro foglio di stile è di ricevere dall'esterno; vedremo in seguito come alcuni valori definiti dall'utente ed utilizzarli nella produzione dell'output, in particolare mediante il parametro *color* si stabilisce il colore di sfondo del documento, mentre con i parametri *font* e *size* si possono specificare il tipo e le dimensioni dei caratteri del testo. Proseguendo nell'analisi del codice XSLT notiamo che mediante espressioni `<xsl:value-of>` il valore dei parametri viene inserito all'interno dei modelli[190]; in realtà, esclusivamente nel template del nodo radice nella parte relativa alla creazione del foglio CSS incorporato per la definizione delle norme di visualizzazione del documento HTML.

Ad esempio, come vediamo nella porzione di codice che riportiamo di seguito relativa al marcatore HTML per i paragrafi del documento, <*p*>:

```
p {
      font-family:<xsl:value-of select="$font"/>;
```

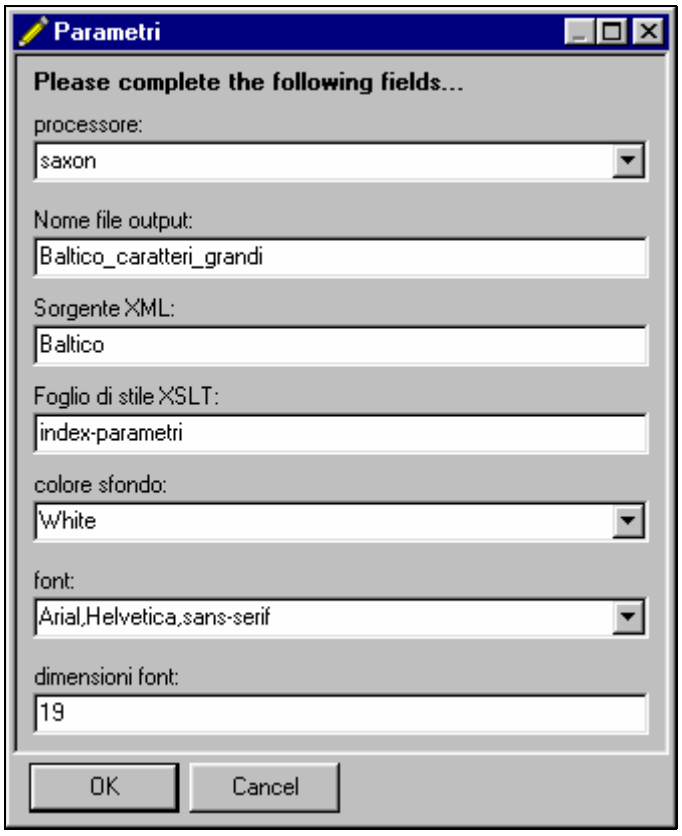
[189] Il nome dei parametri XSLT è definito grazie all'attributo name.
[190] Il riferimento ad un parametro nel foglio di stile è dato dal nome del parametro preceduto dal simbolo del dollaro ($nome_parametro).

```
font-size: <xsl:value-of select="$size"/>pt;
text-align: justify;
text-indent: 25pt;
line-height: 2em;
margin: 0pt 0pt;}
```

sia il tipo di carattere, sia le dimensioni dello stesso vengono definite grazie ai parametri stabiliti dall'utente; in tal modo, un lettore con difficoltà visive potrà, volendo, godere di una versione "personale" dell'output HTML di Baltico riprodotta con un carattere ad alta leggibilità e dalle dimensioni adeguate a sopperire al proprio deficit visivo. Analogamente a quanto appena visto anche in altri punti del template in questione si adottano i parametri per la definizione del valore delle dichiarazioni CSS. Ad esempio, si adopera un parametro (*color*) per determinare il colore dello sfondo, in modo tale da consentire all'utente di impostare una buona combinazione di colori fortemente contrastata, adeguata al tipo di disturbo visivo, per il testo e lo sfondo, opzione questa particolarmente utile per i soggetti affetti da daltonismo[191].

Esistono vari metodi per passare dei parametri al processore XSLT, il più semplice e rudimentale è utilizzando la riga dei comandi, nella maggior parte dei casi tuttavia questa operazione viene eseguita in qualche genere di applicazione che utilizza XSLT per operare sui documenti XML ed elaborarli. Nel nostro lavoro abbiamo optato per seguire la via più semplice e servirci della riga dei comandi per la trasmissione dei parametri al processore; tuttavia, per cercare di agevolare gli utenti abbiamo realizzato una piccola applicazione sotto forma di clip per il freeware NoteTab, grazie alla quale creare rapidamente un file .bat con cui trasmettere i parametri al processore XSLT dalla riga dei comandi, di seguito ne riportiamo una schermata dell'interfaccia.

[191] In realtà sarebbe stato opportuno definire nel nostro foglio di stile un secondo parametro anche per il colore del testo. Torniamo a ribadire che il fine di questo rudimentale lavoro non è la realizzazione di un'applicazione effettivamente utilizzabile sul campo ma piuttosto esemplificare le possibilità offerte sul fronte dell'accessibilità dalla codifica dichiarativa XML.



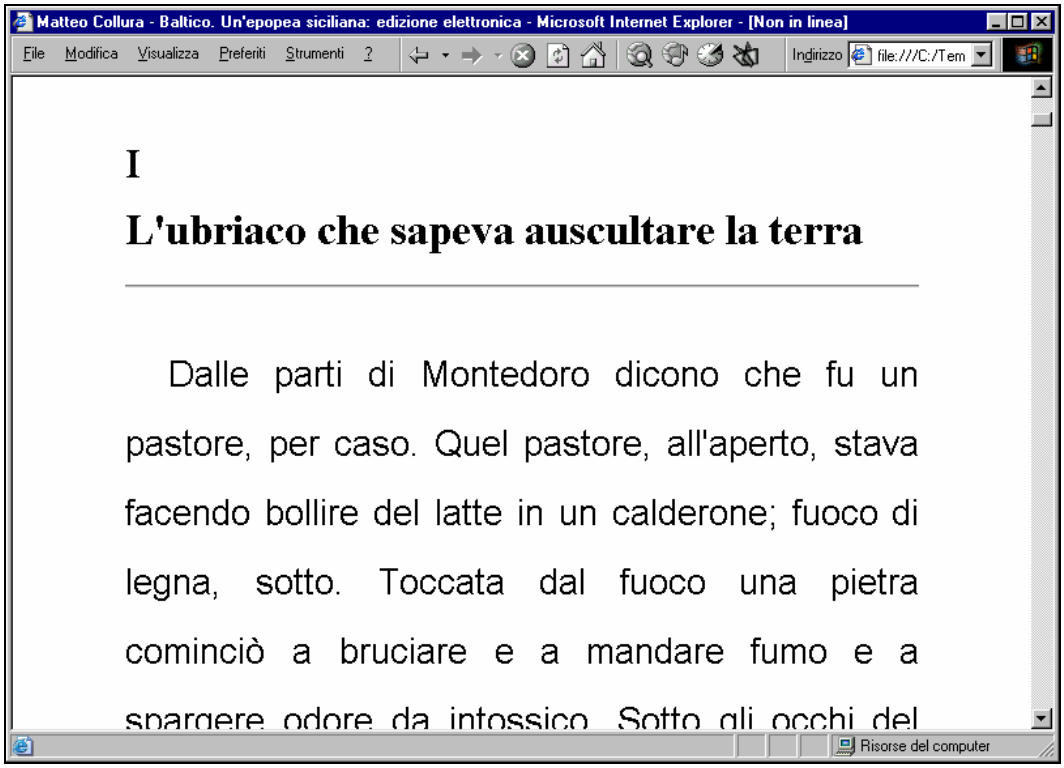
Il primo campo della finestra è utilizzato per specificare il nome del processore XSLT da adottare per la trasformazione; nella schermata riportata, ad esempio, si è optato per l'adozione di Saxon[192]. I tre riquadri successivi servono ad indicare rispettivamente il nome del file HTML di output ed i nomi del sorgente XML e del foglio XSLT da utilizzare per l'elaborazione. I parametri da passare al processore possono venir specificati grazie agli ultimi tre campi della finestra dell'applicazione, nel nostro caso la scelta è caduta su un colore di sfondo bianco[193] e su di un carattere di tipo Arial oppure *senza grazie* (sans-serif) di dimensioni 19pt[194].

Di seguito riportiamo la sintassi di invocazione del processore prodotta dalla clip appena illustrata:

[192] Nell'applicazione si è scelto per praticità di riportare esclusivamente la sintassi per invocare i processori Saxon e Msxsl.
[193] Nella finestra sono specificati i valori in inglese dei colori accettati nella sintassi CSS.
[194] Il foglio è impostato in modo tale che una volta definite le dimensioni del carattere del testo dei capitoli vengano incrementate proporzionalmente anche quelle delle altre componenti testuali (intestazioni di capitolo, link, etc.)

saxon -o Baltico_caratteri_grandi.html Baltico.xml index-parametri.xsl color=white font=Arial,Helvetica,sans-serif size=19[195]

usata per attivare il processore mediante un file .bat ovvero direttamente attraverso la linea di comando, questa sintassi produce un output html con testo Arial 19 su sfondo bianco; di seguito riportiamo una porzione dell'output così come viene visualizzata nel browser Internet Explorer 6.0:



La codifica dichiarativa XML offre, grazie alla scissione dei dati dalla loro rappresentazione ed al forte grado di metainformatività di cui è portatrice, ampie risorse e prospettive per la realizzazione di soluzioni per l'accessibilità dei testi ai disabili. Ad esempio, ritornando al caso concreto di Baltico sarebbe ipotizzabile che un software di Text to Speech[196] rendesse più gradevole ed "umana" la lettura dei testi servendosi dell'attributo *who* dei tag <*q*> con cui abbiamo marcato il discorso diretto nel testo, per esempio cambiando "voce" ogni qual volta muti il parlante, o

[195] Si tenga presente che il testo in questione si sviluppa su di un'unica riga.
[196] Text to speech: applicazione *software* che trasforma un testo digitale nell'audio corrispondente al testo quando viene letto da una persona. È il tipo di *software* alla base dei *tool* assistivi con sintesi vocale, come *screen reader* e *browser* vocali.

ancora adottando una voce da uomo per i personaggi di sesso maschile e viceversa una voce da donna per i personaggi di sesso femminile. A tal proposito merita di essere menzionata una tecnologia basata su XML per la produzione di output audio dai testi e l'interazione attraverso comandi vocali o sonori con applicazioni software, il VoiceXML[197].

Dovrebbe ormai, a questo punto della trattazione, risultare abbastanza chiara la semplicità con cui è possibile trasformare, "tradurre", una struttura XML in un'altra grazie ad XSLT e quindi, ad esempio, convertire, essendo il VoiceXML basato su XML, i dati XML/TEI in una nuova sintassi VXML; tutto ciò ad ulteriore riprova dell'utilità ed, oserei dire, necessità della codifica dichiarativa dei testi, letterari e non.

Quello da noi tracciato in queste poche pagine, ben lungi dall'essere un quadro esaustivo e completo delle problematiche connesse con l'accessibilità dei testi a beneficio delle persone diversamente abili, vuol semplicemente fornire uno sguardo superficiale su queste tematiche al fine di far intravedere alcune delle importanti prospettive, che in questo campo si aprono grazie alle nuove tecnologie di codifica e trattamento informatico dei testi, e che riteniamo dovrebbero rappresentare nella loro concretizzazione una delle sfide prioritarie per le nascenti biblioteche digitali.

[197] VoiceXML è un acronimo che sta per *Voice Extensible Markup Language*, conosciuta anche con la sigla VXML, VoiceXML è una tecnologia che consente agli utenti di interagire con le applicazioni (primariamente legate ad internet) attraverso tecniche di riconoscimento vocale. Usando VXML gli utenti interagiscono con il browser adoperando la propria voce ed ascoltando degli output audio che possono sia essere preregistrati sia prodotti dinamicamente grazie all'ausilio di sintetizzatori vocali.

Appendice 5

FOGLIO DI STILE XSLT PER LA PRODUZIONE DI OUTPUT HTML MEDIANTE L'USO DI PARAMETRI PER ESIGENZE DI ACCESSIBILITA'

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="iso-8859-1" version="4.01"
doctype-public="-//W3C//DTD HTML 4.01 Transitional//EN"/>
  <xsl:param name="color"/>
  <xsl:param name="font"/>
  <xsl:param name="size"/>
  <xsl:strip-space elements="p"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/> - <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
        </title>
        <style type="text/css">
p {
  font-family:<xsl:value-of select="$font"/>;
  font-size: <xsl:value-of select="$size"/>pt;
  text-align: justify;
  text-indent: 25pt;
  line-height: 2em;
  margin: 0pt 0pt;
}

a { color: Black; font-size: <xsl:value-of select="1 + $size"/>pt;
text-align: left;  font-family: "Times New Roman", Times, serif;
text-decoration: none;  line-height: 1em; }

a:hover { color: #1E20FF; }

h1 { font-size: <xsl:value-of select="5 + $size"/>pt;  text-align:
left;  font-weight: bold;  font-family:"Times New Roman", Times,
serif;  }

h2 {font-size: <xsl:value-of select="3 + $size"/>pt;  text-align:
left;  font-family:"Times New Roman", Times, serif; }

cite { font-style: normal; }

.italic {font-style: italic;}

.mc { font-size: 16pt;  font-family: "Times New Roman", Times, serif;
font-weight: bold;  text-align: center;  }

.bal {
  font-size: 25pt;  font-family: "Times New Roman", Times, serif;
font-weight: bold;  text-align: center; letter-spacing: 1em;}

.sub {
```

```
font-size: medium;  font-family: "Times New Roman", Times, serif;
text-align: center;}

.imp {
  font-size: medium;  font-family: "Times New Roman", Times, serif;
text-align: center;
}

.cap { margin-left: 10%;  margin-right: 10%;  }

.part {
font-size: medium;  font-family: "Times New Roman", Times, serif;
 font-style: italic;
 text-align: left;}

 .center { text-align: center; }

 .hr { width: 65%; }

 .bloc {font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
  font-size: <xsl:value-of select="$size"/>pt;
  line-height: 2em;
  margin-left: 10%;  margin-right: 10%; }

 blockquote {
  font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
  font-size: <xsl:value-of select="$size"/>pt;
  text-align: justify;
  line-height: 2em;
}

body { background: <xsl:value-of select="$color"/>;
   font-family: <xsl:value-of select="$font"/>;
   font-size: 12pt;
   line-height: 2em;}
</style>
       </head>
       <body>
         <xsl:apply-templates/>
       </body>
     </html>
  </xsl:template>
  <xsl:template match="teiHeader"/>
  <xsl:template match="text">
    <xsl:apply-templates mode="index"/>
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="front" mode="index">
    <xsl:apply-templates mode="index"/>
  </xsl:template>
  <xsl:template match="titlePage" mode="index">
    <div class="center">
      <br/>
      <hr class="hr"/>
      <br/>
      <xsl:apply-templates mode="index"/>
      <br/>
      <br/>
      <hr class="hr"/>
      <br/>
      <br/>
```

```
      <span class="imp">INDICE</span>
    </div>
</xsl:template>
<xsl:template match="docAuthor" mode="index">
    <span class="mc">
      <xsl:value-of select="."/>
    </span>
    <p> </p>
    <p> </p>
</xsl:template>
<xsl:template match="docAuthor"/>
<xsl:template match="docTitle" mode="index">
    <xsl:apply-templates mode="index"/>
</xsl:template>
<xsl:template match="titlePart[@type='main']" mode="index">
    <span class="bal">
      <xsl:value-of select="."/>
    </span>
    <br/>
</xsl:template>
<xsl:template match="titlePart[@type='main']"/>
<xsl:template match="titlePart[@type='sub']" mode="index">
    <span class="sub">
      <xsl:value-of select="."/>
    </span>
    <br/>
    <p> </p>
    <p> </p>
</xsl:template>
<xsl:template match="titlePart[@type='sub']"/>
<xsl:template match="docImprint" mode="index">
    <span class="imp">
      <xsl:value-of select="."/>
    </span>
</xsl:template>
<xsl:template match="docImprint"/>
<xsl:template match="div[@type='ded']" priority="1">
    <p>***inserire dedica***</p>
</xsl:template>
<xsl:template match="div[@type='ep']" priority="1">
    <p>***inserire epigrafe***</p>
</xsl:template>
<!-- inzio mode index -->
<xsl:template match="body">
    <xsl:apply-templates/>
</xsl:template>
<xsl:template match="div0" mode="index">
    <div class="cap">
      <br/>
      <span class="part">
        <xsl:value-of select="p"/>
      </span>
      <br/>
      <xsl:apply-templates mode="index"/>
    </div>
</xsl:template>
<xsl:template match="div0/p" mode="index"/>
<xsl:template match="div1" mode="index">
    <br/>
    <xsl:apply-templates mode="index"/>
</xsl:template>
```

```
<xsl:template match="back/div" mode="index">
  <div class="cap">
    <br/>
    <xsl:apply-templates mode="index"/>
  </div>
</xsl:template>
<xsl:template match="head" mode="index">
  <a href="#{../@id}">
    <xsl:value-of select="."/>
    <br/>
    <xsl:apply-templates mode="index"/>
  </a>
</xsl:template>
<xsl:template match="text()|@*" mode="index"/>
<!-- fine mode index -->
<xsl:template match="div0">
  <div class="center">
    <br/>
    <br/>
    <br/>
    <span class="imp">
      <xsl:value-of select="p"/>
    </span>
    <br/>
    <br/>
***<br/>
    <br/>
    <br/>
  </div>
  <xsl:apply-templates/>
</xsl:template>
<xsl:template match="div0/p"/>
<xsl:template match="div1">
  <xsl:comment>INIZIO CAPITOLO <xsl:value-of select="@id"/>
  </xsl:comment>
  <div class="cap">
    <xsl:apply-templates/>
  </div>
  <xsl:comment>FINE CAPITOLO <xsl:value-of select="@id"/>
  </xsl:comment>
</xsl:template>
<xsl:template match="head[@type='ord']">
  <a name="{../@id}"/>
  <h2>
    <xsl:apply-templates/>
  </h2>
</xsl:template>
<xsl:template match="head[@type='conv']">
  <a name="{../@id}"/>
  <h2>
    <xsl:apply-templates/>
  </h2>
</xsl:template>
<xsl:template match="head[@type='tem']">
  <h1>
    <xsl:apply-templates/>
  </h1>
  <hr/>
  <br/>
</xsl:template>
<xsl:template match="p">
```

```
    <p>
      <xsl:apply-templates/>
    </p>
  </xsl:template>
  <xsl:template match="div[@type='nota']">
    <div class="cap">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="q[@type='citazione']" priority="1">
    <xsl:choose>
      <xsl:when test="@rend='bloc'">
        <blockquote>
          <xsl:apply-templates/>
        </blockquote>
      </xsl:when>
      <xsl:when test="@rend='italic'">
        <cite class="italic">
          <xsl:apply-templates/>
        </cite>
      </xsl:when>
      <xsl:otherwise>
        <cite>
          <xsl:apply-templates/>
        </cite>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <xsl:template match="l">
    <xsl:apply-templates/>
    <br/>
  </xsl:template>
  <xsl:template match="lb">
    <br/>
  </xsl:template>
  <xsl:template match="*[@rend='bloc']">
    <div class="bloc">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="*[@rend='italic']">
    <span class="italic">
      <xsl:apply-templates/>
    </span>
  </xsl:template>

<xsl:template match="q[@type='epistola']">
  <div><xsl:apply-templates/></div>
  </xsl:template>
</xsl:stylesheet>
```

III.14. XML DATA ISLAND

Oltre che mediante trasformazione XSLT è possibile accedere a dati XML anche per mezzo di tecniche di collegamento dati (data binding). Ad esempio, nelle pagine HTML questo risultato può essere ottenuto attraverso la creazione di "isole di dati XML" (XML data island) con elementi HTML standard collegati ad elementi XML[198]. Mentre con la trasformazione XSLT, come si evince appunto dal termine, i dati vengono inseriti, "trasformati", in una nuova struttura, per lo più di rappresentazione, con le tecniche di XML data island si fornisce una maschera, un'interfaccia per l'accesso ai dati sulla scorta, un po', di quanto avviene in ambito database.

Al fine di investigare ed illustrare il più ampio ventaglio di possibilità di rappresentazione dei testi letterari sottoposti a codifica elettronica XML, abbiamo realizzato una piccola interfaccia HTML per la riproduzione su schermo di Baltico sfruttando tecniche di data binding, anche in questo caso in appendice viene riportato integralmente il codice sia della pagina HTML di collegamento ai dati, sia del foglio XSLT che, come vedremo fra poco, si è reso necessario per la modellazione dei dati originari in una nuova struttura idonea all'utilizzo come data island.

La tecnologia XML data island viene adoperata al meglio con dati rigidamente strutturati e schematizzati, sul modello dei database, e proprio precipuamente per l'utilizzo con i database XML è stata concepita, di per sé, quindi, non si presta bene all'adozione con dati non ordinati rigidamente[199], quali quelli di un testo letterario sottoposto a markup dichiarativo, come appunto la versione XML/TEI del romanzo di Collura da noi realizzata. Si è reso, dunque, necessario rimodellare i dati XML/TEI originari in nuova struttura adeguata ad essere usata come data

[198] La maggior parte delle tecniche di XML data island da noi illustrate in questo capitolo, al momento in cui scriviamo, funzionano soltanto con i browser Microsoft Internet Explorer dalla versione 4 in poi.
[199] Per ordinati rigidamente in questo caso intendiamo con una struttura ed una impostazione rigorosamente ordinata e fissa come quella di un database.

island da incorporare nella nostra pagina HTML di interfaccia; per far ciò, anche in questa occasione, non siamo potuti naturalmente prescindere dall'uso di XSLT, che però, come vedremo, in questo caso è stato utilizzato in modo, vorremmo dire, "creativo". Ci teniamo a precisare che quanto da noi realizzato, almeno sulla base delle nostre conoscenze, è il primo esempio, perlomeno per quanto riguarda l'ambito nazionale, di utilizzo di tecniche di XML data island per la presentazione di testi letterari XML/TEI.

L'uso di questa tecnologia è stato dettato anche dal proposito di produrre una interfaccia di visualizzazione del testo nel browser web che, a differenza dell'output HTML di Baltico già descritto, non presentasse un'unica lunga pagina da leggere scorrendo progressivamente il testo, alla stessa stregua di un moderno "volumen" elettronico, bensì una serie di "pagine" da sfogliare come nei "codex".

In effetti, in occasione della prima trasformazione XSLT avremmo potuto fare in modo di realizzare un output costituito da più pagine HTML; tuttavia, al momento attuale dello sviluppo della tecnologia XSLT, che ricordiamo essere ancora alquanto giovane, la produzione di output su più file non si presenta abbastanza agevole adoperando gli strumenti standard: abbiamo, pertanto, deciso di abbandonare questa strada.

Come detto l'obiettivo che ci siamo prefissi usando XML data island è stato quello di ottenere dei ben precisi risultati nella visualizzazione del testo con un'interfaccia, che fosse caratterizzata da una serie di schermate di lunghezza non eccessiva, in modo tale da eliminare o ridurre al minimo la necessità dello scrolling[200], così da emulare, in certo qual modo, su schermo la consueta pratica dello sfogliare le pagine. In realtà, nel concepire il tutto avevamo a mente come modello principale l'interfaccia del Microsoft E-book Reader. Uno dei primi problemi che ci siamo trovati ad affrontare è stato, dunque, rimodellare i dati originari TEI

[200] Lo scrolling è l'operazione di scorrimento dall'alto verso il basso o da sinistra verso destra che si effettua su una finestra il cui contenuto è più lungo o più largo di quanto si possa visualizzare in una sola videata.

in una struttura dall'organizzazione più idonea all'utilizzo con XML data island[201] e stabilire quale dovesse essere la lunghezza dei blocchi di testo adatta alle nostre esigenze; non volevamo, infatti, che ogni pagina eccedesse di troppo l'aria di visualizzazione di una schermata video alla risoluzione di 1024x768[202]. Infine, dopo numerosi tentativi abbiamo deciso di riproporre su schermo la medesima disposizione delle pagine (intesa come blocchi di testo) della nostra fonte cartacea, per far ciò ci siamo giovati dei tag <*pb*> della sintassi TEI che nella codifica sono serviti appunto per registrare i salti pagina nell'edizione fonte.

Questa scelta non si è rivelata di semplice attuazione ed ha richiesto una certa "fantasia", rimodellando i dati per le esigenze di XML data island, nell'uso degli strumenti offerti da XSLT. Come sappiamo <*pb*> è un tag *milestone*, un tag vuoto usato per indicare dove occorrono nel flusso del testo i salti pagina, è sostanzialmente un segnaposto che non racchiude al proprio interno testo o altri elementi come fanno gli altri marcatori, sorgeva dunque il problema di come delimitare a partire dai tag di *page break* le singole pagine in opportuni blocchi di testo; la soluzione è venuta da un uso per così dire "eterodosso" di XSLT.

Passando ad esaminare il foglio di stile XSLT riportato in allegato possiamo notare che sostanzialmente questi non fa altro che trasferire i dati del nostro file XML/TEI in un nuovo file XML dalla struttura assai semplice, composto da un elemento radice di nome <*liber*> il quale possiede un solo elemento figlio <*pag*>, che è il contenitore per i due elementi <*pagina*>, in cui è contenuto il testo della pagina, e <*numero*> dove abbiamo voluto registrare il numero della pagina corrispondente nella fonte cartacea.

Il file si apre con il template dell'elemento radice il quale scrive oltre al tag radice del nuovo XML <*liber*> i marcatori <*pag*> e <*pagina*>, di seguito crea una sorta di piccolo frontespizio elettronico di questo nuovo file prelevando dal TEI Header

[201] Il ricco markup TEI non si presta per nulla bene ad essere usato immediatamente come isola di dati XML, le quali invece hanno bisogno per funzionare di una struttura essenziale e rigidamente organizzata, sul modello, come detto, dei database.
[202] Secondo statistiche recenti questa risoluzione video è la più diffusa tra i navigatori internet.

alcune informazioni: autore, titolo dell'opera, edizione elettronica, responsabili della codifica e della pubblicazione.

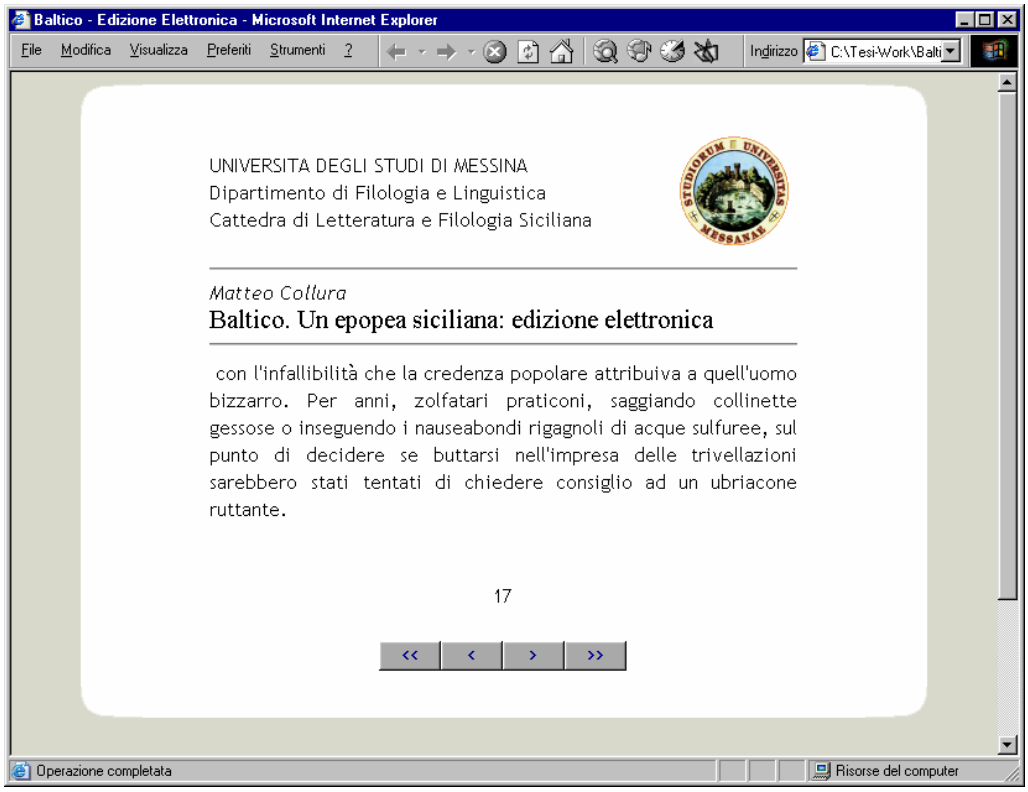
Subito dopo `<xsl:apply-templates/>` invoca gli altri template ed infine il modello chiude il file scrivendo </*pagina*>,</*pag*> e </*liber*>, ciò produce una sorta di sistema ad incastro che combinandosi con il modello successivo per i tag <*pb*> consente il raggiungimento del nostro scopo.

Infatti il template per il tag <*pb*>, che costituisce un po' il fulcro di questa elaborazione, grazie a comandi `<xsl:text>` (necessari per mantenere la buona formazione del file XSLT) ogni qual volta incontra nel sorgente un tag <*pb*> provvede prima a chiudere i tag <*pagina*> e <*pag*> e successivamente dentro un tag <*numero*> manda in output il numero della pagina elaborata (prelevandolo dall'attributo *n* di <*pb*>) dopodiché apre il tag <*pagina*>. Sia il marcatore <*pag*> sia quello <*pagina*> vengono chiusi dall'elaborazione del successivo <*pb*> nel sorgente TEI, infatti ogni <*pb*> nell'elaborazione rappresenta la fine del precedente blocco e contestualmente l'inizio di quello nuovo[203], così facendo, come forse si sarà compreso, siamo riusciti a realizzare un file ben formato in cui le singole pagine dell'edizione fonte (ovvero i corrispondenti blocchi di testo) sono racchiuse dentro coppie di marcatori <*pagina*>...</*pagina*>.

L'interfaccia HTML della sorgente di dati XML da noi creata è concepita in modo tale da visualizzare sequenzialmente le pagine, le quali possono essere sfogliate mediante una serie di pulsanti posti alla base del documento. Sotto il testo viene riportato il numero della pagina, abbiamo optato per indicare il numero della corrispondente pagina dell'edizione fonte piuttosto che ricreare una nuova numerazione per questa versione digitale, ritenendola più utile per effettuare eventuali raffronti e per avere una idea delle caratteristiche di disposizione ed impaginazione dell'edizione originale.

Di seguito riportiamo una schermata dell'interfaccia HTML:

[203] Questo artificio si basa sull'incastro con l'elemento radice che come si può notare è indispensabile per aprire il primo blocco di tag <pag> <pagina> del documento e viceversa chiudere l'ultimo.



Esaminando invece il codice di questa pagina, riportato in appendice, possiamo vedere che si tratta di un semplicissimo file HTML il cui layout principale è definito da una tabella, l'isola di dati XML è richiamata mediante il tag `<XML id="baltico" src="island.xml"/>`, mentre le informazione contenute nel sorgente XML (inserite dentro la tabella) vengono prelevate grazie ai marcatori `<SPAN datasrc="#baltico" datafld="pagina"></SPAN>` e `<SPAN datasrc="#baltico" datafld="numero"></SPAN>`, che come si sarà compreso, fanno riferimento al contenuto dei tag *<pagina>* e *<numero>* della sorgente dati XML da noi precedentemente creata e a cui abbiamo dato il nome di island.xml.

Appendice 6

FOGLIO DI STILE XSLT PER LA RISTRUTTURAZIONE DEL CODICE XML/TEI DI BALTICO PER L'UTILIZZO XML DATA ISLAND

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1"/>
  <xsl:template match="/">
<liber>
<pag>
<pagina>
<xsl:text>&#xD;</xsl:text>
<xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/><xsl:text>&#xD;
</xsl:text>
<xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/><xsl:text>&#xD;
</xsl:text>
<xsl:value-of
select="/TEI.2/teiHeader/fileDesc/editionStmt/edition"/><xsl:text>&#xD
;
</xsl:text>
<xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/respStmt"/><xsl:text>&#xD;
</xsl:text>
Pubblicazione e Distribuzione: <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/publicationStmt/publisher"/><xsl:tex
t>&#xD;
</xsl:text>

<xsl:apply-templates/>
</pagina>
</pag>
</liber>
</xsl:template>

<xsl:template match="pb">
<xsl:text disable-output-escaping="yes">&lt;/pagina&gt;</xsl:text>
<xsl:text disable-output-escaping="yes">&lt;/pag&gt;</xsl:text>
<xsl:text disable-output-escaping="yes">&lt;pag&gt;</xsl:text>
<numero><xsl:value-of select="@n"/></numero>
<xsl:text disable-output-escaping="yes">&lt;pagina&gt;</xsl:text>
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="teiHeader"/>
<xsl:template match="front"/>
<xsl:template match="div0/p"/>
</xsl:stylesheet>
```

Appendice 7

PAGINA HTML PER CON COLLEGAMENTO AI DATI XML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
<HTML>
<HEAD>
<TITLE>Baltico - Edizione Elettronica</TITLE>
<style TYPE="text/css">
  text,p,span,th,td,ul,li { FONT-SIZE: 12pt; FONT-FAMILY: Trebuchet
MS, Verdana, Arial, Helvetica, sans-serif; color: black; background:
white; }

span.titolo{ font-family: "Times New Roman", Times, serif;  font-size:
16pt;  }

  button { azimuth: inherit;  elevation: higher;  color: #00008B;
background: #A9A9A9;  pitch: high;  width: 50px;  font-weight:
lighter;       }
td { text-align: justify; }

</style>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1"></HEAD>

<BODY bgcolor="#D8D9CA" topmargin="10" marginheight="0">
<table width="88%" height="100%" border="0" align="center"
cellpadding="0" cellspacing="0">
  <tr>
    <td valign="top"> <div align="left"><img src="alto-sx.gif"
width="50" height="35"></div></td>
    <td valign="top"> <div align="right"><img src="alto-dx.gif"
width="50" height="35"></div></td>
  </tr>
  <tr>
    <td colspan="2"><table width="70%" height="99%" border="0"
align="center" cellpadding="2
0" cellspacing="0">
        <tr>
         <td width="85%"><p>UNIVERSITA DEGLI STUDI DI MESSINA<br>
        Dipartimento di Filologia e Linguistica<br>
        Cattedra di Letteratura e Filologia Siciliana</p>
         </td>
          <td width="15%"><img src="Logo.png"
alt="Logo Universit&agrave; Messina" width="100" height="100"
align="middle"></td>
        </tr>
        <tr>
          <td colspan="2"><hr>
          <em>Matteo Collura</em><br>
 <span class="titolo">Baltico. Un epopea siciliana: edizione
elettronica</span> <hr></td>
        </tr>
        <tr>
        <td colspan="2"><SPAN datasrc="#baltico"
datafld="pagina"></SPAN></td>
        </tr>
        <tr>
```

```
          <td colspan="2"><div align="center"><SPAN datasrc="#baltico"
datafld="numero"></SPAN></div></td>
        </tr>
        <tr>
          <td colspan="2"><br> <div align="center">
              <BUTTON onClick="baltico.recordset.movefirst()">
&lt;&lt; </BUTTON>
              <BUTTON onClick="baltico.recordset.moveprevious();
                    if (baltico.recordset.BOF)
                        baltico.recordset.movenext()"> &lt; </BUTTON>
              <BUTTON onClick="baltico.recordset.movenext();
                    if (baltico.recordset.EOF)
                        baltico.recordset.moveprevious()"> &gt;
</BUTTON>
              <BUTTON onClick="baltico.recordset.movelast()"> &gt;&gt;
</BUTTON>
            </div></td>
        </tr>
      </table></td>
  </tr>
  <tr>
    <td valign="bottom"><div align="left">
<img src="basso-sx.gif" width="50" height="35"></div></td>
    <td valign="bottom">
<div align="right"><img src="basso-dx.gif" width="50"
height="35"></div></td>
  </tr>
</table>
<XML id="baltico" src="island.xml"/> <br>
<br>
<hr width="70%">
<div align="center">Baltico Edizione elettronica<br>
  &copy; 2004 Universit&agrave; degli Studi di Messina<br>
  Dipartimento di Filologia e Linguistica<br>
  Cattedra di Letteratura e Filologia Siciliana<br>
  <em>Codificatore: Salvatore Salzillo</em> </div>
<hr width="70%">
</BODY>
</HTML>
```

III.15. OEBPS E LIT

Sino ad ora nella trattazione si è discusso della produzione di output (dai dati XML/TEI), destinati ad essere fruiti esclusivamente per mezzo dell'ausilio di un browser web; passiamo adesso a parlare degli output realizzati nei formati e-book, "propriamente detti", per l'edizione digitale del romanzo di Collura, l'OEBPS/LIT ed il PDF.

Intraprendiamo la nostra disamina da quello che, a nostro modesto giudizio, è il formato che, al momento, garantisce una delle esperienze di lettura più gradevoli su di un dispositivo informatico, il Microsoft LIT.

Come sappiamo gli e-book LIT vengono prodotti a partire da un pacchetto OEBPS, lo standard aperto per gli e-book elaborato dall'Open e-Book Forum col fine di favorire lo sviluppo e la diffusione dell'editoria elettronica, pertanto il nostro lavoro è partito dalla creazione di una pubblicazione (pacchetto) conforme agli standard OEBPS che poi abbiamo provveduto a compilare, mediante opportuni strumenti, nel formato Microsoft LIT. In realtà, il pacchetto OEBPS è già di per sé una pubblicazione fruibile autonomamente grazie ad alcuni, pochi per la verità, speciali reader software[204]. Va tuttavia detto che al momento attuale dello sviluppo questi strumenti non denotano una buona efficacia[205] ed una ergonomia paragonabile a quella del lettore Microsoft; inoltre, in taluni casi si sono rivelati poco stabili ovvero il sito che avrebbe dovuto ospitare i file per il download non è stato raggiungibile per lunghi periodi di tempo, per questo motivo abbiamo deciso di non tenerli in considerazione nel nostro lavoro e di puntare principalmente su una delle

[204] I reader OEBPS da noi esaminati sono :
Mentoract Reader: (<http://www.globalmentor.com/software/reader/>);
eMonocle: (<http://www.ionsystems.com/emonocle/>);
NIST reader: (<http://www.itl.nist.gov/div895/download/NISTOEBWin.zip>).

[205] Va tuttavia precisato che a vantaggio di questi software vi è il fatto che siano disponibili per tutti i principali sistemi operativi (Windows, Linux, Macintosch), a differenza del prodotto Microsoft che invece esiste esclusivamente nella versione per Windows.

versioni compilate ottenute a partire da OEBPS[206], appunto il formato LIT di Microsoft.

Come di consueto si è adoperata la tecnologia XSLT per la manipolazione e la trasformazione dei dati XML/TEI in un pacchetto OEBPS, in appendice è riportato il codice dei fogli di stile utilizzati.

Abbiamo già parlato per linee generali delle caratteristiche dello standard OEBPS nella seconda parte di questa relazione, ci accingiamo adesso ad illustrare nello specifico come si sia proceduto per produrre una versione LIT di Baltico a partire dal testo da noi codificato secondo la sintassi XML/TEI.

Come sappiamo un pacchetto OEBPS è costituito da un gruppo di file (pagine HTML, fogli di stile CSS, immagini), collegati fra di loro ed organizzati per mezzo di un file XML detto *package file* e caratterizzato dall'estensione ".opf". Di conseguenza la nostra prima attività è stata la creazione di un foglio XSLT per la produzione del *package file;* di questo si occupa il primo foglio di stile (Lit - opf.xsl) il cui codice è riportato in appendice. Come possiamo vedere l'intero documento opera su di un solo template, quello del nodo radice, e poi mediante riferimenti assoluti XPath estrae dati da altri elementi, principalmente dal TEI Header, e li inserisce nel nuovo file XML (.opf). Compito di questo foglio è quello di creare un file XML (il *package file* ".opf"). All'inizio del documento è indicata la dichiarazione di *doctype* del file di output (+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN) e la DTD di riferimento che come vediamo è quella dei documenti conformi alla versione 1.2 delle specifiche OEBPS (http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd).

Nella prima sezione del documento .opf, il *Package Identity* (<*package*>), l'identificatore univoco della pubblicazione è stato ottenuto combinando il nome

[206] Quello creato dall'Open eBook Forum è un formato "intermedio", uno standard aperto utile per creare una sorta di "sorgente" del libro elettronico che possa poi essere compilato in vari formati proprietari, ad oggi vi è un certo numero di applicazioni, che si basa su versioni compilate di pacchetti OEBPS, che tuttavia non godono del prestigio e della diffusione del prodotto della Microsoft, tra questi meritano di essere ricordati i software della famiglia Mobipocket, destinati soprattutto alle piattaforme dei computer palmari e dei telefoni cellulari avanzati, prodotti dall'omonima azienda finanziata dalla società d'affari Viventures del gruppo Vivendi Universal.

dell'autore con il titolo dell'opera codificata, informazioni prelevate appunto dal TEI Header; per esigenze di compatibilità mediante la funzione *translate()* sono stati rimossi gli spazi tra le parole e sostituiti con degli underscore (_).

L'elemento <*metadata*> del package file ospita, invece tutte le informazioni relative alla pubblicazione OEBPS, come forse il lettore ricorderà, per la definizione dei metadata la specifica OEBPS fa ricorso ad un sottoinsieme del *Dublin Core metadata element set*, nel file XML gli elementi in questione sono identificati dal prefisso di namespace "dc". Anche in questa occasione nella maggior parte dei casi le informazioni sono prelevate semplicemente dalla testata del documento TEI sorgente, soltanto in poche circostanze si è intervenuti manualmente, ovvero in tutti quei casi in cui non vi sia una buona corrispondenza fra metadata TEI e metadata Dublin Core/OEBPS, come ad esempio nel caso dei tag <dc:*Description*> oppure <dc:*Subject*> di cui mancano dei corrispettivi nello schema TEI. La funzione di XSLT si esaurisce con l'elemento <*metadata*> del package file, gli altri elementi infatti servono a combinare ed organizzare i file costituenti il pacchetto OEBPS, non vi è quindi necessità di operare sul sorgente XML.

Il foglio appena illustrato può essere adottato con qualsiasi file conforme alla TEI-Lite per produrre documenti .opf (destinati alla compilazione LIT); l'unica accortezza, che dovrà avere l'eventuale utente, sarà quella di rispettare i nomi dei file del pacchetto definiti nel foglio[207] oppure di modificare nel codice XSLT i nomi dei file.

Una volta realizzato il package file il secondo passo è stato la creazione del file con i contenuti testuali veri e propri, di ciò si occupa il nostro secondo foglio XSLT (Lit - contenuto.xsl). Sostanzialmente il codice di questo foglio ripropone solo con pochi necessari aggiustamenti il codice per la trasformazione HTML di Baltico già visto precedentemente; il dizionario di markup dei documenti testuali OEBPS è costituito da un sottoinsieme dello standard XHTML 1.1, nella sostanza il foglio di

[207] toc.htm per l'indice dei contenuti, contenuto.htm per il contenuto testuale del documento, copyright.htm per le dichiarazioni di copyright ed infine copertinag.jpg e copertinap.jpg per le copertine dell'e-book.

stile che stiamo illustrando provvede appunto a creare un file XHTML conforme agli standard OEBPS.

Il terzo XSLT da noi realizzato (Lit - TOC.xsl) serve a produrre la TOC (Table of Contents) ossia l'indice del documento, la tecnica adottata per la creazione di collegamenti ed ancore, in questo come nel precedente foglio di stile, riprende interamente il sistema già visto in occasione del foglio di stile per l'output HTML, non appare quindi necessario spiegarlo nuovamente, l'unica differenza è nel foglio TOC dove l'indice è creato mediante espressioni `<xsl:for-each>` invece che semplici template. L'ultimo file XHTML che normalmente compone un pacchetto OEBPS è la dichiarazione di copyright dell'opera; in questo caso abbiamo preferito procedere creando manualmente il file con le informazioni sul copyright, in quanto in genere vengono in esso riportate sempre i medesimi dati e poiché alcune delle notizie sui diritti che abbiamo deciso di inserire, come ad esempio le informazioni sulla copertina, non sono presenti nel documento TEI.

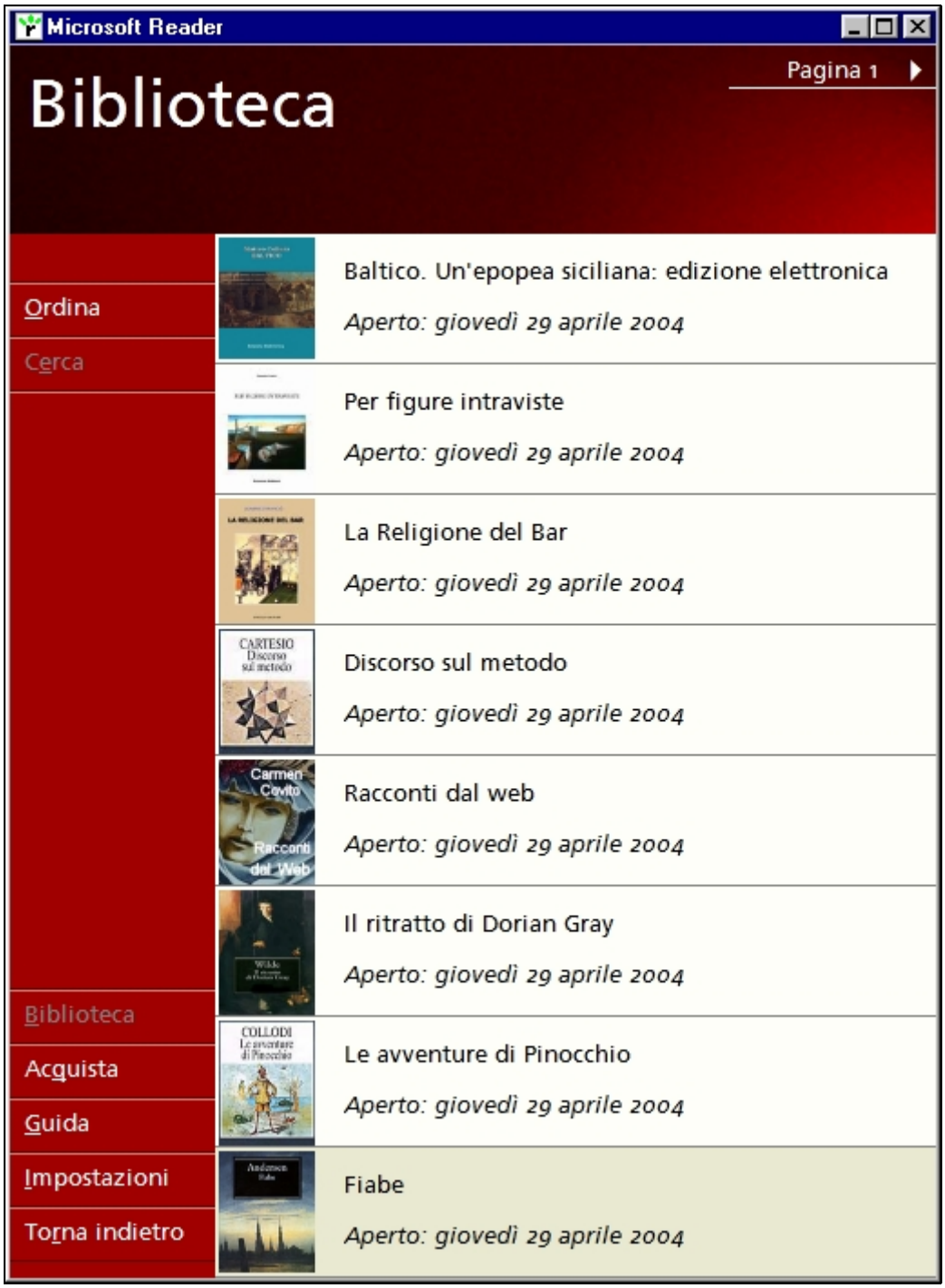
Una volta realizzati i file grafici per le copertine[208] della pubblicazione, il nostro pacchetto OEBPS poteva dirsi pronto; tuttavia, come detto, sin da principio si è previsto che questi venisse fruito non mediante i reader di documenti OEBPS attualmente a disposizione, bensì fosse destinato ad essere trasformato in formato LIT in modo tale da consentirne la visualizzazione grazie al Microsoft Reader. Per la compilazione del nostro pacchetto OEBPS in formato LIT ci siamo giovati di un'applicazione open source[209] gratuita di nome Bulk Lit[210] prodotta dalla Ficsprout[211]. Di seguito riportiamo alcune schermate, che mostrano come venga visualizzata la versione LIT del romanzo di Collura da noi realizzata nel Microsoft Reader: la prima immagine mostra la Biblioteca, ossia l'area del software che funge, per così dire, da Home Page dell'e-book reader e contiene un elenco degli e-book disponibili; dalla Biblioteca è possibile aprire, organizzare ed

[208] Utili soltanto per la versione compilata per il Microsoft Reader.
[209] Il codice sorgente del software è reso disponibile sulla base dei termini della Mozilla Public License.
[210] Questa applicazione si basa su di un componente (litgen.dll) rilasciato dalla Microsoft a beneficio di sviluppatori che intendessero creare software capaci di salvare in formato LIT.
[211] <http://www.ficsprout.com>.

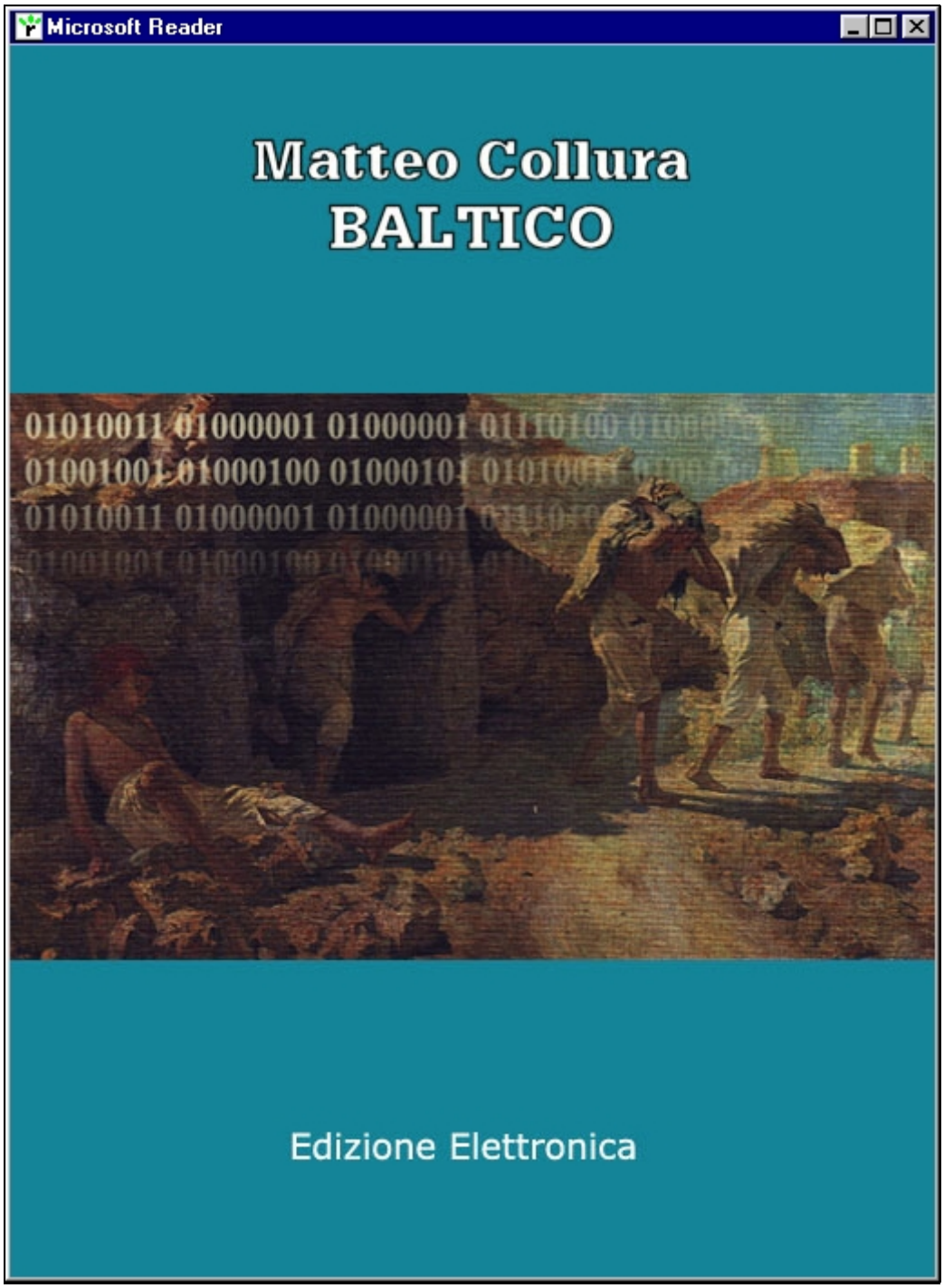
eliminare e-book, Baltico è il primo della lista ed è identificato da una miniatura dell'immagine di copertina e dal titolo del libro elettronico, nel nostro caso: "Baltico. Un'epopea siciliana: edizione elettronica".

La seconda schermata riproduce l'immagine di copertina da noi realizzata per l'occasione rielaborando la copertina originale dell'edizione cartacea.

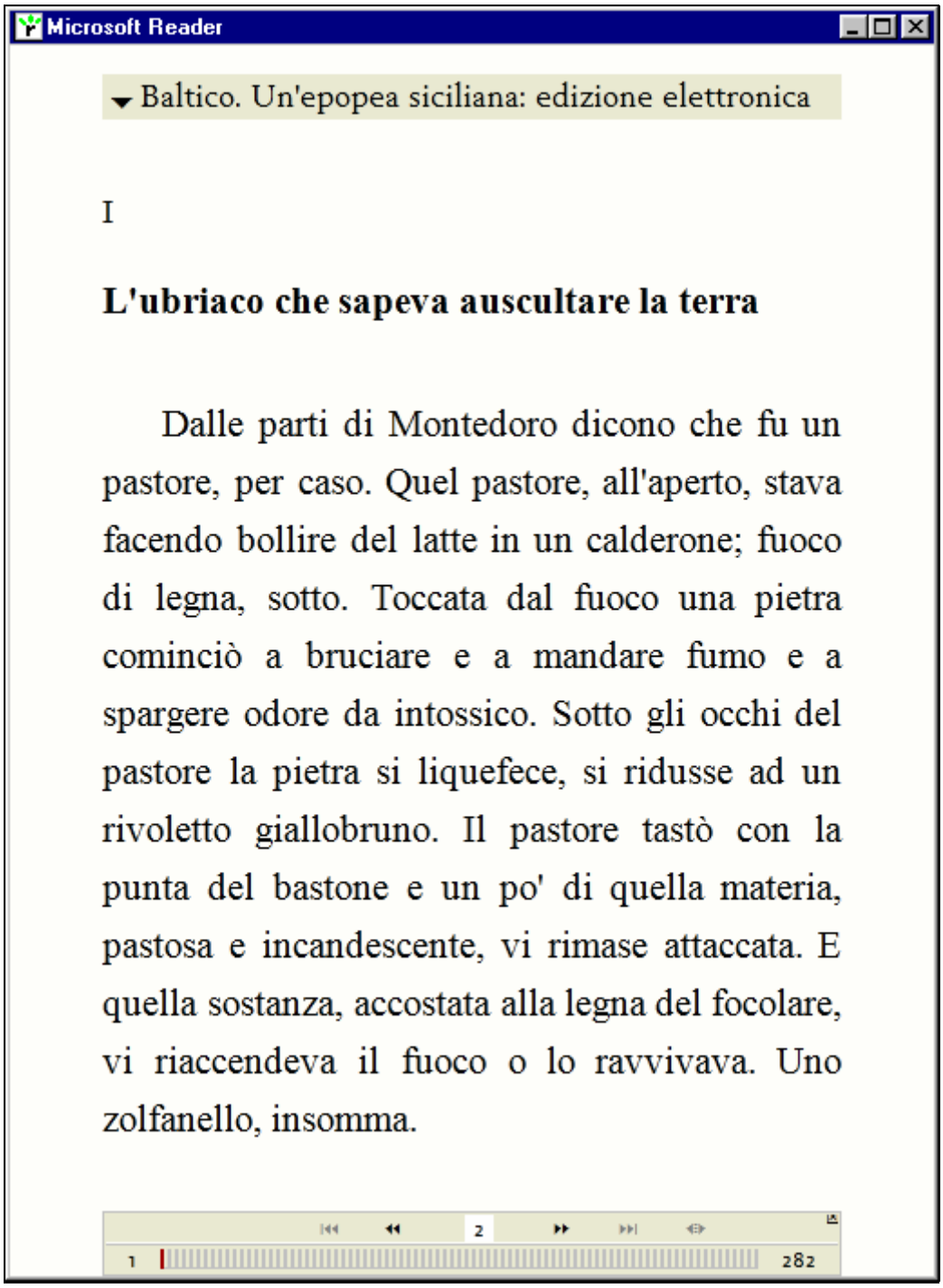
Infine l'ultima immagine mostra la prima pagina del libro così come è visualizzata nel reader.



L'area Biblioteca del Microsoft Reader



La nuova copertina per l'edizione LIT del romanzo.



La prima pagina del romanzo.

Il file LIT da noi prodotto si presta ad essere visualizzato su computer e Tablet PC con sistema operativo Microsoft, oltre che su palmari con sistema operativo Pocket PC.

In origine, durante le prime fasi di elaborazione del nostro progetto di codifica e distribuzione di testi digitali nell'ottica di una eventuale creazione di un archivio elettronico/biblioteca digitale, avevamo preso in considerazione il formato Microsoft LIT oltre che per le sue buone caratteristiche di ergonomia nella visualizzazione su schermo dei testi (layout dinamico, Clear Type), anche per le sue doti nella protezione dei testi. Sebbene con strumenti freeware come quelli da noi utilizzati fosse possibile creare documenti soltanto al più basso livello di protezione "Sealed"[212], l'opportunità di distribuire un testo senza possibilità alcuna che potesse venir modificato ci appariva più che sufficiente per le nostre esigenze di distribuzione per fini accademici e non di sfruttamento economico dei testi. Tuttavia, ormai da alcuni mesi ha visto la luce un piccolo software[213] che permette di bypassare completamente le protezioni del formato Microsoft, l'applicazione in questione provvede a decompilare il file LIT fornendo l'insieme di file, il pacchetto OEBPS originale, da cui è stato prodotto. È da notare che questo programma funziona indifferentemente con tutti i tipi di file LIT, indipendentemente dal tipo di protezione, e quindi anche con gli e-book Inscribed e Owner Exclusive. Quanto appena detto dimostra come sia ancora lungo il cammino da percorre nel campo della protezione dei contenuti digitali.

[212] Sealed (sigillato): è il livello di protezione base applicato al momento della conversione del file, il quale viene criptato in modo tale da assicurarne l'integrità del contenuto. Di un e-book "sigillato", in pratica qualsiasi file LIT, non può essere modificato il testo né qualsiasi altro contenuto. A questo livello di protezione gli e-book possono essere letti da qualsiasi copia del Microsoft Reader anche se quest'ultimo non è attivato, è possibile inoltre copiare tranquillamente il file del libro elettronico.

[213] Onde evitare di contravvenire alle recenti leggi in ambito europeo e nazionale in materia di diritto d'autore evitiamo di fornire il nome e l'indirizzo internet del software in questione, precisiamo tuttavia che il suo reperimento è possibile senza difficoltà con una semplice interrogazione dei motori di ricerca più noti.

Appendice 8

FOGLIO DI STILE XSLT "Lit - opf.xsl" PER LA CREAZIONE DEL PACKAGE FILE DI UNA PUBBLICAZIONE OEBPS A PARTIRE DAI DATI XML/TEI DI BALTICO

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1" doctype-
public="+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN" doctype-
system="http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd" indent="yes"/>
  <xsl:template match="/">

    <!-- l'intero template opera esclusivamente sul nodo radice e
tramite riferimenti assoluti richiama altri elementi, la funzione
translate converte gli spazi in underscores-->

<!—QUESTO ELEMENTO SI SVILUPPA SU DI UN'UNICA RIGA IL SUO CODICE VIENE
SPEZZATO PER ESIGENZE DI STAMPA -->
<package
unique-identifier="{translate(/TEI.2/teiHeader/fileDesc/titleStmt/auth
or,' ','_')}_
{translate(/TEI.2/teiHeader/fileDesc/titleStmt/title,' ','_')}">

<metadata>

<dc-metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:oebpackage="http://openebook.org/namespaces/oeb-package/1.0/">

<!—QUESTO ELEMENTO SI SVILUPPA SU DI UN'UNICA RIGA IL SUO CODICE VIENE
SPEZZATO PER ESIGENZE DI STAMPA -->
<dc:Identifier
id="{translate(/TEI.2/teiHeader/fileDesc/titleStmt/author,' ','_')}_
{translate(/TEI.2/teiHeader/fileDesc/titleStmt/title,' ',' ')}">
  <xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
</dc:Identifier>

<dc:Title>
  <xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
</dc:Title>

<dc:Creator role="aut" file-
as="{/TEI.2/teiHeader/fileDesc/titleStmt/author/name[@type='surname']}
,{/TEI.2/teiHeader/fileDesc/titleStmt/author/name[@type='forename']}">
  <xsl:value-of select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/>
</dc:Creator>

<dc:Subject>Literature/Poetry</dc:Subject>

<!-- Descrizione inserita manualmente in qunato mancante un elemento
analgo nel TEi header -->

<dc:Description>Versione elettronica del romanzo di Matteo Collura
Baltico. Un'epopea sicilian</dc:Description>

<dc:Publisher>
  <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/publicationStmt/publisher"/>
```

```
</dc:Publisher>

<dc:Contributor file-
as="{/TEI.2/teiHeader/fileDesc/titleStmt/respStmt/name[1]}"
role="com">
  <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/respStmt/name[1]"/>
</dc:Contributor>

<dc:Type>
  <xsl:value-of
select="/TEI.2/teiHeader/profileDesc/textClass/keywords/term"/>
</dc:Type>

<dc:Date>
  <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/editionStmt/edition/date"/>
</dc:Date>

<dc:Format>text/html</dc:Format>

<dc:Source><!-- elaborazione condizionale per inserire le virgole ed
il punto finale -->
<xsl:for-each
select="/TEI.2/teiHeader/revisionDesc/change/item/bibl/*">
<xsl:choose>
  <xsl:when test="position() != last()"><xsl:value-of select="."/>,
</xsl:when>
    <xsl:otherwise><xsl:value-of select="."/>.</xsl:otherwise>
  </xsl:choose>
</xsl:for-each>
</dc:Source>

<dc:Language>Italian (Italy) [it]</dc:Language>

<dc:Rights>Copyright 1988 Matteo Collura</dc:Rights>

</dc-metadata>

</metadata>

  <manifest>
    <item id="toc" href="toc.htm" media-type="text/x-oeb1-document"/>
    <item id="contenuto" href="contenuto.htm" media-type="text/x-oeb1-
document"/>
    <item id="copy" href="copyright.htm" media-type="text/x-oeb1-
document"/>
    <item id="cover-standard" href="copertinag.jpg" media-
type="image/jpg"/>
    <item id="thumb-standard" href="copertinap.jpg" media-
type="image/jpg"/>
  </manifest>

  <spine>
      <itemref idref="contenuto"/>
      <itemref idref="toc"/>
  </spine>

  <guide>
    <reference type="toc" title="toc" href="toc.htm"/>
```

```
     <reference type="other.ms-coverimage-standard" title="cover-
standard" href="copertinag.jpg"/>
     <reference type="other.ms-thumbimage-standard" title="thumb-
standard" href="copertinap.jpg"/>
     <reference type="copyright-page" title="copy"
href="copyright.htm"/>
   </guide>

</package>

</xsl:template>
</xsl:stylesheet>
```

Appendice 9

CODICE DEL PACKAGE FILE PRODOTTO DAL FOGLIO DI STILE XSLT "Lit - opf.xsl" OPERANDO SUI DATI XML/TEI DI BALTICO

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE package PUBLIC "+//ISBN 0-9673008-1-9//DTD OEB 1.2
Document//EN" "http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd">
<package unique-
identifier="Matteo_Collura_Baltico._Un'epopea_siciliana:_edizione_elet
tronica">
  <metadata>
    <dc-metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:oebpackage="http://openebook.org/namespaces/oeb-package/1.0/">
      <dc:Identifier
id="Matteo_Collura_Baltico._Un'epopea_siciliana:_edizione_elettronica"
>Matteo_ColluraBaltico._Un'epopea_siciliana:_edizione_elettronica</dc:
Identifier>
      <dc:Title>Baltico. Un'epopea siciliana: edizione
elettronica</dc:Title>
      <dc:Creator role="aut" file-as="Collura,Matteo">Matteo
Collura</dc:Creator>
      <dc:Subject>Literature/Poetry</dc:Subject>
      <dc:Description>Versione elettronica del romanzo di Matteo
Collura Baltico. Un'epopea sicilian</dc:Description>
      <dc:Publisher>Universitá degli Studi di Messina, Facoltá di
Lettere e Filosofia, Dipartimento di Filologia e Linguistica, Cattedra
di Letteratura e Filologia Siciliana</dc:Publisher>
      <dc:Contributor file-as="Salvatore Salzillo"
role="com">Salvatore Salzillo</dc:Contributor>
      <dc:Type>NARRATIVA ITALIANA. 1945 -1999</dc:Type>
      <dc:Date>????2004</dc:Date>
      <dc:Format>text/html</dc:Format>
      <dc:Source>Matteo Collura, Baltico : un'epopea siciliana, Luigi
Reverdito Editore, 1988.</dc:Source>
      <dc:Language>Italian (Italy) [it]</dc:Language>
      <dc:Rights>Copyright 1988 Matteo Collura</dc:Rights>
    </dc-metadata>
  </metadata>
  <manifest>
    <item id="toc" href="toc.htm" media-type="text/x-oeb1-document"/>
    <item id="contenuto" href="contenuto.htm" media-type="text/x-oeb1-
document"/>
    <item id="copy" href="copyright.htm" media-type="text/x-oeb1-
document"/>
    <item id="cover-standard" href="copertinag.jpg" media-
type="image/jpg"/>
    <item id="thumb-standard" href="copertinap.jpg" media-
type="image/jpg"/>
  </manifest>
  <spine>
    <itemref idref="contenuto"/>
    <itemref idref="toc"/>
  </spine>
  <guide>
    <reference type="toc" title="toc" href="toc.htm"/>
    <reference type="other.ms-coverimage-standard" title="cover-
standard" href="copertinag.jpg"/>
```

```
    <reference type="other.ms-thumbimage-standard" title="thumb-
standard" href="copertinap.jpg"/>
    <reference type="copyright-page" title="copy"
href="copyright.htm"/>
  </guide>
</package>
```

Appendice 10

FOGLIO DI STILE XSLT PER PRODUZIONE DEI CONTENUTI TESTUALI DEL PACCHETTO OEBPS

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1" doctype-
public="+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN" doctype-
system="http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd"/>
  <xsl:template match="/">
    <html lang="it" xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/> - <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
        </title>
        <style>
p { text-align: justify;
  text-indent: 25pt;
  margin: 0pt 0pt;}

.cap { page-break-before: always; }

h2 { font-size: 16pt; text-align: left; font-weight: bold; }

h1 {font-size: 14pt; text-align: left; }

cite { font-style: normal; }

.italic {font-style: italic;}

.center { text-align: center; }

.hr { width: 65%; }

.bloc { line-height: 1.5em;
    margin-left: 10%;
    margin-right: 10%; }

blockquote {text-align: justify;
            line-height: 1.5em;}

body {  font-family:"Times New Roman", Times, serif; line-height:
1.5em;}

p.autore {
  font-variant:small-caps;
  text-align:center;
  font-style:normal;
  font-size:18pt;
  font-family:Serif;
  font-stretch: wider;
  page-break-before:always;
  margin-top:10%;
  text-indent:0px;}
```

```
p.titolo {
   font-variant:small-caps;
   text-align:center;
   font-weight: bold;
   font-style:normal;
   font-size:22pt;
   font-family:Serif;
   margin-top:20%;
   margin-left: 5%;
   margin-right: 5%;
   text-indent:0px;}

p.publ {
   font-variant:small-caps;
   text-align:center;
   font-style:normal;
   font-size:14pt;
   font-family:Serif;
   margin-top:40%;
   text-indent:0px;}
</style>
       </head>
       <body>
<p class="autore"><xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/></p>
<p class="titolo"><xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/></p>
<p class="publ"><xsl:value-of
select="/TEI.2/teiHeader/fileDesc/publicationStmt/publisher"/></p>
         <xsl:apply-templates/>
       </body>
     </html>
  </xsl:template>
  <xsl:template match="teiHeader"/>
  <xsl:template match="text">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="front"/>
  <xsl:template match="body">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="div0">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="div0/p"/>
  <xsl:template match="div1">
    <div class="cap" id="{@id}">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="head[@type='ord']">
    <h1>
      <xsl:apply-templates/>
    </h1>
  </xsl:template>
  <xsl:template match="head[@type='tem']">
    <h2>
      <xsl:apply-templates/>
    </h2>
    <br/>
```

```
  </xsl:template>
  <xsl:template match="head[@type='conv']">
    <h2>
      <xsl:apply-templates/>
    </h2>
    <br/>
  </xsl:template>
  <xsl:template match="p">
    <p>
      <xsl:apply-templates/>
    </p>
  </xsl:template>
  <xsl:template match="div[@type='nota']">
    <div class="cap" id="{@id}">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="q[@type='citazione']" priority="1">
    <xsl:choose>
      <xsl:when test="@rend='bloc'">
        <blockquote>
          <xsl:apply-templates/>
        </blockquote>
      </xsl:when>
      <xsl:when test="@rend='italic'">
        <cite class="italic">
          <xsl:apply-templates/>
        </cite>
      </xsl:when>
      <xsl:otherwise>
        <cite>
          <xsl:apply-templates/>
        </cite>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <xsl:template match="l">
    <xsl:apply-templates/>
    <br/>
  </xsl:template>
  <xsl:template match="lb">
    <br/>
  </xsl:template>
  <xsl:template match="*[@rend='bloc']">
    <div class="bloc">
      <xsl:apply-templates/>
    </div>
  </xsl:template>
  <xsl:template match="*[@rend='italic']">
    <span class="italic">
      <xsl:apply-templates/>
    </span>
  </xsl:template>
</xsl:stylesheet>
```

Appendice 11

FOGLIO DI STILE PER LA CREAZIONE DELLA *TABLE OF CONTENTS* (TOC) DEL PACCHETTO OEBPS

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1" doctype-
public="+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN" doctype-
system="http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd" indent="yes"/>
  <xsl:template match="/">
    <html lang="it" xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title>
          <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/author"/> - <xsl:value-of
select="/TEI.2/teiHeader/fileDesc/titleStmt/title"/>
        </title>
        <style>
 .center { text-align: center; }
 a { color:black; font-size: 13pt;  text-align: left;  font-family:
"Times New Roman", Times, serif;  text-decoration: none;    line-
height: 1em; }
a:hover { color: #1E20FF; }
.part {
font-size: 15pt;  font-family: "Times New Roman", Times, serif;
 font-style: italic;
 text-align: center;
 font-weight: normal;}
body {  font-family:"Times New Roman", Times, serif; }
</style>
      </head>
      <body>
<h1 class="center">INDICE</h1>

<xsl:for-each select="//div0">
      <br/><h2 class="part"><xsl:value-of select="p"/></h2>
    <xsl:for-each select="*/head">
        <a href="contenuto.htm#{../@id}"><xsl:value-of
select="."/></a><br/>
    </xsl:for-each>
</xsl:for-each>

<xsl:for-each select="//div">
    <xsl:for-each select="head">
      <br/>
  <a href="contenuto.htm#{../@id}"><xsl:value-of select="."/></a>
    </xsl:for-each>
</xsl:for-each>
      </body>
    </html>
  </xsl:template></xsl:stylesheet>
```

Appendice 12

CODICE XHTML DELLA TOC DEL PACCHETTO OEBPS PRODOTTO DAL FOGLIO DI STILE "Lit - TOC.xsl" OPERANDO SUI DATI XML/TEI DI BALTICO

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

<!DOCTYPE html
  PUBLIC "+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN"
"http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
   <head>
      <title>Matteo Collura - Baltico. Un'epopea siciliana: edizione
elettronica</title>
      <style>
 .center { text-align: center; }

 a { color:black; font-size: 13pt;  text-align: left;  font-family:
"Times New Roman", Times, serif;  text-decoration: none;    line-
height: 1em; }

a:hover { color: #1E20FF; }

.part {
font-size: 15pt;  font-family: "Times New Roman", Times, serif;
 font-style: italic;
 text-align: center;
 font-weight: normal;}

body {  font-family:"Times New Roman", Times, serif; }
</style>
   </head>
   <body>
      <h1 class="center">INDICE</h1>
      <br/>
      <h2 class="part">PARTE PRIMA</h2>
      <a href="contenuto.htm#I.1">I</a>
      <br/>
      <a href="contenuto.htm#I.1">L'ubriaco che sapeva auscultare la
terra</a>
      <br/>
      <a href="contenuto.htm#I.2">II</a>
      <br/>
      <a href="contenuto.htm#I.2">Quando e come i contadini
incubarono il malanno</a>
      <br/>
      <a href="contenuto.htm#I.3">III</a>
      <br/>
      <a href="contenuto.htm#I.3">Il mare, soffrendo,
diede alla luce un'isoletta</a>
      <br/>
      <a href="contenuto.htm#I.4">IV</a>
      <br/>
      <a href="contenuto.htm#I.4">Perchè Dalla poco attraente Grotte
i viandanti giravano al largo</a>
      <br/>
      <a href="contenuto.htm#I.5">V</a>
```


Il gigantesco Bartolomeo
e la sua smania di viaggiare

VI

Una lettera giunse dall'America

VII

Pur mangiando carne
gli zolfatari diventarono incartapecoriti

VIII

Il primo ritorno del Viaggiatore

IX

Disseppellirono un cadaverino
dagli occhi incrostati

X

Di un elegante cavaliere
e della sua misteriosa missione

XI

Perchè la piazza tornò a brulicare
di sfaccendati

XII

Quelle che avvistò l'equivoco
mediatore, erano navi armate di cannoni

XIII

Di come, con uno scatto di tigre,
Bartolomeo Ardito afferrò la sua donna

XIV

Un alito di bestia malata infettò
la terra

XV

I pericolosi amori
di un carrettiere di Girgenti

XVI

Il mondo traslocava in America

XVII

<a href="contenuto.htm#I.17">Il secondo ritorno del Viaggiatore
e

```
 la sua ultima visione</a>
      <br/>
      <br/>

      <h2 class="part">PARTE SECONDA</h2>
      <a href="contenuto.htm#II.1">I</a>
      <br/>
      <a href="contenuto.htm#II.1">Tre colori messi insieme
dall'audacia</a>
      <br/>
      <a href="contenuto.htm#II.2">II</a>
      <br/>
      <a href="contenuto.htm#II.2">Dove giovani imbottiti di scienza
sono visti come fumo negli occhi</a>
      <br/>
      <a href="contenuto.htm#II.3">III</a>
      <br/>
      <a href="contenuto.htm#II.3">La sfrontatezza dei sacrileghi
fece arrossire vallate e colline</a>
      <br/>
      <a href="contenuto.htm#II.4">IV</a>
      <br/>
      <a href="contenuto.htm#II.4">Un carrettiere pretese
di umiliare il progresso</a>
      <br/>
      <a href="contenuto.htm#II.5">V</a>
      <br/>
      <a href="contenuto.htm#II.5">Dove si prende atto
della complicità degli eucalipti</a>
      <br/>
      <a href="contenuto.htm#II.6">VI</a>
      <br/>
      <a href="contenuto.htm#II.6">Ma scoprirono che neanche i santi
dei signori facevano una piega</a>
      <br/>
      <a href="contenuto.htm#II.7">VII</a>
      <br/>
      <a href="contenuto.htm#II.7">Un uomo sommariamente vestito
sbandierò un quadernetto</a>
      <br/>
      <a href="contenuto.htm#II.8">VIII</a>
      <br/>
      <a href="contenuto.htm#II.8">La bizzarra mania
di un soldato mancato</a>
      <br/>
      <a href="contenuto.htm#II.9">IX</a>
      <br/>
      <a href="contenuto.htm#II.9">Dove si riferisce di
una chiacchierata serale tra notabili</a>
      <br/>
      <a href="contenuto.htm#II.10">X</a>
      <br/>
      <a href="contenuto.htm#II.10">Di quando una veterana maestrina
fece sussultare il suo enorme seno</a>
      <br/>
      <a href="contenuto.htm#II.11">XI</a>
      <br/>
      <a href="contenuto.htm#II.11">Grugniva, il Primo Premio,
trascinato in giro</a>
      <br/>
      <a href="contenuto.htm#II.12">XII</a>
```

```
      <br/>
      <a href="contenuto.htm#II.12">Un compromettente viaggio a
Palermo</a>
      <br/>
      <a href="contenuto.htm#II.13">XIII</a>
      <br/>
      <a href="contenuto.htm#II.13">La tremarella contagiò
i sediziosi compromessi</a>
      <br/>
      <br/>
      <a href="contenuto.htm#nota">NOTA</a>
   </body>
</html>
```

Appendice 13

CODICE XHTML DEL FILE CONTENENTE LE DICHIARAZIONI DI COPYRIGHT DEL PACCHETTO OEBPS

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

<!DOCTYPE html
  PUBLIC "+//ISBN 0-9673008-1-9//DTD OEB 1.2 Document//EN"
"http://openebook.org/dtds/oeb-1.2/oebdoc12.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
   <head>
      <title>Matteo Collura - Baltico. Un'epopea siciliana: edizione
elettronica</title>
      <style>
 .center { text-align: center; }
p
{font-style:normal;
text-indent:0px;}

body {  font-family:"Times New Roman", Times, serif; }

</style>
   </head>
   <body>
<h2 class="center">Copyright</h2>
<br/><p>Matteo Collura</p><br/>
<p>Baltico: un'Un'epopea siciliana</p> <br/>
<p>In Copertina Onofrio Tomaselli<br/>"I carusi [particolare
rielaborato]" <br/>(Civica galleria d'Arte Moderna, Palermo)</p><br/>
<p>&#169; 1988 Luigi Reverdito Editore</p><br/>
<p>ISBN 88-342-0229-5</p><br/>
<p>
Edizione Elettronica:<br/> Università degli Studi di Messina, Facoltà
di Lettere e Filosofia, Dipartimento di Filologia e Linguistica,
Cattedra di Letteratura e Filologia Siciliana
</p>

   </body>
</html>
```

III.16. PDF E PRINT ON DEMAND

Nel capitolo dedicato al Print-on-Demand nella seconda parte della nostra relazione, abbiamo affermato che probabilmente, almeno per il prossimo futuro, sarà ancora la carta il supporto di lettura principale per i libri, elettronici e non; a riprova di ciò vi è il dato che vede crescere in tutti i paesi industrializzati il consumo totale di carta, crescita dei consumi che ovunque sembra riferirsi soprattutto alla carta formato A4, quella usata da stampanti e da fotocopiatrici. Il boom degli home computer e di internet, associato con lo sviluppo dei dispositivi di stampa personale, sembra proprio che, paradossalmente, abbia favorito la crescita del consumo mondiale di carta invece che rallentarlo: ciò dimostra come nel mondo digitale il testo digitale (fruizione ed archiviazione) sia ancora ben lungi dall'essere una realtà pienamente consolidata e diffusa.

Sulla base di queste considerazioni anche per il nostro lavoro sul romanzo di Collura abbiamo deciso di prevedere la possibilità di una "materializzazione cartacea" del testo elettronico di Baltico, abbozzando una sorta di applicazione per la stampa su richiesta (print on demand) basata sul formato Adobe PDF.

PDF è stato il pioniere dei formati per la distribuzione dei testi digitali e rappresenta oggi uno standard per i documenti elettronici ampiamente affermato; essendo figlio di Postscript, il linguaggio di descrizione della pagina sviluppato dalla Adobe, PDF esprime il meglio delle proprie potenzialità proprio con quelle applicazioni finalizzate alla stampa, pur mantenendo una buona resa a video. Nel corso della sua evoluzione il PDF ha visto una sua incarnazione come formato e-book, si ricordi l'esperienza Glassbook, che si è conteso a lungo la palma di miglior tecnologia per i libri elettronici con il rivale Microsoft LIT. Nel nostro lavoro, tuttavia, abbiamo deciso di non considerare primariamente la tecnologia Adobe da questo punto di vista, ma piuttosto come formato finale destinato alla stampa, questo per due ordini di motivi, il primo di natura economica, il secondo di

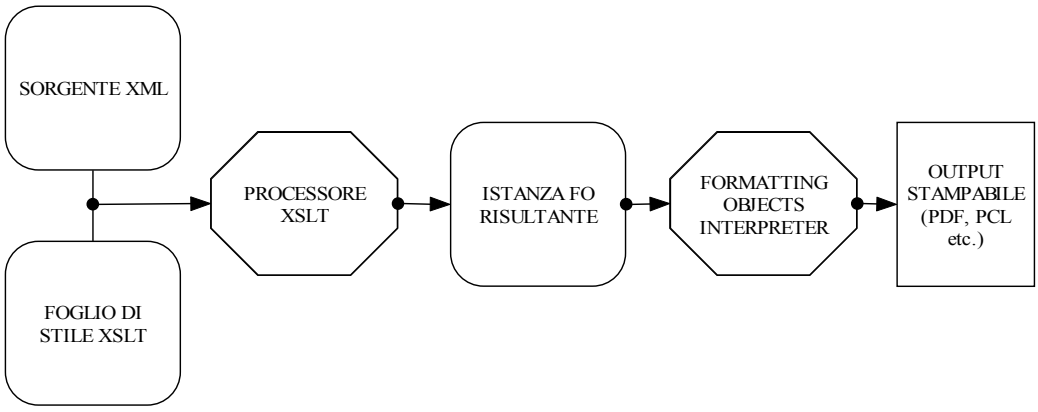
natura pratica. Gli e-book PDF dispongono, o almeno dovrebbero, di particolari caratteristiche quali la presenza di tag[214] oppure di funzioni avanzate di navigazione del documento, che è possibile, al momento, aggiungere ai file soltanto grazie a costosi software commerciali. D'altra parte, la creazione di un e-book PDF curato nei particolari richiede in genere una buona dose di lavoro "manuale" da parte dell'assemblatore del libro digitale (definizione tag, copertina, salti pagina, ridefinizione layout e formattazione etc.) allorché ci si trovi a lavorare, come nel nostro caso, partendo da un testo codificato in XML.

Pertanto, avendo nel nostro progetto optato per l'adozione di tecnologie di elaborazione (il più possibile) automatica dei documenti per la produzione dei vari output di lettura del testo codificato e soprattutto a basso costo, abbiamo deciso di porre in secondo piano la realizzazione di una versione PDF di Baltico indirizzata principalmente alla fruizione su schermo.

Lo strumento, di cui ci siamo avvalsi per la definizione di un output stampabile dai dati XML del romanzo di Collura, è stato la tecnologia XSL-FO. L'Extensibile Stylesheet Language Formatting Objects (XSL-FO) è un vocabolario per specificare la semantica di formattazione, sviluppato congiuntamente ad XSLT, costituisce insieme ad XPath uno dei tre componenti di cui è composta la tecnologia XSL. Se si ricorda quanto detto nella prima parte della relazione, XSL-FO nasce per essere utilizzato in accoppiata con XSLT, mentre il compito di XSLT è quello di trasferire, trasformare, i dati XML originari in una nuova struttura adatta alla riproduzione degli stessi, la funzione di XSL-FO è quella di definire le regole di formattazione di questa nuova struttura e sostanzialmente di fornire la struttura in cui trasferire i dati per mezzo di XSLT. I documenti XSL-FO vengono utilizzati in accoppiata con il Formatting Object Processor (FOP), una applicazione software la quale si occupa di effettuare il rendering (la formattazione grafica) del documento e produrre un

[214] I file PDF con tag posseggono delle informazioni aggiuntive sui contenuti del file, che aggiungono alle caratteristiche rappresentazionali della tecnologia Adobe alcune delle qualità specifiche delle codifiche dichiarative, offrendo così una migliore fruizione su dispositivi elettronici dei testi. Permettono ad esempio una migliore indicizzazione dei documenti ed il reflow (una sorta di layout dinamico) delle pagine.

nuovo documento di output adatto alla stampa oppure alla visualizzazione. L'attività di XSL per la produzione di output usando oggetti di formattazione (Formatting Object) consta di una serie di passaggi: dapprima il processore XSLT applica all'istanza XML sorgente le regole per la trasformazione del sorgente in una nuova struttura XSL-FO, la quale successivamente viene elaborata dal Formatting Object Processor che legge l'istanza FO risultante dall'elaborazione XSLT e produce documenti di output in formati come PDF, PCL (Printer control Language), testo o altro in base alle esigenze di agenti utente esterni. Lo schema seguente potrà aiutare a comprendere meglio i vari passaggi del processo appena illustrato:



Dopo questa breve introduzione generale all'argomento passiamo ad illustrare come si è operato nello specifico per il nostro progetto; prima di procedere è, tuttavia, opportuna una precisazione. Il TEI Consortium ha messo a disposizione sul proprio sito una serie di fogli XSLT elaborati da Sebastian Rahtz per la trasformazione di istanze XML/TEI in documenti PDF; la soluzione realizzata dallo studioso inglese è basata sull'utilizzo di Passivetex, un sistema che adopera XSL-FO per produrre a partire da dati XML documenti PDF attraverso il software di compilazione per il linguaggio TeX, LaTeX. È necessario a questo punto fornire alcune informazioni al lettore; TeX è un linguaggio di programmazione per la composizione tipografica, ideato nel 1977 da Donald Knuth, un professore della Stanford University. La "compilazione" di un sorgente TeX produce un file in formato finale per la stampa. TeX viene soprattutto utilizzato per la pubblicazione di testi di carattere matematico-scientifico per le sue doti di riproduzione di

formule e grafici anche molto complessi. Da principio il linguaggio non ebbe una grande diffusione a causa della sua complessità, in un primo momento, il suo utilizzo rimase limitato principalmente a tipografi e programmatori esperti. Tuttavia, nel 1985 Leslie Lamport sviluppò *LaTeX*, una raccolta di macro scritte in TeX che consentendo un approccio semplificato al linguaggio TeX ne favorirono la diffusione e lo sviluppo.

XMLTeX è un sistema per comporre file XML attraverso TeX, normalmente viene affiancato a LaTeX per trasformare i file XML in un documenti pronti per la stampa, attraverso un metodo di trasformazione degli elementi e degli attributi del file di origine in comandi di LaTeX. Realizzato da Sebastian Rahtz, PassiveTeX è un insieme di fogli di stile aggiuntivi, che in particolare consentono a XMLTeX (o altri compositori TeX) di elaborare file XSL-FO. Nel nostro progetto abbiamo deciso di non adottare la soluzione proposta nel sito del TEI Consortium, in primo luogo a causa della nostra scarsa conoscenza di TeX e dei programmi ad esso collegati, in secondo luogo poiché si è stimato che potesse esser utile proporre alla comunità degli studiosi una soluzione alternativa a quella offerta dal Tei Consortium e Sebastian Rahtz basata su di un software anch'esso gratuito e dalla minore difficoltà di installazione ed implementazione rispetto a PassiveTeX.

Il Formatting Object Processor adottato per il nostro progetto è stato Apache FOP 0.20.5, un'applicazione Java open source distribuita gratuitamente via Internet dalla Apache Software Foundation[215].

Le specifiche XSL-FO definiscono una collezione di oggetti, i formatting objects appunto, mediante i quali controllare l'impaginazione ed il layout generale dei documenti, questi costrutti sono dotati di una cospicua serie di proprietà, sotto forma di coppie "`attributo=valore`", con cui descrivere nel dettaglio tutte le caratteristiche di rappresentazione dell'oggetto (posizionamento, dimensioni, font, colore, etc.), una buona parte delle proprietà di formattazione usate in XSL-FO sono state riprese direttamente dalla specifica dei CSS (Cascading Style Sheet),

[215] <http://xml.apache.org/fop/>.

tuttavia i FO hanno capacità di rappresentazione nettamente superiori agli stili CSS.

La struttura base di un file XSL-FO è la seguente:

```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">

      <fo:layout-master-set>
        <fo:simple-page-master master-name="pagina_mastro">
        <fo:region-body />
        </fo:simple-page-master>
      </fo:layout-master-set>

      <fo:page-sequence master-reference="pagina_mastro">
       <fo:flow flow-name="xsl-region-body">
         <fo:block>testo</fo:block>
         <fo:block>testo</fo:block>
       </fo:flow>
      </fo:page-sequence>

</fo:root>
```

Essendo un file XML, ogni documento XSL-FO si apre con la consueta dichiarazione XML.

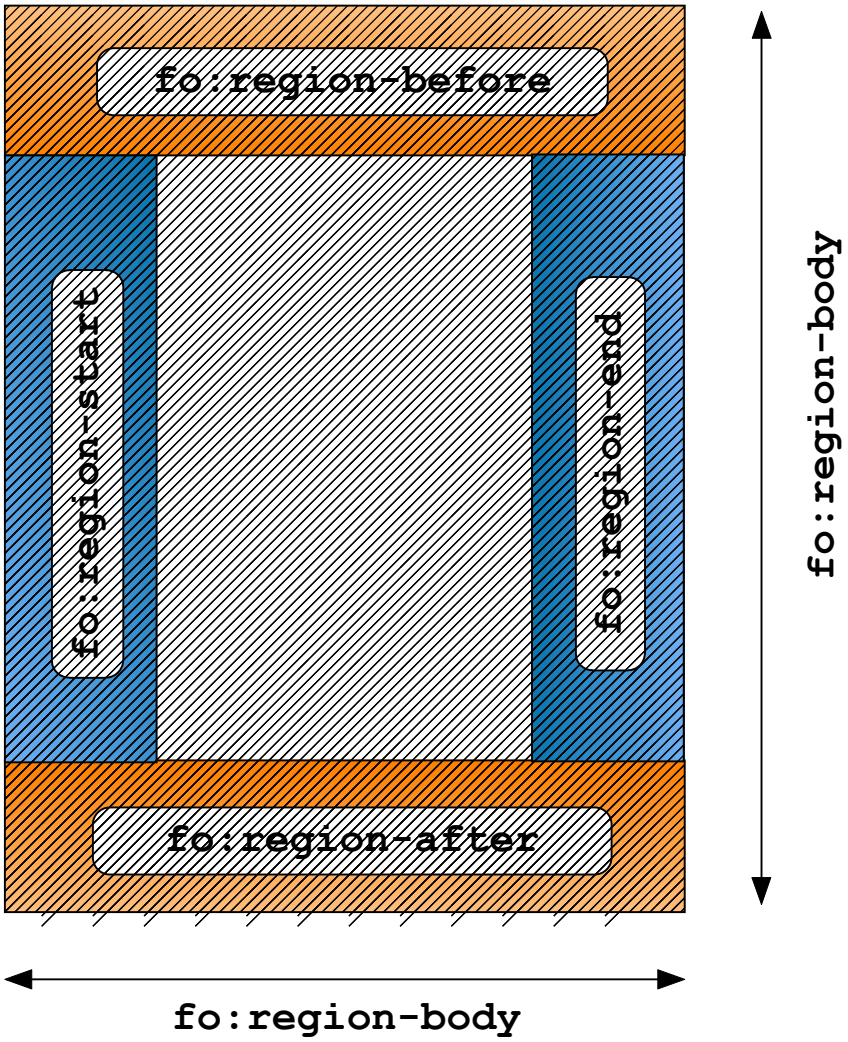
`<fo:root>` è l'elemento radice dell'albero del documento e costituisce il contenitore per tutti gli altri oggetti; dentro l'elemento `<fo:layout-master-set>` vengono definiti mediante elementi `<fo:simple-page-master>` i modelli di impaginazione, le "pagine mastro"[216], del documento. Gli elementi `<fo:simple-page-master>` devono essere associati ad un attributo master-name che si può codificare con un valore a scelta e serve a fornire un nome univoco all'elemento; il nome è necessario per poter distinguere più master uno dall'altro, è inoltre indispensabile per permettere ai singoli gruppi di pagine (`<fo:page-sequence>`) di

[216] Nel gergo tipografico viene definita pagina mastro quella pagina che costituisce la gabbia, il modello di impaginazione, contenente caratteristiche ed elementi comuni a tutte le pagine cui viene applicata la pagina mastro.

far riferimento al modello di impaginazione definito nel <fo:simple-page-master> mediante l'attributo master-reference.

All'interno dei margini fissati con gli attributi appropriati nell'elemento <fo:simple-page-master> si collocano delle regioni dichiarate attraverso gli elementi <fo:region-before>, <fo:region-after>, <fo:region-start>, <fo:region-end> e <fo:region-body>[217],questo ultimo elemento è l'unico obbligatorio, l'area definita dall'elemento <fo:region-body> contiene, sovrapposte, le altre quattro regioni corrispondenti alla testata, al piè di pagina ed ai margini destro e sinistro del corpo del documento. La seguente immagine mostra graficamente quanto appena detto:

[217] Si noti che i nomi delle regioni, rappresentano una collocazione relativa all'orientamento della scrittura in cui è redatto il documento; ad esempio, una scrittura che procede da destra verso sinistra fa sì che si scambi la collocazione delle regioni dichiarate dagli elementi <fo:region-start> e <fo:region-end>; mentre per una
scrittura che procede dal basso verso l'alto, si scambiano di posto le regioni <fo:region-before> e <fo:region-after> rispetto alla figura.



L'elemento `<fo:page-sequence>` contiene la definizione delle informazioni contenute nelle pagine che costituiranno il documento finale. Questo elemento rappresenta, come si evince dal nome, una sequenza di pagine, quali ad esempio un capitolo o una parte di un libro. Pertanto, un documento può avere più elementi `<fo:page-sequence>` distinti. L'attributo obbligatorio `master-reference` serve a specificare il riferimento al tipo di impaginazione, ossia alla "pagina mastro", alle regole di layout definite nell'elemento `<fo:simple-page-master>` collegato. L'elemento `<fo:page-sequence>` deve contenere necessariamente un unico elemento `<fo:flow>`, un oggetto contenitore che racchiude tutto ciò, testo ed altro, che viene distribuito nelle pagine. L'elemento `<fo:flow>` deve dichiarare, attraverso l'attributo `flow-name`, in quale regione della pagina si inserisce il flusso di contenuti in questione. Dentro `<fo:flow>` i contenuti sono ripartiti e strutturati

mediante elementi "blocco", il principale di questi è l'elemento `<fo:block>` la cui funzione è quella di contenere sia testo lineare, sia altri blocchi (compresi altri elementi `<fo:block>`). Viene in genere adoperato per racchiudere i singoli paragrafi, può essere considerato come un equivalente dell'elemento <*p*> o dell'elemento <*div*> di HTML.

Passando ad esaminare il foglio di stile[218] per la trasformazione in XSL-FO del testo codificato in XML di Baltico, si nota che nei vari modelli viene riprodotta la struttura dei documenti XSL-FO appena illustrata[219]. Nel template del nodo radice, da cui come di consueto prende l'avvio l'elaborazione, troviamo oltre all'elemento radice delle istanze XSL-FO (`<fo:root>`) anche la definizione delle "pagine mastro" del documento (`<fo:layout-master-set>`); come si può osservare abbiamo definito un solo modello di impaginazione (`<fo:simple-page-master>`) per il nostro file, le regole di layout definite nell'elemento stabiliscono che le pagine del nostro documento dovranno avere un'altezza di 29.7cm ed una larghezza di 21cm, ossia le dimensioni di un foglio A4. Si è optato per dimensioni corrispondenti al formato A4, in quanto questo è il formato di pagina comunemente adoperato da stampanti e fotocopiatrici. Come detto, uno dei nostri obiettivi è stato quello di realizzare un rudimentale sistema di "print on demand", in cui i dati elettronici dell'archivio digitale potessero essere all'occorrenza "materializzati", stampati, anche con risorse esigue (un computer ed una stampante). Potrà sembrare pretenziosa la definizione di applicazione per il print-on-demand per questo lavoro che sostanzialmente non fa altro che realizzare dei file per la stampa da dei dati XML. Si tenga però presente che adottando le medesime tecnologie che in queste pagine stiamo via via illustrando è ipotizzabile, la creazione di un elementare box per la stampa dei testi digitali di un potenziale archivio elettronico; si ricordi che

[218] In appendice si sono riportati sia il codice del foglio XSLT per la trasformazione in XSL-FO di Baltico, sia l'output XSL-FO sia alcune pagine stampate sulla base del file PDF ottenuto dalla conversione del documento XSL-FO usando Apache FOP.

[219] Per comprender meglio il commento sul lavoro di trasformazione in XSL-FO del codice XML di Baltico, consigliamo al lettore di esaminare oltre al foglio di stile XSLT anche l'output XSL-FO, in cui più chiaramente si può osservare la nuova struttura realizzata.

Apache FOP, il processore che abbiamo utilizzato, è in grado di inviare l'output elaborato direttamente alla stampante. Inoltre Apache FOP è un software, seppure "ancora giovane", che nasce non per essere utilizzato principalmente da linea di comando, ma per essere integrato in altre applicazioni (java, applicazioni lato server). Riesce quindi facile immaginare che volendo non sarebbe eccessivamente difficoltoso ed oneroso approntare un sistema di "stampa su richiesta" per i testi TEI/XML di un ipotetico archivio digitale. Vorremmo aggiungere che altrettanto semplice sarebbe, sulla base di quanto già visto nel capitolo sull'accessibilità, consentire agli utenti una personalizzazione, sulla base di proprie specifiche esigenze, delle caratteristiche tipografiche degli output di stampa (dimensioni e tipo di carattere, interlinea etc.); infatti, analogamente a quanto si è visto per il rendering a video anche gli output a stampa vengono generati dinamicamente a partire dai dati XML sulla base delle indicazioni fornite dall'utente ai software di elaborazione.

Tornando al nostro foglio di stile vediamo che l'elemento `<fo:simple-page-master>`, cui abbiamo dato nome di "principale" (`master-name="principale"`), definisce le dimensioni dei margini della pagina; all'interno dell'area delimitata da questi margini si trova la `<fo:region-body>`; anche per questa regione abbiamo definito dei margini, un margine superiore ed uno inferiore entrambi di 1.8cm. Si noti che la pagina mastro in questione definisce anche altre due regioni `<fo:region-before>` `<fo:region-after>`, praticamente intestazione e piè di pagina, si noti che l'estensione di queste due regioni è inferiore a quella dei margini di `<fo:region-body>`; infatti come sappiamo tutte le regioni diverse dal corpo del documento (`<fo:region-body>`) si sovrappongono a questo, se avessimo impostato una dimensione per queste aree superiore a quella dei margini di `<fo:region-body>`, avremmo rischiato che i contenuti di queste si potessero venire a sovrapporre con i contenuti del corpo del documento, ossia della `<fo:region-body>`.

Proseguendo nell'analisi del foglio di stile incontriamo il template del <*teiHeader*>; come vediamo non ha alcun contenuto, la sua funzione infatti è quella di escludere dall'elaborazione questo nodo con tutti i suoi figli. Di seguito il modello dell'elemento <*front*> dapprima, mediante il comando `<xsl:apply-templates/>`, invoca i modelli per gli elementi figli, che in questa circostanza si occupano della creazione della copertina/frontespizio del documento, e successivamente inserisce nell'output una serie di pagine contenenti le dediche e le epigrafi; coerentemente con quanto fatto per le altre elaborazioni anche in questa occasione questi elementi sono stati inseriti manualmente, probabilmente nel caso ci fossimo trovati a lavorare non su di un solo testo bensì su di un corpus di testi, avremmo cercato o di definire una pratica univoca per la codifica ed il trattamento di dediche ed epigrafi ovvero, sulla scorta peraltro di quanto fatto in progetti analoghi, avremmo optato per l'esclusione pressoché generalizzata dalla codifica di queste componenti del testo.

Uno dei modelli principali del nostro codice è, come è ovvio, quello dei nodi <*div1*>, elemento con cui nel sorgente XML, come il lettore dovrebbe ricordare, abbiamo marcato i singoli capitoli. Questo template crea una sequenza di pagine `<fo:page-sequence>` in cui racchiude il contenuto del capitolo. Nell'elemento `<fo:page-sequence>` mediante due elementi `<fo:static-content>`[220] vengono definiti una intestazione ed un piè di pagina fissi per tutte le pagine della sequenza. Come intestazione (`xsl-region-before`) di tutte le pagine del testo abbiamo scelto: "MATTEO COLLURA - BALTICO: edizione elettronica"[221]. Nel piè di pagina (`xsl-region-after`) abbiamo usato il comando `<fo:page-number/>` per produrre il numero di pagina corrente[222]. I contenuti testuali del capitolo vengono inseriti, come detto, all'interno del marcatore `<fo:flow>`, dove per mezzo del tag

[220] Questo Formatting Object contiene altri oggetti di formattazione che devono essere inseriti e ripetuti in tutte le pagine della sequenza; il suo uso più comune è per la ripetizione di intestazioni e piè di pagina.
[221] Nel foglio di stile avremmo anche potuto fare in modo di produrre dinamicamente il valore dell'intestazione, tuttavia in questa circostanza abbiamo preferito inserirla noi per evitare di complicare ulteriormente la leggibilità del, già non semplice, codice.
[222] Si noti che il numero della pagina corrente viene creato dinamicamente sulla base delle pagine che vengono via via generate nel corso della composizione dell'output.

XSLT `<xsl:apply-templates/>` vengono invocati i template per gli elementi figli di <*div1*>, ossia i singoli paragrafi, i titoli di capitolo etc..

Tutti i contenuti dei capitoli, paragrafi, titoli e così via, sono stati racchiusi dentro elementi `<fo:block>` per mezzo dei rispettivi template. Si noti che nei modelli dei titoli di capitolo (<*head*>) di tipo ordinale e convenzionale si è fatto in modo che nell'elemento `<fo:block>`, che li deve marcare nell'output FO, fosse inserito un attributo *id* il cui valore è generato sulla base dell'attributo *id* dell'elemento <*div1*> genitore dell'<*head*> nodo contesto, sulla scorta di quanto già fatto nelle elaborazioni precedentemente analizzate; ciò anche in questo caso è servito per la creazione di un indice del documento. Se il lettore vorrà tornare per un momento ad esaminare il template del nodo radice noterà la presenza di due comandi `<xsl:apply-templates/>`, il secondo dei quali caratterizzato dall'attributo mode con valore *index*[223]. I modelli della modalità "index", i quali vengono attivati subito dopo la prima elaborazione dell'istanza XML sorgente, provvedono a creare un indice generale del documento. Anche in questa circostanza, come già visto nell'occasione della produzione dell'output HTML, tutto si basa su un sistema di ancore e riferimenti incrociati, in particolare il comando `<fo:page-number-citation ref-id="{../@id}"/>` restituisce il numero della pagina in cui occorre il Formatting Object il cui valore dell'attributo *id* corrisponde al valore del proprio attributo *ref-id*, e quindi nello specifico dà il valore della pagina in cui occorre il corrispettivo titolo di capitolo e perciò dell'inizio capitolo stesso. Mentre l'indice della versione HTML era basato su di un insieme di collegamenti ipertestuali, questo che stiamo illustrando è un indice "tradizionale" caratterizzato da riferimenti a numeri di pagina, ossia a pagine effettivamente esistenti. In appendice riportiamo alcune pagine stampate a partire dall'output PDF prodotto grazie all'elaborazione XSLT/XSL-FO del codice XML di Baltico, che mostrano il risultato finale di tutto questo processo appena illustrato.

[223] Abbiamo nei capitoli precedenti discusso dell'uso delle modalità nell'elaborazione dei sorgenti XML con XSLT evitiamo perciò di ripetere in questa occasione quanto già detto.

Come detto all'inizio del capitolo non si è considerata prioritaria la creazione di una versione e-book PDF di Baltico; ciò nonostante non abbiamo totalmente escluso questa ipotesi, abbiamo pertanto cercato di ottenere questo risultato con gli strumenti a nostra disposizione. Abbiamo in più circostanze detto che il PDF è un formato orientato alla descrizione di pagina ed alla stampa, avendo quindi limitate caratteristiche di adattamento dinamico al device di visualizzazione come accade per il formato LIT o lo stesso HTML, occorre nella produzione di un e-book in questo formato adottare delle dimensioni della pagina che si adattino al meglio allo schermo del dispositivo su cui il file verrà aperto; l'optimum è quindi la realizzazione di output di lettura appositamente confezionati per specifiche categorie reader (Tablet PC, palmari, computer etc.), ove non si voglia o non si possa far ciò è prassi comune adottare per gli e-book PDF una dimensione di pagina di circa nove pollici di altezza per sei pollici di larghezza, misure che costituiscono un buon compromesso al fine di garantire una visualizzazione confortevole su di una vasta gamma di dispositivi ed allo stesso tempo consentirne la stampa senza difficoltà[224]. Tra le altre caratteristiche che contraddistinguono un file PDF destinato ad una fruizione a video vi sono la presenza dei segnalibri (bookmark), una caratteristica di navigazione dei file PDF, ed i cosiddetti tag PDF che dovrebbero garantire in una certa misura la possibilità di adattamento dinamico del documento PDF al device di lettura. Originariamente gli e-book PDF erano destinati ad essere fruiti grazie ad uno specifico lettore, l'Acrobat eBook Reader (ex Glassbook Reader); oggi questo programma è stato abbandonato dalla Adobe e le funzioni di e-book reader sono state integrate nel visualizzatore standard di file PDF , l'Acrobat reader che per l'occasione è stato ribattezzato Adobe Reader. A nostro modesto parere con questa nuova versione del prodotto della software house americana si è avuta se non proprio una perdita di

[224] Cfr. ADOBE SYSTEMS INCORPORATED, *How to Create Adobe PDF Files for eBooks*, *e-book* disponibile on-line sul sito *Adobe*
<http://www.adobe.com/products/acrdis/createbooks.html>
07 Ottobre 2000, (19 Maggio 2004).

funzionalità quanto meno una perdita di praticità del programma nell'utilizzo degli e-book; è oggi meno netta la distinzione fra un normale documento PDF e un e-book PDF. Dal punto di vista della protezione dei contenuti la Adobe offre una delle soluzioni più efficienti e versatili con il proprio Adobe Content Server, questa piattaforma per il DRM e la distribuzione di contenuti, come abbiamo già visto, ha una delle sue caratteristiche di spicco nel consentire il prestito dei libri elettronici, cosa impossibile al momento con altri sistemi di DRM, d'altra parte i suoi alti costi di licenza non la rendono appetibile per realtà accademiche o no profit. I file PDF prevedono dei meccanismi di protezione[225] standard contro la modifica non autorizzata dei documenti, tuttavia tale protezione può venire facilmente eliminata con alcuni software, tra i quali quello di una nota azienda russa che è possibile acquistare senza difficoltà dal sito internet della stessa.[226]

Nella realizzazione di una versione e-book PDF del testo di Baltico abbiamo cercato di tener conto di tutti questi elementi pur servendoci degli strumenti precedentemente descritti. La nostra attività ha preso l'avvio dal lavoro già svolto per il primo output PDF del romanzo di Collura, quello per la stampa, integrando e rielaborando i fogli di stile già creati in quel progetto per la produzione di un file che fosse dotato quanto più possibile di tutte quelle caratteristiche che contraddistinguono un e-book PDF secondo i canoni sopra descritti, pur usando le tecnologie XSLT/XSL-FO, Apache FOP e pochi altri strumenti gratuiti.

Per quanto riguarda le dimensioni pagina è bastato modificare le misure di altezza e larghezza in `<fo:simple-page-master>`, come si può notare dal codice del foglio di stile anch'esso riportato in allegato, si è definita un'altezza di 23cm ed un larghezza di 15cm, corrispondenti all'incirca alla dimensione di 9x6 pollici detta

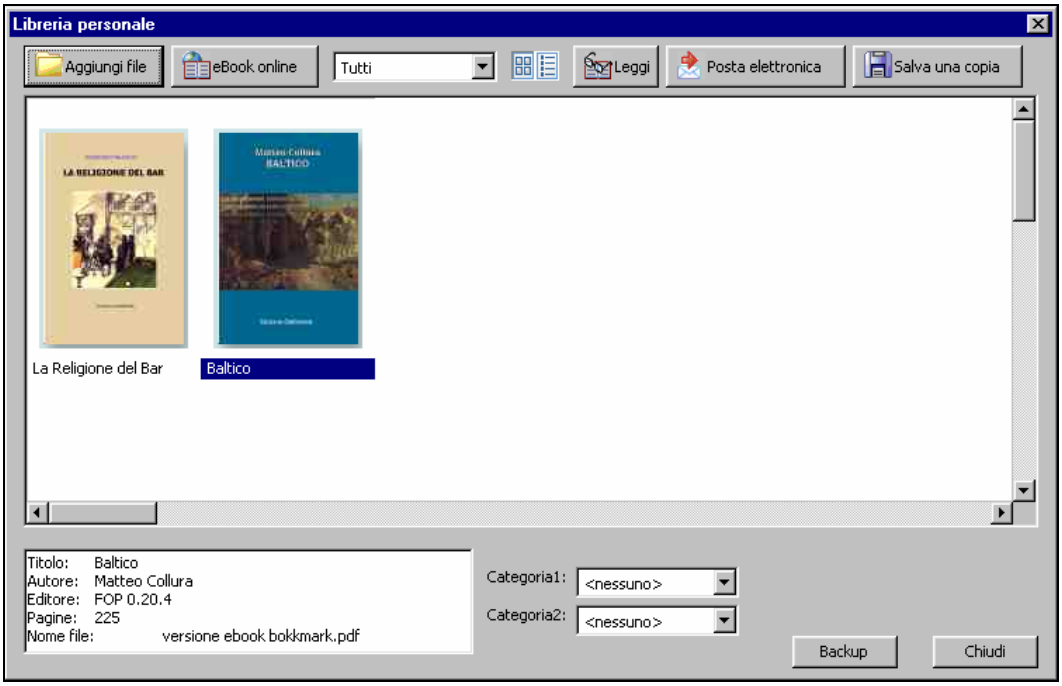
[225] Ci riferiamo in particolare ai meccanismi di protezione standard di Acrobat basati su sistemi crittografici a 40 bit e 128 bit. Occorre precisare che il software Adobe permette anche la firma digitale dei documenti e l'implementazione di sistemi di protezione di terze parti.

[226] Meno semplice è invece la sprotezione di files PDF in cui sia stata impostata una password di apertura, in tal caso il software in questione è obbligato ad adottare una attacco di tipo brute force (un metodo per decriptare password o chiavi di registrazione tentando, in sequenza, tutte le possibili combinazioni di caratteri utilizzabili) per la rimozione delle protezioni al file, è ovvio che nel caso la password sia conosciuta questa necessità non sussiste. Occorre dire tuttavia che in questi ultimi mesi si sta assistendo al nascere di numerose piccole realtà che, non volendo sostenere gli alti costi dell'Adobe Content Server, hanno deciso di adottare questa elementare forma di protezione per la commercializzazione di contenuti in PDF.

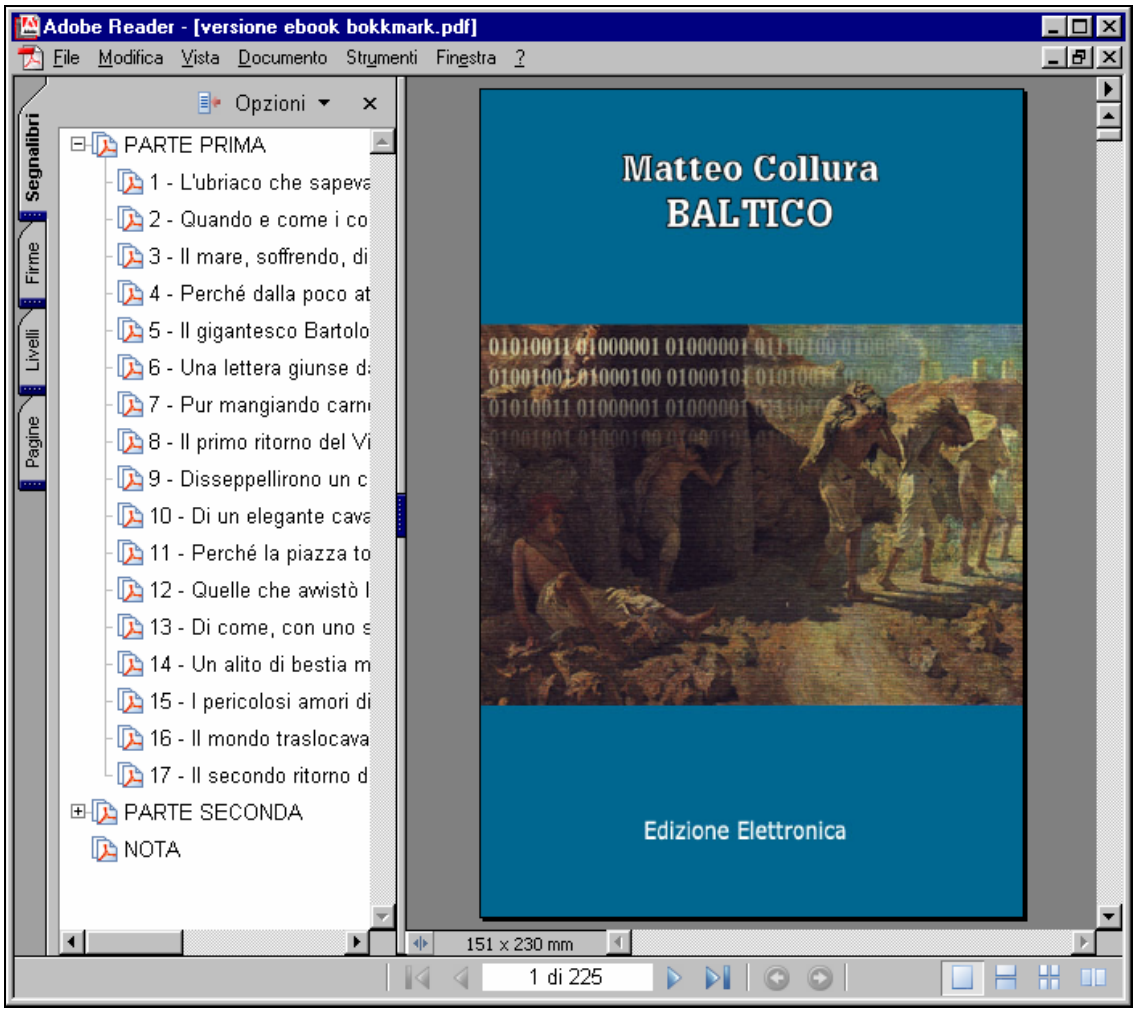
poco prima. Si sarà notato che in questo nuovo foglio XSLT nel `<fo:layout-master-set>` sono stati inseriti due `<fo:simple-page-master>`, la prima pagina mastro, identificata dal nome "cover" serve da modello di pagina per la copertina, la quale abbiamo riciclato dal progetto per la versione e-book LIT del romanzo, e che, seppure non indispensabile, costituisce un vezzo estetico che abbiamo voluto concedere al nostro lavoro. Più difficoltosa è stata l'inserzione nel nostro file dei segnalibro di navigazione (bookmark), allo stadio attuale dello sviluppo, XSL-FO non supporta i bookmark PDF[227], per fortuna Apache FOP, il processore FO da noi adottato, è dotato di un'estensione[228], che consente l'elaborazione dei segnalibri. Identificati dal namespace "fox" i comandi per la definizione dei bookmark, situati nel template della radice immediatamente dopo l'elemento `<fo:layout-master-set>`, sono `<fox:outline>` e `<fox:label>`, il primo si occupa di stabilire la destinazione del segnalibro, il secondo invece fornisce il nome del bookmark che apparirà nella barra dei segnalibri di Adobe Acrobat. Il sistema per la definizione dei segnalibri in XSLT, che ricalca da vicino quello già adoperato per la creazione della Table of Contents della versione OEBPS del romanzo, si basa su di una serie di elaborazioni `<xsl:for-each>` su tutti i nodi <*div0*>, <*div*> e su i nodi <*head*>, in cui l'attributo *type* abbia valore *'tem'*, la struttura di ancore e riferimenti incrociati si basa sul consueto meccanismo fondato sull'utilizzo degli attributi *id* e *n* degli elementi <*div1*> e <*div0*> già più volte visto in precedenti fogli di stile, si noti soltanto l'utilizzo della funzione translate() nella definizione dell'etichette dei titoli dei capitoli per rimuovere i ritorno a capo, che in fase di codifica nel sorgente abbiamo mantenuto così come occorrevano nella fonte cartacea. Per il resto il nostro foglio di stile si rifà in tutto e per tutto a quello approntato per la versione "stampabile", con l'unica differenza della minore dimensione delle pagine e dell'assenza dell'indice finale. Di seguito riportiamo

[227] Questa funzionalità sembra verrà inserita in una nuova versione delle specifiche XSL.
[228] Le estensioni sono delle funzionalità ulteriori che vengono talvolta fornite dai produttori ai programmi di elaborazione XML (XSLT, XSL-FO) per semplificare alcuni dei compiti più complessi ovvero per svolgere quelle funzioni non coperte dalle specifiche del linguaggio, come nel nostro caso.

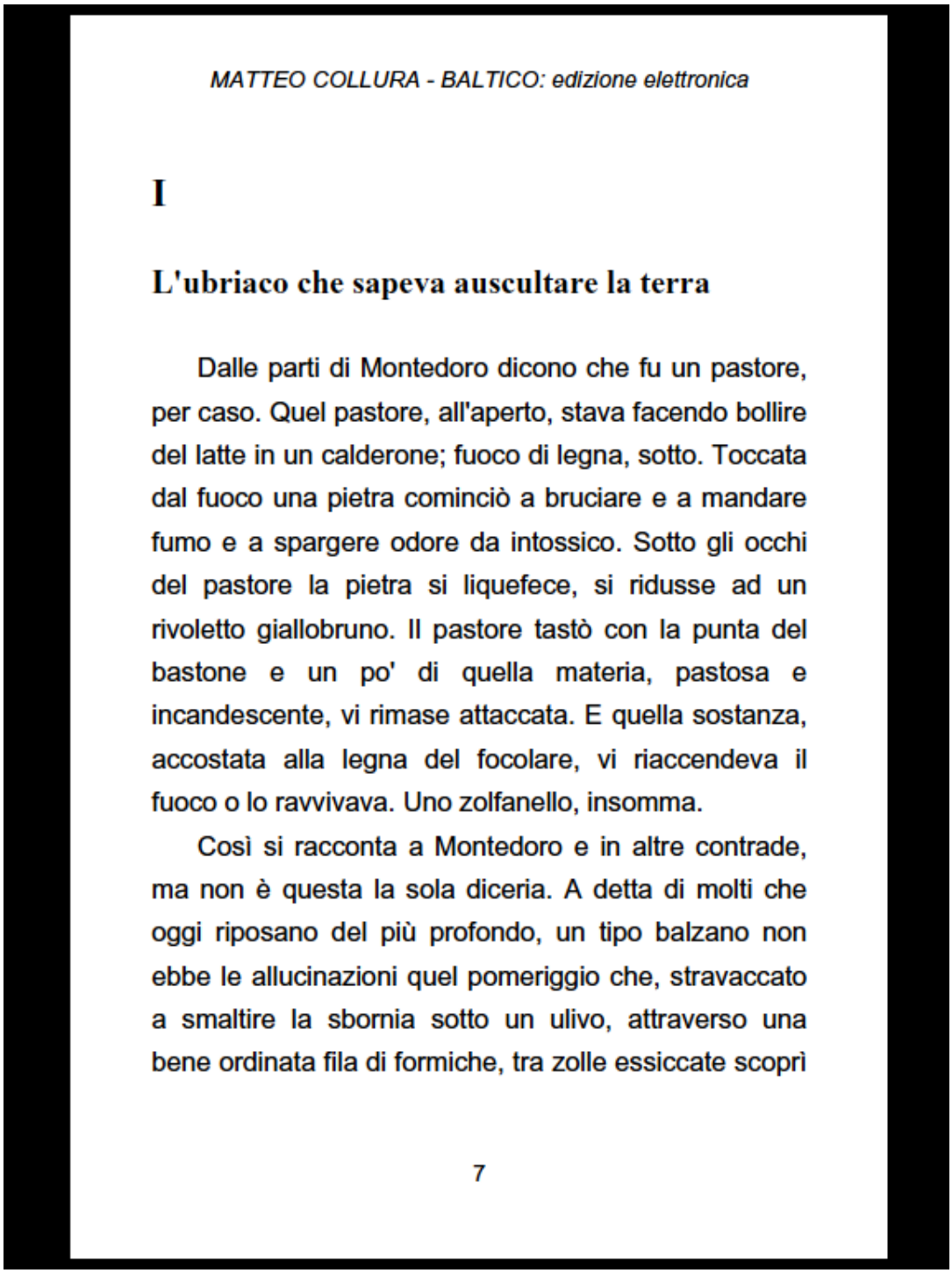
alcune schermate che mostrano come l'output PDF ottenuto si presenti nell'Adobe Reader 6.0:



L'area di Adobe Reader Libreria personale con la miniatura della copertina



Segnalibri e copertina



La prima pagina del testo visualizzata in modalità schermo intero

Una volta conclusa l'elaborazione automatica e prodotto l'output PDF abbiamo voluto eseguire su questo alcuni piccoli lavori ulteriori, riguardanti la protezione del documento ed i metadati dello stesso, coerentemente con quanto gia fatto in precedenza, anche in questa occasione ci siamo serviti di programmi freeware.

Ogni documento PDF può essere associato a dei metadati contenenti il titolo, l'autore, il soggetto della pubblicazione etc. che ne agevolano l'archiviazione e l'indicizzazione. Il processore FO da noi utilizzato non permette di inserire automaticamente queste informazioni nel file finale; pertanto, abbiamo deciso di giovarci di un piccolo programma freeware dal nome *PDF info,* prodotto dalla Bureausoft[229], che permette di inserire e modificare manualmente queste informazioni. Un'altra funzione avanzata con cui abbiamo ritenuto opportuno accompagnare il nostro file, sebbene fossimo consci della possibilità di aggirare questa protezione, è stata l'impostazione di alcune limitazioni di sicurezza; abbiamo, infatti, disabilitato la possibilità di modificare il documento, stabilendo la necessità di inserire una password per l'esecuzione di operazioni di modifica e cambiamento delle impostazioni di sicurezza: per far ciò ci siamo serviti del software open source pdftk[230] e della relativa interfaccia grafica guipdftk[231] realizzata da un giovane programmatore di nome Dirk Paehl.

Ci preme far notare che quanto mostrato in queste pagine è stato realizzato con programmi a costo zero; d'altra parte, buona parte dei software e per certi versi delle stesse tecnologie utilizzate sono ancora molto giovani e, quindi, verosimilmente soggette a futuri miglioramenti.

[229] <http://www.bureausoft.com>.
[230] <http://www.accesspdf.com/pdftk/>.
[231] <http://de.geocities.com/dpaehl2004/guipdftk/>.

Appendice 14

FOGLIO DI STILE XSLT PER LA TRASFORMAZIONE IN XSL-FO DI BALTICO

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <xsl:output method="xml" encoding="iso-8859-1" indent="yes"/>
  <xsl:strip-space elements="p"></xsl:strip-space>

<xsl:template match="/">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
 <fo:layout-master-set>
  <fo:simple-page-master master-name="principale" page-height="29.7cm"
page-width="21cm" margin-top="1cm" margin-bottom="1cm" margin-
left="2.5cm" margin-right="2.5cm">
       <fo:region-before extent="1.5cm" />
            <fo:region-after  extent="1.5cm" />
       <fo:region-body margin-top="1.8cm"  margin-bottom="1.8cm" />
     </fo:simple-page-master>
  </fo:layout-master-set>
  <xsl:apply-templates/> <xsl:apply-templates mode="index"/>
</fo:root>
</xsl:template>

<xsl:template match="teiHeader"></xsl:template>
<xsl:template match="text"><xsl:apply-templates/></xsl:template>
<!-- INIZIO FRONT -->
<xsl:template match="front"><xsl:apply-templates/>
<!-- INIZIO dediche inserite manualmente -->
  <fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-before="1cm" font-size="15pt"  text-align="end"
font-family="Times Roman" font-style="italic">
A Bartolomeo Collura, mio padre.
    </fo:block>
  </fo:flow>
  </fo:page-sequence>

  <fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-before="1cm" font-size="13pt"  text-
align="justify"  font-family="Times Roman" line-height="1.5em" margin-
right="0.5cm" margin-left="0.5cm">
Contagiati dal delirio delle escavazioni, subito accompagnato dalla
comparsa di affaristi scrocconi, si scoprirono impensate doti di
imprenditori; e sventrando valli e colline sognarono di arricchire,
mentre copioso colava lo zolfo e si ampliavano i cimiteri. Due secoli
di picconate cambiarono la faccia della terra. Subito si appales&#242;
il disastro, ma in quel turbinio di fortune immaginate nessuno vi fece
caso. Corsero ai ripari quando gi&#224; il vento screpolava gli spalti
delle zolfare e le erbacce cominciavano a nasconderne le bocche. Fu
come se un'ostinata bonaccia si fosse posata su un mare che era stato
in tempesta. Non lontano dai ruderi, oziosi, aspettarono sussidi e
pensioni; e polvere e silenzio sedimentarono sulla loro assurda
epopea.
    </fo:block>
  </fo:flow>
```

```
  </fo:page-sequence>

    <fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-before="1cm" space-after="0.4cm" font-size="13pt"
text-align="left"  font-family="Times Roman" line-height="1.2em"
margin-left="9.4cm">
Zufolava mentre andava al lavoro
e parlava spesso di un futuro di
benessere e di abbondanza
    </fo:block>
  <fo:block  font-size="13pt"  text-align="end"  font-family="Times
Roman" font-variant="small-caps">
Sherwood Anderson,
  </fo:block>
  <fo:block  font-size="13pt"  text-align="end"  font-family="Times
Roman" font-style="italic">
Un povero
bianco.         
  </fo:block>
  </fo:flow>
  </fo:page-sequence>
<!-- FINE dediche inserite manualmente -->
</xsl:template>
<!-- le dediche non sono state elaborate -->
<xsl:template match="div[@type='ded']" priority="1"></xsl:template>
<xsl:template match="div[@type='ep']" priority="1"></xsl:template>

<xsl:template match="titlePage">
  <fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <xsl:apply-templates/>
  </fo:flow>
  </fo:page-sequence>
</xsl:template>

<xsl:template match="docAuthor"><fo:block space-after="3.8cm" font-
size="23pt"  text-align="center"  font-weight="bold" font-
family="Times Roman"><xsl:apply-templates/></fo:block></xsl:template>

<xsl:template match="docTitle"><xsl:apply-templates/></xsl:template>

<xsl:template match="titlePart[@type='main']"><fo:block space-
after="10pt" font-size="30pt"  text-align="center"  font-weight="bold"
font-family="Times Roman"><xsl:apply-
templates/></fo:block></xsl:template>

<xsl:template match="titlePart[@type='sub']"><fo:block space-
after="14.7cm" font-size="18pt"  text-align="center"  font-
weight="bold" font-family="Times Roman"><xsl:apply-
templates/></fo:block></xsl:template>

<xsl:template match="docImprint"><fo:block  font-size="18pt"  text-
align="center"  font-weight="bold" font-family="Times
Roman"><xsl:apply-templates/></fo:block></xsl:template>

<!-- FINE FRONT -->

<xsl:template match="body"><xsl:apply-templates/></xsl:template>

<xsl:template match="div0">
```

```
  <fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-after="15pt" font-size="20pt"  text-align="center"
font-weight="bold" font-family="Times Roman">
<xsl:value-of select="p"></xsl:value-of>
    </fo:block>
  </fo:flow>
  </fo:page-sequence>
  <xsl:apply-templates/>
</xsl:template>

<xsl:template match="div0/p"></xsl:template>

<xsl:template match="div1">
<fo:page-sequence master-reference="principale">
  <fo:static-content flow-name="xsl-region-before">
           <fo:block text-align="center" font-size="12pt" font-
family="sans-serif" font-weight="normal" font-style="italic"> MATTEO
COLLURA - BALTICO: edizione elettronica</fo:block>
        </fo:static-content>
        <fo:static-content flow-name="xsl-region-after" margin-
top="0.5cm">
           <fo:block  space-before="0.7cm" text-align="center" font-
size="12pt" font-family="sans-serif" font-weight="normal">
         <fo:page-number/>
           </fo:block>
        </fo:static-content>
  <fo:flow flow-name="xsl-region-body" text-align="justify" font-
size="14pt" font-family="Verdana, Geneva, Arial, Helvetica, sans-
serif" font-weight="normal"  line-height="1.7em">
  <xsl:apply-templates/>
  </fo:flow>
  </fo:page-sequence>
</xsl:template>

<xsl:template match="head[@type='ord']">
<fo:block space-after="17pt" font-size="20pt"  text-align="left"
font-weight="bold" font-family="Times Roman" id="{../@id}"><xsl:apply-
templates/></fo:block>
</xsl:template>

<xsl:template match="head[@type='conv']">
<fo:block space-after="17pt" font-size="20pt"  text-align="left"
font-weight="bold" font-family="Times Roman" id="{../@id}"><xsl:apply-
templates/></fo:block>
</xsl:template>

<xsl:template match="head[@type='tem']">
       <fo:block space-after="20pt" font-size="17pt"  text-
align="left"  font-weight="bold" font-family="Times Roman"><xsl:apply-
templates/></fo:block>
</xsl:template>

<xsl:template match="p">
  <fo:block text-indent="24pt"><xsl:apply-templates/></fo:block>
</xsl:template>

<xsl:template match="back" >
<fo:page-sequence master-reference="principale">
```

```
<fo:flow flow-name="xsl-region-body" text-align="justify" font-
size="14pt" font-family="Verdana, Geneva, Arial, Helvetica, sans-
serif" font-weight="normal"  line-height="1.7em">
  <xsl:apply-templates/>
  </fo:flow>
  </fo:page-sequence>
</xsl:template>

<xsl:template match="div/signed">
  <fo:block text-align="end" font-size="14pt" font-family="Verdana,
Geneva, Arial, Helvetica, sans-serif" font-weight="normal"
><xsl:apply-templates/></fo:block> </xsl:template>

<xsl:template match="l"><fo:block><xsl:apply-
templates/></fo:block></xsl:template>


<xsl:template match="*[@rend='bloc']">
<fo:block  margin-left="3.5cm" margin-right="2.5cm"><xsl:apply-
templates/></fo:block>
</xsl:template>


<xsl:template match="*[@rend='italic']">
<fo:inline font-style="italic"><xsl:apply-templates/></fo:inline>
</xsl:template>

<xsl:template match="q[@type='epistola']">
  <fo:block text-indent="24pt"><xsl:apply-templates/></fo:block>
  </xsl:template>

<xsl:template match="q/text"><xsl:apply-templates/></xsl:template>

<xsl:template match="q/body"><xsl:apply-templates/></xsl:template>
<!-- inizio INDEX -->

<xsl:template match="text" mode="index"><xsl:apply-templates
mode="index"/></xsl:template>

<xsl:template match="front" mode="index"></xsl:template>

<xsl:template match="body" mode="index">
<fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-before="1cm" font-size="23pt"  text-align="center"
font-weight="bold" font-family="Times Roman">
INDICE
    </fo:block>
  </fo:flow>
  </fo:page-sequence><xsl:apply-templates
mode="index"/></xsl:template>

<xsl:template match="div0" mode="index">
<fo:page-sequence master-reference="principale">
  <fo:flow flow-name="xsl-region-body">
    <fo:block space-before="2.5cm" space-after="0.8cm" font-
size="13pt"  text-align="center"  font-family="Times Roman" font-
style="italic" text-transform="uppercase">
<xsl:value-of select="p"/>
    </fo:block>
    <fo:table   table-layout="fixed">
```

```
                    <fo:table-column column-width="16cm"/>
                    <fo:table-column column-width="1cm"/>
                    <fo:table-body>
  <xsl:apply-templates mode="index"/>
              </fo:table-body>
                    </fo:table>

  </fo:flow>
  </fo:page-sequence></xsl:template>

<xsl:template match="div0/p" mode="index"></xsl:template>

<xsl:template match="div1" mode="index">
<xsl:apply-templates mode="index"/>
</xsl:template>

<xsl:template match="head[@type='ord']" mode="index">
  <fo:table-row>
  <fo:table-cell>
  <fo:block  font-size="15pt" font-family="Times Roman">
<xsl:value-of select="."/><xsl:apply-templates mode="index"/>
  </fo:block>
  </fo:table-cell>
  </fo:table-row>
</xsl:template>

<xsl:template match="head[@type='conv']" mode="index">
  <fo:table-row>
  <fo:table-cell>
  <fo:block  font-size="15pt" font-family="Times Roman">
   
  </fo:block>
  </fo:table-cell>
  </fo:table-row>
  <fo:table-row>
  <fo:table-cell>
  <fo:block  font-size="14pt" font-family="Times Roman">
<xsl:value-of select="."/>    <fo:leader leader-
pattern="dots" rule-style="solid" rule-thickness="1mm"/>
  </fo:block>
  </fo:table-cell>
  <fo:table-cell>
  <fo:block  font-size="14pt" font-family="Times Roman">
  <fo:page-number-citation ref-id="{../@id}"/><xsl:apply-
templates mode="index"/>
  </fo:block>
  </fo:table-cell>
  </fo:table-row>
</xsl:template>

<xsl:template match="head[@type='tem']" mode="index">
  <fo:table-row>
  <fo:table-cell>
  <fo:block  font-size="14pt" font-family="Times Roman">
<xsl:value-of select="."/>    <fo:leader leader-
pattern="dots" rule-style="solid" rule-thickness="1mm"/>
  </fo:block>
  </fo:table-cell>
  <fo:table-cell>
  <fo:block  font-size="14pt" font-family="Times Roman">
```

```
  <fo:page-number-citation ref-id="{../@id}"/><xsl:apply-
templates mode="index"/>
  </fo:block>
  </fo:table-cell>
  </fo:table-row>
</xsl:template>

<xsl:template match="q/text" mode="index"></xsl:template>

<xsl:template match="q/body" mode="index"></xsl:template>

<xsl:template match="back" mode="index"></xsl:template>

<xsl:template match="text()|@*" mode="index"></xsl:template>

</xsl:stylesheet>
```

Appendice 15

CODICE DELL'OUTPUT XSL-FO CONTENENTE IL FRONTESPIZIO IL PRIMO CAPITOLO E L'INDICE FINALE DI BALTICO

```
<?xml version="1.0" encoding="iso-8859-1"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="principale" page-
height="29.7cm" page-width="21cm" margin-top="1cm" margin-bottom="1cm"
margin-left="2.5cm" margin-right="2.5cm">
         <fo:region-before extent="1.5cm"/>
         <fo:region-after extent="1.5cm"/>
         <fo:region-body margin-top="1.8cm" margin-bottom="1.8cm"/>
      </fo:simple-page-master>
   </fo:layout-master-set>

   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">

         <fo:block space-after="3.8cm" font-size="23pt" text-
align="center" font-weight="bold" font-family="Times Roman">
 MATTEO COLLURA
 </fo:block>


         <fo:block space-after="10pt" font-size="30pt" text-
align="center" font-weight="bold" font-family="Times
Roman">BALTICO</fo:block>

         <fo:block space-after="14.7cm" font-size="18pt" text-
align="center" font-weight="bold" font-family="Times Roman">Un'epopea
siciliana</fo:block>


         <fo:block font-size="18pt" text-align="center" font-
weight="bold" font-family="Times Roman">
 BIBLIOTECA - REVERDITO EDITORE
 </fo:block>

      </fo:flow>
   </fo:page-sequence>




   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
         <fo:block space-before="1cm" font-size="15pt" text-
align="end" font-family="Times Roman" font-style="italic">
A Bartolomeo Collura, mio padre.
     </fo:block>
      </fo:flow>
   </fo:page-sequence>
   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
```

```
         <fo:block space-before="1cm" font-size="13pt" text-
align="justify" font-family="Times Roman" line-height="1.5em" margin-
right="0.5cm" margin-left="0.5cm">
Contagiati dal delirio delle escavazioni, subito accompagnato dalla
comparsa di affaristi scrocconi, si scoprirono impensate doti di
imprenditori; e sventrando valli e colline sognarono di arricchire,
mentre copioso colava lo zolfo e si ampliavano i cimiteri. Due secoli
di picconate cambiarono la faccia della terra. Subito si appalesò il
disastro, ma in quel turbinio di fortune immaginate nessuno vi fece
caso. Corsero ai ripari quando già il vento screpolava gli spalti
delle zolfare e le erbacce cominciavano a nasconderne le bocche. Fu
come se un'ostinata bonaccia si fosse posata su un mare che era stato
in tempesta. Non lontano dai ruderi, oziosi, aspettarono sussidi e
pensioni; e polvere e silenzio sedimentarono sulla loro assurda
epopea.
     </fo:block>
      </fo:flow>
   </fo:page-sequence>
   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
         <fo:block space-before="1cm" space-after="0.4cm" font-
size="13pt" text-align="left" font-family="Times Roman" line-
height="1.2em" margin-left="9.4cm">
Zufolava mentre andava al lavoro
e parlava spesso di un futuro di
benessere e di abbondanza
     </fo:block>
         <fo:block font-size="13pt" text-align="end" font-
family="Times Roman" font-variant="small-caps">
Sherwood Anderson,
  </fo:block>
         <fo:block font-size="13pt" text-align="end" font-
family="Times Roman" font-style="italic">
Un povero bianco.
  </fo:block>
      </fo:flow>
   </fo:page-sequence>


   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
         <fo:block space-after="15pt" font-size="20pt" text-
align="center" font-weight="bold" font-family="Times Roman">PARTE
PRIMA</fo:block>
      </fo:flow>
   </fo:page-sequence>



   <fo:page-sequence master-reference="principale">
      <fo:static-content flow-name="xsl-region-before">
         <fo:block text-align="center" font-size="12pt" font-
family="sans-serif" font-weight="normal" font-style="italic"> MATTEO
COLLURA - BALTICO: edizione elettronica</fo:block>
      </fo:static-content>
      <fo:static-content flow-name="xsl-region-after" margin-
top="0.5cm">
         <fo:block space-before="0.7cm" text-align="center" font-
size="12pt" font-family="sans-serif" font-weight="normal">
            <fo:page-number/>
         </fo:block>
```

</fo:static-content>
<fo:flow flow-name="xsl-region-body" text-align="justify" font-size="14pt" font-family="Verdana, Geneva, Arial, Helvetica, sans-serif" font-weight="normal" line-height="1.7em">

<fo:block space-after="17pt" font-size="20pt" text-align="left" font-weight="bold" font-family="Times Roman" id="I.1">I</fo:block>

<fo:block space-after="20pt" font-size="17pt" text-align="left" font-weight="bold" font-family="Times Roman">L'ubriaco che sapeva auscultare la terra</fo:block>

<fo:block text-indent="24pt">
Dalle parti di Montedoro dicono che fu un pastore, per caso. Quel pastore, all'aperto, stava facendo bollire del latte in un calderone; fuoco di legna, sotto. Toccata dal fuoco una pietra cominciò a bruciare e a mandare fumo e a spargere odore da intossico. Sotto gli occhi del pastore la pietra si liquefece, si ridusse ad un rivoletto giallobruno. Il pastore tastò con la punta del bastone e un po' di quella materia, pastosa e incandescente, vi rimase attaccata. E quella sostanza, accostata alla legna del focolare, vi riaccendeva il fuoco o lo ravvivava. Uno zolfanello, insomma.
</fo:block>

<fo:block text-indent="24pt">
Così si racconta a Montedoro e in altre contrade, ma non è questa la sola diceria. A detta di molti che oggi riposano del più profondo, un tipo balzano non ebbe le allucinazioni quel pomeriggio che, stravaccato a smaltire la sbornia sotto un ulivo, attraverso una bene ordinata fila di formiche, tra zolle essiccate scoprì il minuscolo ingresso dal quale si raggiungeva uno sterminato tesoro. Una formica, non più grossa delle altre, attirò la sua attenzione emersa per qualche secondo dal molle pantano in cui affondava. La formica trascinava una pietruzza gialla, di un giallo brillante, oro si sarebbe detto. E altre formiche trascinavano pietruzze gialle e tutte affioravano da un buco seminascosto nel disordine polveroso di quel palmo di terra. Più ispirato che incuriosito, l'ubriaco affondò un dito nella terra e lo scosse provocandone frane interne. Brulicarono le formiche, allargandosi a stella, mentre il dito si accaniva a cagionare cataclismi in quel piccolo mondo. Quando la minuscola caverna fu liberata delle formiche e del terriccio, apparve l'indizio giallo e luccicante di una vena di zolfo. Il cielo incendiato dal tramonto non poté competere con quel brillante incastonato nelle radici dell'ulivo. Abbagliava quell'oro che, s'intuiva, da quelle parti doveva abbondare.
</fo:block>

<fo:block text-indent="24pt">
L'uomo che nonostante i deliri dell'alcool o che, forse, grazie ad essi per primo aveva scoperto la ricchezza che il suolo malcelava, diventò un oracolo vivente. Barcollante e cantilenante discorsi incomprensibili, fu trascinato di qua e di là per frugare col suo occhio smorto, ma pur sempre ritenuto infallibile, la crosta terrestre. Con sollievo s'inginocchiava sui terreni prescelti dopo marce estenuanti, e come fosse un rabdomante in cerca di acqua, li grattava con le dita callose, li auscultava persino, tendendo l'orecchio come a voler strappare da un sospiro o da qualche altro arcano rumore del sottosuolo il segreto che gli serviva.
</fo:block>

```
         <fo:block text-indent="24pt">«Qui puoi darci sotto»,
incoraggiava alla fine. Oppure frustrava speranze cresciute troppo in
fretta: «Qui non ci caveresti una briciola di zolfo neanche ad
ammazzarti cent'anni».
</fo:block>

         <fo:block text-indent="24pt">
Non sempre ci azzeccava, ma la sua fama finì col non tollerare i
dubbi. Per anni, periti e geologi imbottiti di scienza avrebbero
dovuto fare i conti con l'infallibilità che la credenza popolare
attribuiva a quell'uomo bizzarro. Per anni, zolfatari praticoni,
saggiando collinette gessose o inseguendo i nauseabondi rigagnoli di
acque sulfuree, sul punto di decidere se buttarsi nell'impresa delle
trivellazioni sarebbero stati tentati di chiedere consiglio ad un
ubriacone ruttante.
 </fo:block>

      </fo:flow>
   </fo:page-sequence>

[. . .]

  <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
         <fo:block space-before="1cm" font-size="23pt" text-
align="center" font-weight="bold" font-family="Times Roman">
INDICE
     </fo:block>
      </fo:flow>
   </fo:page-sequence>
   <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
         <fo:block space-before="2.5cm" space-after="0.8cm" font-
size="13pt" text-align="center" font-family="Times Roman" font-
style="italic" text-transform="uppercase">PARTE PRIMA</fo:block>
         <fo:table table-layout="fixed">
            <fo:table-column column-width="16cm"/>
            <fo:table-column column-width="1cm"/>
            <fo:table-body>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="15pt" font-family="Times
Roman">I</fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">L'ubriaco che sapeva auscultare la terra    <fo:leader leader-
pattern="dots" rule-style="solid" rule-thickness="1mm"/>
                     </fo:block>
                  </fo:table-cell>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">
  <fo:page-number-citation ref-id="I.1"/>
                     </fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
```

```
                     <fo:block font-size="15pt" font-family="Times
Roman">II</fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">Quando e come i contadini
incubarono il malanno    <fo:leader leader-pattern="dots" rule-
style="solid" rule-thickness="1mm"/>
                     </fo:block>
                  </fo:table-cell>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">
  <fo:page-number-citation ref-id="I.2"/>
                     </fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="15pt" font-family="Times
Roman">III</fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">Il mare, soffrendo,
diede alla luce un'isoletta    <fo:leader leader-pattern="dots" rule-
style="solid" rule-thickness="1mm"/>
                     </fo:block>
                  </fo:table-cell>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">
  <fo:page-number-citation ref-id="I.3"/>
                     </fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
[…]
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="15pt" font-family="Times
Roman">XXIII</fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">Quel giorno il paese vide
i battaglioni del benessere    <fo:leader leader-pattern="dots" rule-
style="solid" rule-thickness="1mm"/>
                     </fo:block>
                  </fo:table-cell>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">
  <fo:page-number-citation ref-id="II.23"/>
```

```
                     </fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="15pt" font-family="Times
Roman">

  </fo:block>
                  </fo:table-cell>
               </fo:table-row>
               <fo:table-row>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">CONCLUSIONE    <fo:leader leader-pattern="dots" rule-
style="solid" rule-thickness="1mm"/>
                     </fo:block>
                  </fo:table-cell>
                  <fo:table-cell>
                     <fo:block font-size="14pt" font-family="Times
Roman">
  <fo:page-number-citation ref-id="II.24-conclusione"/>
                     </fo:block>
                  </fo:table-cell>
               </fo:table-row>
            </fo:table-body>
         </fo:table>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

Appendice 16

STAMPA DEL FRONTESPIZIO, DEL PRIMO CAPITOLO, DELLA NOTA FINALE, DELL'INDICE DI *BALTICO* DALL'OUTPUT PDF OTTENUTO MEDIANTE ELABORAZIONE XSLT/XSL-FO DEL CODICE XML/TEI

MATTEO COLLURA

BALTICO

Un'epopea siciliana

BIBLIOTECA - REVERDITO EDITORE

A Bartolomeo Collura, mio padre.

Contagiati dal delirio delle escavazioni, subito accompagnato dalla comparsa di affaristi scrocconi, si scoprirono impensate doti di imprenditori; e sventrando valli e colline sognarono di arricchire, mentre copioso colava lo zolfo e si ampliavano i cimiteri. Due secoli di picconate cambiarono la faccia della terra. Subito si appalesò il disastro, ma in quel turbinio di fortune immaginate nessuno vi fece caso. Corsero ai ripari quando già il vento screpolava gli spalti delle zolfare e le erbacce cominciavano a nasconderne le bocche. Fu come se un'ostinata bonaccia si fosse posata su un mare che era stato in tempesta. Non lontano dai ruderi, oziosi, aspettarono sussidi e pensioni; e polvere e silenzio sedimentarono sulla loro assurda epopea.

Zufolava mentre andava al lavoro e
parlava spesso di un futuro di
benessere e di abbondanza

SHERWOOD ANDERSON,
Un povero bianco.

PARTE PRIMA

I

L'ubriaco che sapeva auscultare la terra

Dalle parti di Montedoro dicono che fu un pastore, per caso. Quel pastore, all'aperto, stava facendo bollire del latte in un calderone; fuoco di legna, sotto. Toccata dal fuoco una pietra cominciò a bruciare e a mandare fumo e a spargere odore da intossico. Sotto gli occhi del pastore la pietra si liquefece, si ridusse ad un rivoletto giallobruno. Il pastore tastò con la punta del bastone e un po' di quella materia, pastosa e incandescente, vi rimase attaccata. E quella sostanza, accostata alla legna del focolare, vi riaccendeva il fuoco o lo ravvivava. Uno zolfanello, insomma.

Così si racconta a Montedoro e in altre contrade, ma non è questa la sola diceria. A detta di molti che oggi riposano del più profondo, un tipo balzano non ebbe le allucinazioni quel pomeriggio che, stravaccato a smaltire la sbornia sotto un ulivo, attraverso una bene ordinata fila di formiche, tra zolle essiccate scoprì il minuscolo ingresso dal quale si raggiungeva uno sterminato tesoro. Una formica, non più grossa delle altre, attirò la sua attenzione emersa per qualche secondo dal molle pantano in cui affondava. La formica trascinava una pietruzza gialla, di un giallo brillante, oro si sarebbe detto. E altre formiche trascinavano pietruzze gialle e tutte affioravano da un buco seminascosto nel disordine polveroso di quel palmo di terra. Più ispirato che incuriosito, l'ubriaco affondò un dito nella terra e lo scosse provocandone frane interne. Brulicarono le formiche, allargandosi a stella, mentre il dito si accaniva a cagionare cataclismi in quel piccolo mondo. Quando la minuscola caverna fu liberata delle formiche e del terriccio, apparve l'indizio giallo e luccicante di una vena di zolfo. Il cielo incendiato dal

tramonto non poté competere con quel brillante incastonato nelle radici dell'ulivo. Abbagliava quell'oro che, s'intuiva, da quelle parti doveva abbondare.

L'uomo che nonostante i deliri dell'alcool o che, forse, grazie ad essi per primo aveva scoperto la ricchezza che il suolo malcelava, diventò un oracolo vivente. Barcollante e cantilenante discorsi incomprensibili, fu trascinato di qua e di là per frugare col suo occhio smorto, ma pur sempre ritenuto infallibile, la crosta terrestre. Con sollievo s'inginocchiava sui terreni prescelti dopo marce estenuanti, e come fosse un rabdomante in cerca di acqua, li grattava con le dita callose, li auscultava persino, tendendo l'orecchio come a voler strappare da un sospiro o da qualche altro arcano rumore del sottosuolo il segreto che gli serviva.

«Qui puoi darci sotto», incoraggiava alla fine. Oppure frustrava speranze cresciute troppo in fretta: «Qui non ci caveresti una briciola di zolfo neanche ad ammazzarti cent'anni».

Non sempre ci azzeccava, ma la sua fama finì col non tollerare i dubbi. Per anni, periti e geologi imbottiti di scienza avrebbero dovuto fare i conti con l'infallibilità che la credenza popolare attribuiva a quell'uomo bizzarro. Per anni, zolfatari praticoni, saggiando collinette gessose o inseguendo i nauseabondi rigagnoli di acque sulfuree, sul punto di decidere se buttarsi nell'impresa delle trivellazioni sarebbero stati tentati di chiedere consiglio ad un ubriacone ruttante.

NOTA

All'inizio avevo pensato ad un racconto che raccogliesse le cose che mio padre, grottese, mi narrava sugli zolfatari del suo paese. In parte mi sono attenuto al programma (perciò questa storia va considerata immaginaria), ma ho aggiunto qualcos'altro e, con mia stessa sorpresa, ne è venuta fuori una sorta di epopea degli zolfatari siciliani. E non poteva non essere così, dato che un paio di secoli di storia siciliana sono stati, se non proprio condizionati, caratterizzati almeno dalla situazione di monopolio obiettivo che la Siciliaha avuto nel campo dell'estrazione zolfifera. La Sicilia, insomma, avrebbe potuto fare con lo zolfo quello che i paesi arabi fanno con il petrolio. Avrebbe potuto e non l'ha fatto: ecco, scrivendo degli zolfatari che mio padre conobbe, mi sono imbattuto in uno dei nodi dell'intricato, irrimediabile dissesto economico della Sicilia. E il risultato complessivo non poteva che dare il senso di un fallimento, l'eterno fallimento delle speranze dei siciliani.

m.c.

INDICE

PARTE PRIMA

I
L'ubriaco che sapeva auscultare la terra 6
II
Quando e come i contadini incubarono il malanno 8
III
Il mare, soffrendo, diede alla luce un'isoletta 14
IV
Perché dalla poco attraente Grotte i viandanti giravano al largo 19
V
Il gigantesco Bartolomeo e la sua smania di viaggiare 24
VI
Una lettera giunse dall'America 27
VII
Pur mangiando carne gli zolfatari diventarono incartapecoriti 30
VIII
Il primo ritorno del Viaggiatore 33
IX
Disseppellirono un cadaverino dagli occhi incrostati 36
X
Di un elegante cavaliere e della sua misteriosa missione 40
XI
Perché la piazza tornò a brulicare di sfaccendati 43
XII
Quelle che avvistò l'equivoco mediatore, erano navi armate di cannoni 46
XIII
Di come, con uno scatto di tigre, Bartolomeo Ardito afferrò la sua donna 50
XIV
Un alito di bestia malata infettò la terra 56
XV
I pericolosi amori di un carrettiere di Girgenti 62
XVI
Il mondo traslocava in America 67
XVII
Il secondo ritorno del Viaggiatore e la sua ultima visione 70

PARTE SECONDA

I
Tre colori messi insieme dall'audacia 78
II
Dove giovani imbottiti di scienza sono visti come fumo negli occhi 81
III
La sfrontatezza dei sacrileghi fece arrossire vallate e colline 84
IV
Un carrettiere pretese di umiliare il progresso 90
V
Dove si prende atto della complicità degli eucalipti 93
VI
Ma scoprirono che neanche i santi dei signori facevano una piega 97
VII
Un uomo sommariamente vestito sbandierò un quadernetto 100
VIII
La bizzarra mania di un soldato mancato 102
IX
Dove si riferisce di una chiacchierata serale tra notabili 105
X
Di quando una veterana maestrina fece sussultare il suo enorme seno 108
XI
Grugniva, il Primo Premio, trascinato in giro 115
XII
Un compromettente viaggio a Palermo 117
XIII
La tremarella contagiò i sediziosi compromessi 123
XIV
Quella mattina che segnava il trentanovesimo giorno di sciopero si udì, chiaro, il cric-crac degli otturatori 126
XV
Della fortuna che ebbe un nuovo passatempo 132
XVI
Non c'è cosa che uno fa e che qualche altro non veda 137
XVII
Il nipote del Viaggiatore scoprì che la Sicilia era un ammasso di pietre 140

XVIII
I reduci dalle divise logore raccontavano di un grande fiume e delle infami fucilazioni 146
XIX
Era pratico di armi lo sconosciuto che sparò al Commendatore 149
XX
Tutta d'oro appariva la Sicilia, ma nei suoi recessi nascondeva convegni osceni 151
XXI
L'avvenenza di una signora fece pericolosamente distrarre il barone Arcidiacono 157
XXII
Dove fedelmente si trascrive una lettera trovata per caso 160
XXIII
Quel giorno il paese vide i battaglioni del benessere 162

CONCLUSIONE 166

Appendice 17

FOGLIO DI STILE XSLT PER LA CREAZIONE DI UNA VERSIONE E-BOOK PDF DI BALTICO USANDO APACHE FOP (PASSANDO DA UN OUTPUT XSL-FO)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <xsl:output method="xml" encoding="iso-8859-1" indent="yes"/>
  <xsl:strip-space elements="p"/>
  <xsl:template match="/">
    <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
xmlns:fox="http://xml.apache.org/fop/extensions">
      <fo:layout-master-set>
        <fo:simple-page-master master-name="cover" page-height="23cm"
page-width="15.1cm">
          <fo:region-body/>
        </fo:simple-page-master>
        <fo:simple-page-master master-name="principale" page-
height="23cm" page-width="15.1cm" margin-top="1cm" margin-bottom="1cm"
margin-left="1.5cm" margin-right="1.5cm">
          <fo:region-before extent="1.5cm"/>
          <fo:region-after extent="1.5cm"/>
          <fo:region-body margin-top="1.8cm" margin-bottom="1.8cm"/>
        </fo:simple-page-master>
      </fo:layout-master-set>
      <!-- INIZIO APACHE FOP PDF outline extensions bookmark -->
      <xsl:for-each select="//div0">
        <fox:outline internal-destination="{@type}-{@n}">
          <fox:label>
            <xsl:value-of select="p"/>
          </fox:label>
          <xsl:for-each select="*/head[@type='tem']">
            <fox:outline internal-destination="{../@id}">
              <fox:label>
                <xsl:value-of select="../@n"/> - <xsl:value-of
select="translate(.,'&#10;',' ')"/>
<!--la funzione translate viene usata per eliminare i ritorno a capo
nei titoli head -->
              </fox:label>
            </fox:outline>
          </xsl:for-each>
        </fox:outline>
      </xsl:for-each>
      <xsl:for-each select="//div">
        <xsl:for-each select="head">
          <fox:outline internal-destination="{../@id}">
            <fox:label>
              <xsl:value-of select="."/>
            </fox:label>
          </fox:outline>
        </xsl:for-each>
      </xsl:for-each>
      <!—FINE APACHE FOP PDF outline extensions -->
      <xsl:apply-templates/>
    </fo:root>
```

```
  </xsl:template>
  <xsl:template match="teiHeader"/>
  <xsl:template match="text">
    <xsl:apply-templates/>
  </xsl:template>
  <!-- INIZIO FRONT -->
  <xsl:template match="front">
    <!-- copertina -->
    <fo:page-sequence master-reference="cover">
      <fo:flow flow-name="xsl-region-body">
        <fo:block text-align="center" vertical-align="middle">
          <fo:external-graphic src="c:\temp\PDF\copertinag.jpg"
width="15.1cm" height="23cm"/><!-- le dimensioni della copertina sono
uguali a quelle della pagina, il file usato è quello creato per la
versione lit -->
        </fo:block>
      </fo:flow>
    </fo:page-sequence>
    <xsl:apply-templates/>
    <!-- INIZIO dediche inserite manualmente -->
    <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
        <fo:block space-before="1cm" font-size="15pt" text-
align="end" font-family="Times Roman" font-style="italic">
A Bartolomeo Collura, mio padre.
    </fo:block>
      </fo:flow>
    </fo:page-sequence>
    <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
        <fo:block space-before="1cm" font-size="13pt" text-
align="justify" font-family="Times Roman" line-height="1.5em" margin-
right="0.5cm" margin-left="0.5cm">
Contagiati dal delirio delle escavazioni, subito accompagnato dalla
comparsa di affaristi scrocconi, si scoprirono impensate doti di
imprenditori; e sventrando valli e colline sognarono di arricchire,
mentre copioso colava lo zolfo e si ampliavano i cimiteri. Due secoli
di picconate cambiarono la faccia della terra. Subito si appales&#242;
il disastro, ma in quel turbinio di fortune immaginate nessuno vi fece
caso. Corsero ai ripari quando gi&#224; il vento screpolava gli spalti
delle zolfare e le erbacce cominciavano a nasconderne le bocche. Fu
come se un'ostinata bonaccia si fosse posata su un mare che era stato
in tempesta. Non lontano dai ruderi, oziosi, aspettarono sussidi e
pensioni; e polvere e silenzio sedimentarono sulla loro assurda
epopea.
    </fo:block>
      </fo:flow>
    </fo:page-sequence>
    <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body">
        <fo:block space-before="1cm" space-after="0.4cm" font-
size="13pt" text-align="left" font-family="Times Roman" line-
height="1.2em" margin-left="3.4cm">
Zufolava mentre andava al lavoro
e parlava spesso di un futuro di
benessere e di abbondanza
    </fo:block>
        <fo:block font-size="13pt" text-align="end" font-
family="Times Roman" font-variant="small-caps">
Sherwood Anderson,
  </fo:block>
```

```
          <fo:block font-size="13pt" text-align="end" font-
family="Times Roman" font-style="italic">
Un povero
bianco.&amp;#160;&amp;#160;&amp;#160;&amp;#160;&amp;#160;&amp;#160;&amp;#160;&amp;#160;
  </fo:block>
       </fo:flow>
     </fo:page-sequence>
     <!-- FINE dediche inserite manualmente -->
  </xsl:template>
  <!-- le dediche non sono state elaborate -->
  <xsl:template match="div[@type='ded']" priority="1"/>
  <xsl:template match="div[@type='ep']" priority="1"/>
  <xsl:template match="titlePage">
     <fo:page-sequence master-reference="principale">
       <fo:flow flow-name="xsl-region-body">
         <xsl:apply-templates/>
       </fo:flow>
     </fo:page-sequence>
  </xsl:template>
  <xsl:template match="docAuthor">
     <fo:block space-after="3.8cm" font-size="23pt" text-align="center"
font-weight="bold" font-family="Times Roman">
       <xsl:apply-templates/>
     </fo:block>
  </xsl:template>
  <xsl:template match="docTitle">
     <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="titlePart[@type='main']">
     <fo:block space-after="10pt" font-size="30pt" text-align="center"
font-weight="bold" font-family="Times Roman">
       <xsl:apply-templates/>
     </fo:block>
  </xsl:template>
  <xsl:template match="titlePart[@type='sub']">
     <fo:block space-after="8.7cm" font-size="18pt" text-align="center"
font-weight="bold" font-family="Times Roman">
       <xsl:apply-templates/>
     </fo:block>
  </xsl:template>
  <xsl:template match="docImprint">
     <fo:block font-size="18pt" text-align="center" font-weight="bold"
font-family="Times Roman">
       <xsl:apply-templates/>
     </fo:block>
  </xsl:template>
  <!-- FINE FRONT -->
  <xsl:template match="body">
     <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="div0">
     <fo:page-sequence master-reference="principale">
       <fo:flow flow-name="xsl-region-body">
         <fo:block space-after="15pt" font-size="20pt" text-
align="center" font-weight="bold" font-family="Times Roman"
id="{@type}-{@n}">
           <xsl:value-of select="p"/>
         </fo:block>
       </fo:flow>
     </fo:page-sequence>
     <xsl:apply-templates/>
```

```
  </xsl:template>
  <xsl:template match="div0/p"/>
  <xsl:template match="div1">
    <fo:page-sequence master-reference="principale">
      <fo:static-content flow-name="xsl-region-before">
        <fo:block text-align="center" font-size="12pt" font-
family="sans-serif" font-weight="normal" font-style="italic"> MATTEO
COLLURA - BALTICO: edizione elettronica</fo:block>
      </fo:static-content>
      <fo:static-content flow-name="xsl-region-after" margin-
top="0.5cm">
        <fo:block space-before="0.7cm" text-align="center" font-
size="12pt" font-family="sans-serif" font-weight="normal">
          <fo:page-number/>
        </fo:block>
      </fo:static-content>
      <fo:flow flow-name="xsl-region-body" text-align="justify" font-
size="14pt" font-family="Verdana, Geneva, Arial, Helvetica, sans-
serif" font-weight="normal" line-height="1.7em">
        <xsl:apply-templates/>
      </fo:flow>
    </fo:page-sequence>
  </xsl:template>
  <xsl:template match="head[@type='ord']">
    <fo:block space-after="17pt" font-size="20pt" text-align="left"
font-weight="bold" font-family="Times Roman" id="{../@id}">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="head[@type='conv']">
    <fo:block space-after="17pt" font-size="20pt" text-align="left"
font-weight="bold" font-family="Times Roman" id="{../@id}">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="head[@type='tem']">
    <fo:block space-after="20pt" font-size="17pt" text-align="left"
font-weight="bold" font-family="Times Roman">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="p">
    <fo:block text-indent="24pt">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="back">
    <fo:page-sequence master-reference="principale">
      <fo:flow flow-name="xsl-region-body" text-align="justify" font-
size="14pt" font-family="Verdana, Geneva, Arial, Helvetica, sans-
serif" font-weight="normal" line-height="1.7em">
        <xsl:apply-templates/>
      </fo:flow>
    </fo:page-sequence>
  </xsl:template>
  <xsl:template match="div/signed">
    <fo:block text-align="end" font-size="14pt" font-family="Verdana,
Geneva, Arial, Helvetica, sans-serif" font-weight="normal">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
```

```
  <xsl:template match="l">
    <fo:block>
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="*[@rend='bloc']">
    <fo:block margin-left="3.5cm" margin-right="2.5cm">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
  <xsl:template match="*[@rend='italic']">
    <fo:inline font-style="italic">
      <xsl:apply-templates/>
    </fo:inline>
  </xsl:template>

 <xsl:template match="q[@type='epistola']">
  <fo:block text-indent="24pt"><xsl:apply-templates/></fo:block>
  </xsl:template>
</xsl:stylesheet>
```

III.17. ANALISI ED INTERPRETAZIONE

Sin da principio tra gli obiettivi del lavoro che stiamo in queste pagine illustrando non vi sono stati l'analisi e lo studio "tradizionale" del romanzo di Matteo Collura Baltico, il quale più che altro ha costituito la "cavia" per i nostri esperimenti di codifica e trattamento informatico del testo; nonostante ciò, nel momento in cui ci apprestiamo a concludere la nostra esposizione, appare utile il tentativo di fornire un abbozzo di analisi dell'opera letteraria, da noi sottoposta a codifica, giovandoci del supporto di tutti quegli strumenti che sino ad ora sono stati, per lo più, adoperati per la rappresentazione del testo piuttosto che per la sua interpretazione; potremmo dire che proprio su quella linea "sperimentale", che ha un po' costituito il filo conduttore delle nostre attività si inseriscono i lavori che illustreremo fra breve e che primariamente mirano a dimostrare l'utilità della codifica elettronica anche nell'analisi dei testi.

Un'epopea siciliana, recita il sottotitolo del romanzo di Collura fornendoci un'importante chiave di lettura dell'opera, in cui lo scrittore agrigentino mette in scena l'epopea dello zolfo e degli zolfatari. La vicenda si sviluppa in un arco di tempo, che va dalla prima metà dell'ottocento (il riferimento temporale è fornito dalla narrazione dell'emersione nel Canale di Sicilia dell'isola Ferdinandea) sino agli anni Sessanta del Novecento. Teatro della vicenda è la cittadina di Grotte in provincia di Agrigento, paese natale del padre dello scrittore, elemento questo rilevante in quanto ha costituito il primo spunto per la genesi del romanzo, come ci rivela lo stesso scrittore nella nota finale: "All'inizio avevo pensato ad un racconto che raccogliesse le cose che mio padre, grottese, mi narrava sugli zolfatari del suo paese". Ma nella narrazione, al livello della memoria si affianca quello della storia, e così il racconto finisce per diventare non soltanto la storia dell'ascesa e del successivo declino di un paese di zolfatari, ma delle opportunità mancate, delle aspettative disattese e, sono parole dello scrittore, "dell'eterno fallimento delle

speranze dei siciliani". Baltico è un romanzo atipico, come brillantemente ha osservato la Prof. Lucrezia Lorenzini in occasione del suo intervento *"Microstorie frantumate" in Baltico di Matteo Collura*[232], costituito da "microstorie frantumate", un insieme di racconti strettamente interrelati fra di loro in modo tale da costituire un unicum narrativo; nota ancora Lucrezia Lorenzini come Baltico sia "racconto di documenti e di favole, di universo mentale e di mito", romanzo corale che attraverso una folla di personaggi rappresenta il tema principale dell'avventura delle zolfare, su cui si innesta l'altro grande tema del romanzo, l'emigrazione dei siciliani a cavallo fra Ottocento e Novecento. Ma questi temi non vengono svolti attraverso gli occhi e le vicende di un personaggio principale; il narratore eterodiegetico attraverso i casi e le parole di una folta pletora di personaggi, in molti casi figure appena sbozzate o addirittura voci indefinite di un vasto "coro", tesse una tela di vicende e rapporti che costituiscono il tessuto della narrazione; una narrazione fatta di "documenti e favole", di esperienze umane e vicende storiche, di "universo mentale e mito"; un caleidoscopio di situazioni e figure, in cui non spicca un protagonista, eroe dell'azione su gli altri personaggi, fatta eccezione forse per il "gigante" Bartolomeo; ma a ben guardare anche la vicenda di Bartolomeo Ardito e del suo desiderio di vedere il Baltico, metafora dell'"altrove", del sogno e della speranza che diverranno l'America, l'Australia e tutte quelle terre in cui approderanno i numerosi siciliani in fuga dalla miseria e dall'arretratezza: è soltanto una delle tante storie, uno dei tanti fili intrecciati che, come in un arazzo, formano l'intero disegno.

Potremmo forse dire che è proprio il paese di Grotte il protagonista della narrazione, Grotte nel racconto rappresenta l'osservatorio della realtà isolana durante la stagione dello zolfo, osservatorio che tuttavia assurge a metafora e simbolo della Sicilia tutta, microcosmo che rispecchia il macrocosmo di cui fa parte.

---

[232] Cfr. LUCREZIA LORENZINI, *"Microstorie frantumate" in Baltico di Matteo Collura*, in AA.VV., *Atti del VII Convegno Internazionale S.I.L.F.I. Genesi, architetture e forme testuali,* (Roma1-5 ottobre 2002), pp.223-232 (in corso di stampa).

Dopo questa necessaria premessa passiamo a vedere come abbiamo ipotizzato si potesse utilizzare il testo codificato di Baltico per cercare di investigare l'opera. È, tuttavia, bene puntualizzare che quello che ci apprestiamo a mostrare è soltanto un esperimento, una esemplificazione delle opportunità offerte dalla codifica dichiarativa XML/TEI agli studiosi, con tutte le limitazioni del caso. Occorre, inoltre precisare che la nostra codifica dell'opera di Collura non è stata rivolta alle esigenze dell'interpretazione e dell'analisi, ma è stata piuttosto una codifica "minima" con finalità principalmente di distribuzione ed integrazione in archivi digitali.

D'altronde, la codifica stessa può essere funzionale all'indagine letteraria e prospettarsi, per certi versi, come attività di interpretazione del testo, fornendo così un nuovo utile strumento agli studiosi.

Se si ricorda la definizione che di codifica informatica del testo abbiamo fornito, seguendo le idee di Fabio Ciotti, all'inizio della trattazione come *rappresentazione formale di un testo ad un qualche livello descrittivo*, si comprenderà il valore che la codifica può rappresentare anche in attività di studio dei testi, ove appunto il fine della marcatura del testo sia orientato alla rappresentazione di un *livello descrittivo* utile all'analisi dello stesso; ad esempio, sono realizzabili codifiche dei testi indirizzate ad evidenziare fenomeni linguistici oppure ancora l'evoluzione del testo nelle varianti, oltre ad innumerevoli altri tipi di codifica caratterizzate da un approccio al testo ogni volta differente sulla base di diverse esigenze analitiche. Il codificatore è in primo luogo un lettore "attento" del testo che evidenzia, annota, mediante il markup determinate porzioni di testo sulla base di specifiche esigenze, rappresentative o interpretative, in vista di un successivo trattamento delle informazioni da lui codificate con l'ausilio di strumenti informatici. Vorremmo dire che la codifica informatica dichiarativa dei testi potrebbe prospettarsi come una forma di approccio alternativo all'analisi delle opere letterarie, quasi un'attività propedeutica all'interpretazione ed allo studio dei testi, ferma restando l'importanza delle forme tradizionali di indagine; è bene, tuttavia, tener presente

che l'analisi eseguita con l'ausilio di strumenti informatici non è da vedersi come sostitutiva delle forme tradizionali di approccio analitico ai testi letterari; semmai, va considerata come complementare ed integrativa proprio a quelle metodiche di studio.

Un piccolo esempio di quanto appena sostenuto lo abbiamo cercato di fornire nei lavori che ci apprestiamo ad illustrare. Torniamo a rammentare che quella che stiamo per mostrare è solo un'esemplificazione delle potenzialità offerte allo studioso dalle codifiche dichiarative nell'interpretazione dei testi; ricordiamo, inoltre, che quella da noi effettuata è stata una codifica "minimale": avremmo, infatti, potuto soffermarci su particolari aspetti del testo oppure cercare di sottolineare particolari livelli della narrazione o eventuali fenomeni linguistici per poi sottoporli ad analisi con gli strumenti automatici, ma in sede di codifica non siamo stati mossi dalle esigenze dell'analisi letteraria, ma soprattutto, come detto, da quelle della rappresentazione del testo. Nonostante ciò siamo riusciti senza eccessivo sforzo ad approntare due piccoli lavori, che esemplificano e dimostrano come è possibile utilizzare gli strumenti dedicati al trattamento dei dati XML, piegandoli ed adattandoli agli scopi dell'analisi ed interpretazione dei testi.

Abbiamo precedentemente affermato che uno dei caratteri distintivi di Baltico è la "coralità", proprio per aiutarci ad analizzare questo aspetto dell'opera abbiamo approntato il primo foglio di stile XSLT che adesso passiamo ad analizzare ed il cui codice, come di consueto, è riportato in appendice. Finalità di questo foglio è quella di fornire alcuni dati sul testo, naturalmente opera sulla base degli elementi da noi codificati nel sorgente XML; l'output in formato puro testo ci offre alcuni dati sul numero dei personaggi (suddividendoli per tipologia) e dei discorsi diretti, dopodiché restituisce l'elenco completo dei personaggi e dei parlanti. Passando ad un'analisi più precisa del foglio XSLT vediamo che il primo comando `<xsl:value-of>` opera su tutti gli elementi <*name*> del sorgente in cui il valore dell'attributo *type* sia *persona.fit*, questo comando in realtà non fornisce come indicato nell'etichetta dell'output il numero dei personaggi di fantasia del testo, bensì

indica il numero di tutti gli elementi <*name*> che hanno un attributo *type=persona.fit*; è ovvio che l'utilità di questo componente come del successivo, considerando che normalmente un nome ricorre per più di una volta, è abbastanza opinabile, abbiamo deciso comunque di mantenerli per mostrare con diversi esempi come sia possibile estrapolare dati di vario tipo e natura (sulla base degli elementi codificati) da un sorgente XML/TEI.

In Baltico la coralità si estrinseca principalmente mediante l'ampio ricorso al discorso diretto, serve ad evidenziare questo aspetto della scrittura di Collura il terzo comando `<xsl:value-of>` del foglio di stile che stiamo osservando, che ci fornisce il numero totale dei discorsi diretti presenti nel testo, come sappiamo identificati nel sorgente dal tag <*q*> accompagnato dall'attributo type con valore diretto; l'output dell'elaborazione ci informa che lo scrittore agrigentino si è servito nella sua narrazione del discorso diretto per ben 827 volte, fatto questo che attesta l'importanza del dialogo drammatico nell'economia del racconto, denotato dalla presenza di una narrazione a focalizzazione zero assai discreta che spesso preferisce commentare o raccontare attraverso il dialogo tra i numerosi personaggi "coro". Proprio sul "coro" dei personaggi focalizza la propria attenzione il resto del foglio di stile che stiamo esaminando. Abbiamo precedentemente affermato che spesso sono personaggi indefiniti, figure appena sbozzate, le voci di questo vasto "coro"; cerca di evidenziare questo aspetto il quarto elemento `<xsl:value-of>` del foglio XSLT, il quale restituisce il numero di tutti gli elementi <*q*> in cui il parlante sia di tipo indefinito, o meglio, per esser precisi, in cui nell'attributo *who*, che specifica il parlante, sia contenuto il termine *indefinito* oppure *indefiniti*. L'output ci informa che ciò accade per ben 246 volte, fatti i dovuti calcoli vediamo che ben il 29.75%, quindi quasi un terzo, dei discorsi diretti del romanzo è pronunciato da personaggi indefiniti, si tenga inoltre presente che questo numero è errato per difetto, in quanto in fase di codifica non abbiamo dedicato ampia attenzione a questo aspetto, anzi dobbiamo confessare di esserci resi conto della rilevanza di questo fenomeno proprio esaminando i dati preliminari delle prime versioni di

prova di questo foglio di stile, quindi un buon numero di personaggi che potremmo far rientrare nella categoria degli "indefiniti" è identificato mediante un appellativo generico (ad es. brigadiere, capicantiere) ovvero un soprannome o un'altra caratteristica, sarebbe bastato in sede di codifica aggiungere all'identificativo contenuto nell'attributo *who* di <*q*> il termine *indefinito/i* per farli rientrare nel computo indicato sopra[233]. Proprio nel tentativo di approfondire il tema del numero dei personaggi e soprattutto dei parlanti abbiamo impostato le due elaborazioni `<xsl:for-each>` del foglio di stile, la prima restituisce l'elenco di tutti i nomi di personaggi di fantasia, ben più utile, la seconda invece fornisce la sequenza, disposta in ordine alfabetico, di tutti i valori degli attributi *who* degli elementi <*q*> marcanti discorsi diretti, e quindi, in pratica, l'elenco di tutti i parlanti. Visti così questi dati possono apparire poco utili; in realtà, invece, trattati con l'ausilio di altri programmi riescono a fornirci delle valide informazioni; nello specifico abbiamo operato in questo modo: abbiamo dapprima importato l'elenco dei parlanti nel freeware NoteTab e, dopo aver sostituto con degli underscore (trattini bassi) gli spazi nei nomi, ci siamo serviti del comando Text Statistics per avere alcune utili informazioni sui nostri dati, quali il numero di occorrenze, la frequenza e la percentuale di presenza dei vari nomi, dopodiché abbiamo utilizzato un foglio di calcolo[234] per ordinare i dati; di seguito riportiamo l'elenco delle prime venti posizioni disposte in ordine decrescente in base alla frequenza di apparizione:

Parlante	Frequenza	%
personaggio_indefinito	168	20.36
personaggi_indefiniti	45	5.45
Bartolomeo_Ardito	25	3.03
Vittorio_Scuderi	24	2,91

[233] Occorre precisare che la codifica XML/TEI può esser vista come una codifica di tipo incrementale, infatti nulla vieta in un secondo momento di modificare i valori di attributo in questione ovvero di aggiungere nuovi elementi.
[234] Nello specifico ci siamo serviti sia di Microsoft Excel, sia di Open Office Calc.

popolo_indefinito	20	2.42
Giuseppe_Agozzino	19	2.30
Raimondo_Battaglia	17	2.06
Gerlando_Boccadoro	15	1,82
Speranza_Ardito	14	1,70
Filippo_Salvaggio	14	1,70
Raimondo_Dileo	12	1.45
uomo_dagli_occhialini_dorati	11	1.33
Serafino_Collica	11	1.33
Ignazio_Pirrera	10	1.21
capitano	10	1.21
Attilio_Barreca	10	1.21
Michelangelo_Teseo	9	1.09
Bartolomeo_Ardito_-_nipote	9	1.09

Si noti la preponderanza dei personaggi indefiniti nei discorsi diretti, da soli i valori "personaggio indefinito" e "personaggi indefiniti" rappresentano ben il 25% circa dei parlanti nei discorsi diretti. Continuando ad esaminare questi dati è possibile effettuare altre considerazioni, l'elevato numero totale di personaggi parlanti, ben 174, offre un'ulteriore riprova del fatto che il romanzo sia costituito da "microstorie frantumate", si noti anche quanto numerosi siano quei personaggi che pronunciano solamente una o due battute, a dimostrazione della natura corale dei racconti. Altre considerazioni si potrebbero trarre continuando ad analizzare questi dati, d'altra parte anche altri dati avremmo probabilmente potuto ottenere dal nostro sorgente XML/TEI, ma un'analisi più ampia esula dagli obiettivi che ci siamo prefissati, rimandiamo questo lavoro ad altre circostanze, in questa occasione abbiamo soltanto voluto offrire un superficiale esempio delle possibilità che le codifiche dichiarative e gli strumenti informatici offrono agli studiosi come ausilio nell'analisi testuale.

Col primo esempio abbiamo visto un foglio che forniva delle statistiche sul testo, col secondo vediamo come sia possibile creare degli output di navigazione del testo specificamente concepiti per esigenze di analisi. In particolare con questo secondo progetto abbiamo voluto dimostrare come sia possibile creare un file ipertestuale che punti a tutte le occorrenze di un determinato elemento, e quindi utile per lo studio e l'interpretazione di determinati fenomeni. Nello specifico, a

titolo esemplificativo, abbiamo realizzato un piccolo lavoro che crea un output HTML del testo di Baltico in cui tutti questi nomi sono evidenziati dal colore rosso nella riproduzione video, ed identificati mediante un ancora nel codice, in modo tale che fossimo in grado di realizzare una sorta di indice ipertestuale che consenta di puntare a tutte le occorrenze di un certo nome. In primo luogo abbiamo scritto un foglio di stile che ricreasse il codice XML di Baltico aggiungendo un attributo *id* con valore univoco a tutti gli elementi <*name*> del sorgente, in modo tale da poi poter generare a partire da questi *id* il nome (ossia la destinazione) delle varie ancore HTML cui gli indici dovranno puntare; si occupa di ciò il foglio dal nome *copia e aggiungi attributi.xsl*, il quale non fa altro che copiare l'intero sorgente XML e, mediante il template per l'elemento <*name*>, aggiungere un attributo *id* il cui valore è costituito dal valore dell'attributo *id* dell'elemento <*div1*> in cui <*name*> è contenuto e da un codice univoco generato dalla funzione generate-id().

Il secondo foglio di stile, da noi chiamato *HTML con ancore.xsl*[235], non fa altro che creare un output HTML di Baltico a partire dal nuovo sorgente XML arricchito con gli attributi *id* per gli elementi <*name*> prodotto col precedente foglio XSLT. In sostanza non si tratta altro che del medesimo foglio di stile per l'output HTML del romanzo che già abbiamo visto in precedenza, con l'unica differenza che è stato aggiunto un template per gli elementi <*name*> che si occupa di creare nell'output HTML un ancora in corrispondenza di ogni nome, di seguito ne riportiamo il codice[236]:

```
<xsl:template match="name">
```

[235] Evitiamo di riportare in appendice il codice di questo foglio di stile, in quanto a parte le poche differenze che abbiamo illustrato nella trattazione è in tutto e per tutto identico al foglio per l'output HTML di Baltico.

[236] Si noti che la colorazione rossa di tutti i nomi nell'output è stata ottenuta mediante una classe CSS specificata nell'head HTML, questo è il codice:

```
.nome {font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size: 12pt; color:red;}
```

```
        <a name="{@id}" class="nome">
        <xsl:apply-templates/></a>
    </xsl:template>
```

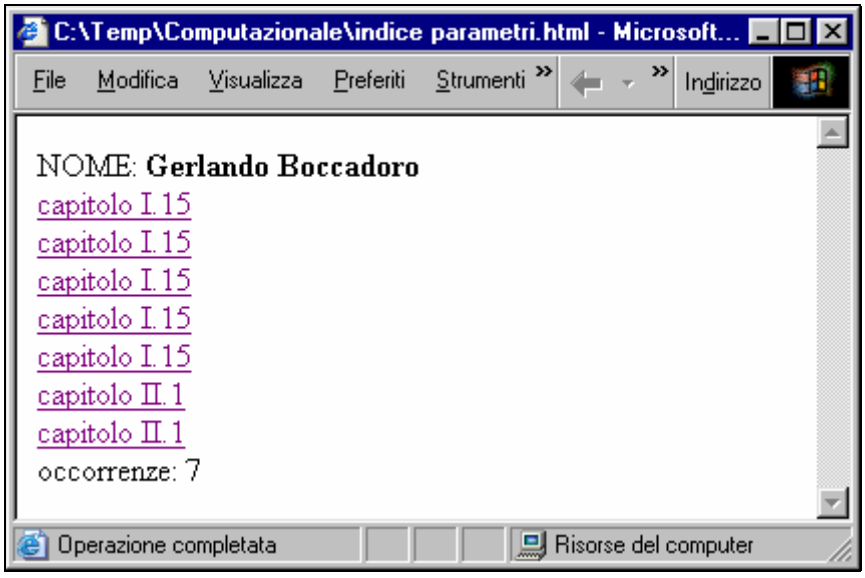
Il terzo foglio di stile viene utilizzato per la creazione degli indici di navigazione; di seguito ne riportiamo il codice[237]:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output encoding="ISO-8859-1"/>
    <xsl:param name="pp">Gerlando Boccadoro</xsl:param>
    <xsl:template match="/">
        <xsl:apply-templates/>
    </xsl:template>
    <xsl:template match="text">
    NOME: <b><xsl:value-of select="$pp"/></b><br/>
        <xsl:for-each select="//name[@key=$pp]">
            <xsl:sort select="@key"/>
            <a href="baltico.htm#{@id}">capitolo <xsl:value-of
select="./ancestor::div1/@id"/></a><br/>
            </xsl:for-each>
        occorrenze: <xsl:value-of
select="count(//name[@key=$pp])"/>
    </xsl:template>
</xsl:stylesheet>
```

Questo codice XSLT opera sul sorgente XML modificato dal primo foglio di stile (quello con gli attributi id per gli elementi <*name*>) e crea una piccola interfaccia di navigazione che consente di puntare nel nuovo file HTML creato dal secondo foglio XSLT, alle occorrenze di un determinato nome. Di seguito riportiamo una

[237] Per comodità non sono stati inseriti i tag che definiscono la struttura base del file HTML, ciò nonostante anche in questo modo l'output funziona.

schermata di questo output HTML visualizzata dentro il browser Internet Explorer 6.0:



Come vediamo troviamo il nome analizzato[238], un link per ogni occorrenza in cui è specificato il capitolo dove questa compare ed infine il computo totale delle occorrenze; cliccando su uno dei link si viene portati nel punto preciso dell'output HTML dove compare il nome in questione[239]. Quello che abbiamo qui mostrato è soltanto un esempio senza una grande utilità effettiva nel caso concreto per lo studio del testo, quello che abbiamo voluto qui mostrare è l'opportunità, in accordo con l'attività di codifica, di costruire degli strumenti di navigazione ed interrogazione del testo che possano agevolare lo studioso durante la lettura e l'interrogazione delle opere letterarie per le necessità dell'analisi. Ad esempio usando le medesime metodiche qui illustrate si possono costruire degli indici di navigazione per ritrovare velocemente tutti i discorsi diretti di un certo personaggio ovvero con certe caratteristiche, o ancora indici di navigazione per qualsivoglia elemento, e quindi fenomeno testuale (fonetico, narrativo, linguistico, etc.) opportunamente codificato, tutto sta all'acume ed all'estro dello studioso.

[238] Il nome viene specificato mediante parametri con le procedure consuete.
[239] Oltre questo foglio di stile ne è stato realizzato un altro che non funziona mediante parametri e che restituisce l'intero elenco dei nomi sotto forma di link, sul modello dell'output illustrato nella trattazione, il codice di questo foglio è riportato in appendice.

Appendice 18

FOGLIO DI STILE PER L'ESTRAPOLAZIONE DI ALCUNI DATI UTILI ALL'ANALISI DI BALTICO

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="text"/>
  <xsl:template match="/">
Numero Personaggi .fit: <xsl:value-of
select="count(//name[@type='persona.fit'])"/>
Numero Personaggi .Real: <xsl:value-of
select="count(//name[@type='persona.real'])"/>
Numero Discorsi diretti: <xsl:value-of
select="count(//q[@type='diretto'])"/>
Numero Personaggi indefiniti: <xsl:value-of
select="count(//q[contains(@who,'indefinito')])+count(//q[contains(@wh
o,'indefiniti')])"/>
<!-- questa espressione manda in output il numero (ottenuto grazie
alla funzione count) di tutti in nodi q il cui attributo who contiene
la parola indefinito più tutti in nodi q il cui attributo who contiene
la parola indefiniti -->

ELENCO PERSONAGGI

    <xsl:for-each select="//name[@type='persona.fit']">
<xsl:text>&#10;</xsl:text>
      <xsl:sort select="@key"/>
      <xsl:value-of select="@key"/>
    </xsl:for-each>

ELENCO PARLANTI

<xsl:for-each select="//q[@type='diretto']">
<xsl:text>&#10;</xsl:text>
      <xsl:sort select="@who"/>
      <xsl:value-of select="@who"/>
    </xsl:for-each>
  </xsl:template>

  <xsl:template match="teiHeader"/>
  <xsl:template match="text"/>
</xsl:stylesheet>
```

Appendice 19

Stampa del foglio di calcolo con cui sono stati elaborati i dati sui discorsi diretti del romanzo.

Parlante	Frequenza	%
personaggio_indefinito	168	20,31
personaggi_indefiniti	45	5,44
Bartolomeo_Ardito	25	3,02
Vittorio_Scuderi	24	2,90
popolo_indefinito	20	2,42
Giuseppe_Agozzino	19	2,30
Raimondo_Battaglia	17	2,06
Gerlando_Boccadoro	15	1,81
Speranza_Ardito	14	1,69
Filippo_Salvaggio	14	1,69
Raimondo_Dileo	12	1,45
uomo_dagli_occhialini_dorati	11	1,33
Serafino_Collica	11	1,33
Ignazio_Pirrera	10	1,21
capitano	10	1,21
Attilio_Barreca	10	1,21
Michelangelo_Teseo	9	1,09
Bartolomeo_Ardito_-_nipote	9	1,09
Arcidiacono	9	1,09
Venerando_Scuderi	8	0,97
Sebastiano_Azzalora	8	0,97
Gueli	8	0,97
Francesco_Ardito	8	0,97
Filippo_Melisenda	8	0,97
capoccia_intermediari	8	0,97
Santo_Ingrao	7	0,85
Salvatore_Maravantano	7	0,85
Antonino_Licata	7	0,85
Giovanni_Todaro	6	0,73
Gaetano_Vella	6	0,73
cavaliere	6	0,73
vecchio_ulceroso	5	0,60
Sindaco	5	0,60
Silvestro	5	0,60
Raffaele_Baldo	5	0,60
mediatore	5	0,60
Francesco_De_Luca	5	0,60
Calogero_Spitali	5	0,60
Archimede_Galioto	5	0,60
Agostino_Cinquemani	5	0,60
Accursio_Rametta	5	0,60
Accursio_Pilato	5	0,60
vecchio_monaco	4	0,48
Salvatore_Parrinello	4	0,48
oratore_indefinito	4	0,48
moglie_Filippo_Salvaggio	4	0,48

Direttore	4	0,48
Deputato_indefinito	4	0,48
Cesare_Valenti	4	0,48
Carmelo_Castronovo	4	0,48
Biagio_Tascarella	4	0,48
Baldassare_Morreale	4	0,48
arciprete	4	0,48
uomo_dai_rattoppi_multicolori	3	0,36
Salvatore_Elia	3	0,36
Questore	3	0,36
padre_di_Speranza_Ardito	3	0,36
Michele_Mancuso	3	0,36
medico_di_Caltanissetta	3	0,36
Maurizio_Bellanca	3	0,36
Giudice_Regio	3	0,36
Furino	3	0,36
Francesco_Pumello	3	0,36
Filippo_Maida	3	0,36
esercente_di_Casteltermini	3	0,36
erborista	3	0,36
Delegato_di_Pubblica_Sicurezza	3	0,36
Calogero_Saccone	3	0,36
Calogero_Bellomo	3	0,36
brigadiere	3	0,36
Benintendi	3	0,36
Amilcare_Ciulla	3	0,36
zolfatari_scioperanti	2	0,24
Vuturo	2	0,24
uomo_dai_baffetti_coltivati	2	0,24
Umberto_Mancuso	2	0,24
ubriacone	2	0,24
Ubaldo_Cosentino	2	0,24
Tortorici	2	0,24
sbirri	2	0,24
Salvatore_Carpinello	2	0,24
Rosario_Brandino	2	0,24
prete	2	0,24
Luigi_Galioto	2	0,24
Lorenzo_Sferrazza	2	0,24
Giuseppe_Fastuca	2	0,24
Giovanni_Soresano	2	0,24
Giacinto_Favata	2	0,24
Gabriele_Zummo	2	0,24
Fioretta	2	0,24
Fiorenza	2	0,24
Ferdinando_Spirio	2	0,24
Emanuele_Cupani	2	0,24
Emanuele_Cosentino	2	0,24
Diego_Vadala	2	0,24
Diego_Carrubba	2	0,24
Cosimo_Notonica	2	0,24
carrettiere	2	0,24
capicantiere	2	0,24
Alfonso_Randazzo	2	0,24

Alessio_Carlisi	2	0,24
Accursio_Cutaia	2	0,24
Vincenzo_Iapichino	1	0,12
terza_e_quarta_elementare	1	0,12
Teresa_Battaglia	1	0,12
tenente_commissione_reclutamento_leva	1	0,12
sovversivo	1	0,12
sottufficiale	1	0,12
soci_Battaglia_Belladonna	1	0,12
sicari	1	0,12
Salvatore_Tumbiolo	1	0,12
Salvatore_Stella	1	0,12
Rosario_Badalamenti	1	0,12
Rita_Pilato	1	0,12
preti_indefiniti	1	0,12
Pasquale_Pagliarello	1	0,12
parente_indefinito_Bartolomeo_Ardito	1	0,12
Paolo_Sessa	1	0,12
Paolo_Amoroso	1	0,12
padre_di_Bartolomeo_Ardito	1	0,12
Ottavio_Argento	1	0,12
Nicolo_DAvanzo	1	0,12
moglie_Michele_Bellavia	1	0,12
moglie_Agostino_Cinquemani	1	0,12
Michele_Sinatra	1	0,12
Michele_Burgio	1	0,12
Michele_Belladonna	1	0,12
medico_indefinito	1	0,12
medico_di_Girgenti	1	0,12
mediatori_d'affari	1	0,12
Matteo_Alfano	1	0,12
Martino_Ciraolo	1	0,12
Mario_Zaffuto	1	0,12
Maria_Spartivento	1	0,12
Margherita_Chimento	1	0,12
Manlio_Zaffuto	1	0,12
madre_di_Bartolomeo_Ardito	1	0,12
Lorenzo_Palillo	1	0,12
Lorenzo_Giangrande	1	0,12
Lorenzo_Costanza	1	0,12
Liborio_Bellavia	1	0,12
ispettore_indefinito	1	0,12
Ignazia_Sciortino	1	0,12
Giuseppe_Parodi	1	0,12
Gioacchino_Piscopo	1	0,12
Giacomo_Cimino	1	0,12
galantuomini	1	0,12
Francesco_Pillitteri	1	0,12
Figlio_indefinito_Bartolomeo_Ardito	1	0,12
familiari_Bartolomeo_Ardito	1	0,12
Eduardo_Micciche	1	0,12
donne_indefinite	1	0,12
Don_Virgilio	1	0,12
Domenico_Turano	1	0,12

Domenica_Terrana	1	0,12
Diego_Pilato	1	0,12
decifratore_di_missive	1	0,12
Cosimo_Lo_Presti	1	0,12
Cinquemani	1	0,12
Cesare_Baio	1	0,12
cercatori	1	0,12
carrettieri	1	0,12
bambini_seconda_classe	1	0,12
bambini_prima_classe	1	0,12
Baio	1	0,12
Antonio_Licata	1	0,12
Antonino_Cardella	1	0,12
Angioletta_Maravantano	1	0,12
Angelo_Rizzo	1	0,12
Angelo_Cipolla	1	0,12
Angelo Arnone	1	0,12
Alfonso_Triolo	1	0,12
Agostino_Palmieri	1	0,12
Abramo_Ingrassia	1	0,12
Totale discorsi diretti		**827**
Totale Parlanti		**174**

Appendice 20

FOGLIO DI STILE XSLT "copia e aggiungi attributi.xsl" PER AGGIUNGERE UN'ATTRIBUTO ID CON VALORE UNIVOCO A TUTTI GLI ELEMENTI NAME

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet                                          version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output     method="xml"     version="1.0"     encoding="UTF-8"
indent="yes"/>
  <xsl:strip-space elements="*"/>
  <xsl:template match="/">
    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="TEI.2">
    <xsl:copy>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>
  <xsl:template match="teiHeader"/>
  <xsl:template match="text">
    <xsl:copy>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>

  <xsl:template match="back"/>

  <xsl:template match="front"/>

  <xsl:template match="body">
    <xsl:copy>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>
  <!-- questo modello copia tutti gli elementi di BODY insieme ad i
loro attributi -->
    <xsl:template match="body//*">
    <xsl:copy>
      <xsl:copy-of select="@*"/>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>

    <!-- questo modello aggiunge un'attributo ID all'elemento NAME -->
  <xsl:template match="name" priority="1">
  <xsl:copy>
  <xsl:copy-of select="@*"/>
    <xsl:attribute                                  name="id"><xsl:value-of
select="./ancestor::div1/@id"/>-<xsl:value-of         select="generate-
id()"/></xsl:attribute>
  <xsl:apply-templates/>
  </xsl:copy>
  </xsl:template>
</xsl:stylesheet>
```

Appendice 21

FOGLIO DI STILE PER LA CREAZIONE DEGLI INDICI DI NAVIGAZIONE (VERSIONE CON PARAMETRI)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output encoding="ISO-8859-1"/>
  <xsl:param name="pp">Gerlando Boccadoro</xsl:param>
  <xsl:template match="/">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="text">
  NOME: <b><xsl:value-of select="$pp"/></b><br/>
    <xsl:for-each select="//name[@key=$pp]">
         <xsl:sort select="@key"/>
      <a href="baltico.htm#{@id}">capitolo <xsl:value-of
select="./ancestor::div1/@id"/></a><br/>
      </xsl:for-each>
    occorrenze: <xsl:value-of select="count(//name[@key=$pp])"/>

  </xsl:template>
</xsl:stylesheet>
```

Appendice 22

FOGLIO DI STILE PER LA CREAZIONE DEGLI INDICI DI NAVIGAZIONE (VERSIONE SENZA PARAMETRI)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output encoding="ISO-8859-1"/>
  <xsl:template match="/">
    <xsl:apply-templates mode="nome"/>
  </xsl:template>
  <xsl:template match="text" mode="nome">

    <xsl:for-each select="//name">
           <xsl:sort select="@key"/>
      <a href="baltico.htm#{@id}"><xsl:value-of select="@key"/> -
capitolo <xsl:value-of select="./ancestor::div1/@id"/></a><br/>
      </xsl:for-each>


  </xsl:template>
</xsl:stylesheet>
```

BIBLIOGRAFIA

CIOTTI FABIO, *Cosa è la codifica informatica dei testi*, in AA.VV., *Atti del Convegno Umanesimo & Informatica* (Trento 24-25 maggio 1996), a cura di Daniela Gruber e Patrick Pauletto.

CIOTTI FABIO, RONCAGLIA GINO, *Il mondo digitale. Introduzione ai nuovi media*, Roma-Bari, Editori Laterza, 2001[4].

DENNIS ANITA, *Guida all'uso. PDF con Adobe Acrobat 5*, Milano, Mondadori Informatica, 2002 (ed. orig. *Real World PDF with Adobe Acrobat 5*, 2002).

FORTE FRANCO, SOSIO SILVIO, *Pixel da leggere*, in *PC World Italia* n.6, giugno 2003, pp.168-175.

GIGLIOZZI GIUSEPPE, *Il testo e il computer. Manuale di informatica per studi letterari*, Milano, Bruno Mondadori, 1997.

LAVARONE FRANCO (a cura di), *DRM, i sistemi anti-pirateria*, in *Le grandi guide di PC World Italia* n.3, maggio 2002, pp.36-37.

LEANDRI PAOLO (a cura di), *Il libro del futuro, l'e-book e le nuove frontiere dell'editoria alla fiera di Torino*, in *Cultura & Libri* n.134, maggio/giugno 2001.

LONGO BRUNELLA, *La nuova editoria*, Milano, Editrice Bibliografica, 2001.

LORENZINI LUCREZIA, *"Microstorie frantumate" in Baltico di Matteo Collura*, in AA.VV., *Atti del VII Convegno Internazionale S.I.L.F.I. Genesi, architetture e forme testuali,* (Roma1-5 ottobre 2002), pp.223-232 (in corso di stampa).

METITIERI FABIO, RIDI RICCARDO, *Biblioteche in Rete. Istruzioni per l'uso*, Roma-Bari, Editori Laterza, 2002.

NEGROPONTE NICHOLAS, *Essere Digitali*, trad. it. di Franco e Giuliana Filippazzi, Milano, Sperling & Kupfer Editori S.p.A. 1999[2], (ed. orig. *Being Digital*, Alfred A. Knopf, Inc., 1995).

SALA VIRGINIO B., *e-book*, Milano, Apogeo, 2001.

SALARELLI ALBERTO, TAMMARO ANNA MARIA, *La biblioteca digitale*, Milano, Editrice Bibliografica, 2000.

SHEPHERD DEVAN, *XML. Guida completa*, trad. it. staff Apogeo, Milano, Apogeo, 2002, (ed. orig. *Teach yourself XML in 21 Days second edition*, Sams, 2001).

STANEK WILLIAM R., *XML*, Milano, Mondadori Informatica, 2002, (ed. orig. *XML Pocket Consultant*, Microsoft Corporation, 2002).

TITTIEL ED, MIKULA NORBERT, CHANDAK RAMESH, *XML for Dummies*, Milano, Apogeo, 1998, (ed. orig. *XML for Dummies*, IDG Books Worldwide, Inc., 1998).

VAN OTEGEM MICHIEL, *XSLT. Guida completa*, trad. it. di William Balduzzo, Michele Pacifico, Milano, Apogeo, 2002, (ed. orig. *Teach yourself XSLT in 21 Days*, Sams, 2002).

WEBLIOGRAFIA

Tutti gli Url sono stati verificati a maggio del 2004

AA.VV., *Quale futuro per l'e-book?*, e-book disponibile on-line nel sito *Apogeonline*
<http://www.apogeonline.com/ebook/2002/90035/pdf/EbookQualeFuturo.pdf>
2002.

ADDOBBATI ANDREA, *Incartati nella rete. Inchiesta sui consumi di carta nell'epoca di Internet*, in *Athenet On Line*, *notizie e approfondimenti dall'Università di Pisa* n.4, settembre 2001
<http://www.unipi.it/athenet/04/articoli/0004Carta_box3A.html>
19 settembre 2002.

ADOBE SYSTEMS INCORPORATED, *DIECI ANNI DI ADOBE PDF: UN SUCCESSO A LIVELLO MONDIALE*, in *Adobe.it Pressroom*
<www.adobe.it/aboutadobe/pressroom/ pr/jul2003/AdobeAcrobat_10.pdf>
15 luglio 2003.

ADOBE SYSTEMS INCORPORATED, *How to Create Adobe PDF Files for eBooks*, e-book disponibile on-line nel sito *Adobe*
<http://www.adobe.com/products/acrdis/createbooks.html>
07 ottobre 2000.

AUGIAS CORRADO, *E-book, istruzioni per l'uso. Intervista a Roger Chartier sugli orizzonti del libro elettronico*, in *Repubblica.it*
<http://www.repubblica.it/online/cultura_scienze/fran/fran/fran.html>
18 ottobre 2000.

BERNERS-LEE TIM, *Information Management: A Proposal*, in *W3 archive*
<http://www.w3.org/History/1989/proposal.html>.

BIANCHI DAVIDE, *Breve introduzione al formato* RTF, in *Articoli, documenti, faq e tutto quello che fa programmazione*
<http://www.soft-land.org/documenti/rtf.html>
21 dicembre 2000.

BOCCUZZI MASSIMO, *Verso il non libro*, in *Ebook Italia Forum*, *Italianistica Online*
<http://www.italianisticaonline.it/e-book/forum_2002/relazioni/index.htm>
30 settembre 2002.

BOS BERT, *XML in 10 punti*, trad. it. Emiliano Tellina, in *Latoserver.it*
<http://www.latoserver.it/XML/in10punti/>
(ed. orig. *XML in 10 points*, <http://www.w3.org/XML/1999/XML-in-10-points>).

BRIAN RAMPERSAD, *Alan Kay's Dynabook*, in *The Design of Everyday Things. Computer Software: Mapping our minds to create a brain prosthesis*.
<http://www.sheridanc.on.ca/~randy/design.dir/webctsoftware/alan_kay.htm>
aprile 1997.

BURNARD LOU, SPERBERG-MCQUEEN C. M., *TEI Lite: introduzione alla codifica dei testi*, trad. it. Fabio Ciotti, Guendalina Demontis, Giuseppe Gigliozzi, Massimo Guerrieri, Andrea Loreti in *TEI Website*
<http://www.tei-c.org/Lite/teiu5_it.htm>
(ed. orig. *TEI U5: Encoding for Interchange: an introduction to the TEI*, <http://www.tei-c.org/Lite/teiu5_en.tei >)
gennaio 1998.

CALVO MARCO, CIOTTI FABIO, RONCAGLIA GINO, ZELA MARCO, *Come funziona World Wide Web*, in *Internet 2000* (versione online)
<http://www.laterza.it/internet/leggi/internet2000/online/testo/30_testo.htm>.

CERAVOLO PAOLO, Tesi di laurea *Dai graffiti al web: come cambia la struttura referenziale nell'età della rete*
<http://gircse.marginalia.it/~ceravolo/intestazione.php>
02 ottobre 2001.

CHARTIER ROGER, *Lettori e letture nell'era della testualità elettronica* in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=8>.

CIARFAGLIA PIERVITO, MIANI MATTIA, PELLEGRINI SWAMI, UBOLDI FRANCESCO, VALLAURI UGO, *e-book: quando il libro diventa elettronico*, ricerca realizzata nell'ambito del corso di Informatica Generale del prof. Paolo Ciancarini Università di Bologna 2001
<http://www.geocities.com/katanankes/ebook/ebook.pdf>***
gennaio 2001.

CIOTTI FABIO, *Breve introduzione alla Text Encoding Initiative*, in *Biblioteca Italiana*
<http://www.bibliotecaitaliana.it/tei_intro.asp>
01 dicembre 2003.

CIOTTI FABIO, *Cosa è la codifica informatica dei testi*, in AA.VV., *Atti del Convegno Umanesimo & Informatica* (Trento 24-25 maggio 1996), a cura di Daniela Gruber e Patrick Pauletto.
<http://circe.lett.unitn.it/circe/html/attivita/uman/ciotti.pdf>.

CIOTTI FABIO, *Documento BibIt-1 Manuale di riferimento per la codifica testuale: livello 1 Revisione 3* in *Biblioteca Italiana*
<http://www.bibliotecaitaliana.it/dtd/Bibit-Manuale_codifica_L1.pdf>
11 marzo 2003.

CIOTTI FABIO, *TEI in digital library: an italian case*, intervento al *TEI Members Meeting*, Pisa 17-18 Nov 2001, in TEI Consortium Website
<http://www.tei-c.org.uk/Members/2001-Pisa/Talks/ciotti.htm>.

CIOTTI FABIO, T*esti elettronici e banche dati testuali. Problemi teorici e tecnologie*, in *Schede umanistiche*, n.s., 2, 1995, pp. 147-178, disponibile on-line in
<http://crilet.let.uniroma1.it/ciotti/pubblicazione/tes_el.htm>.

CUZZOCREA ANNALISA, *Stephen King ferma il suo e-book*, in *Repubblica.it*
<http://www.repubblica.it/online/tecnologie_internet/king/plant/plant.html>
29 novembre 2000.

DE LAZZARO PAOLO, *XML L'ipertesto sbarca sul Web*
<http://www.geocities.com/SiliconValley/Network/7776/xml.htm>
Articolo non più disponibile in rete (21 ottobre 2003).

DELGADO-KLOOS CARLOS, SÁNCHEZ-FERNÁNDEZ LUIS, *XML: The ASCII of the 21st Century* in *Upgrade The European Online Magazine for the IT Professional* Vol. III, No. 4, August 2002
<http://www.upgrade-cepis.org/issues/2002/4/upgrade-vIII-4.pdf>
04 agosto 2002.

ECO UMBERTO, *Autori e autorità*, in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=11>.

EPSTEIN JASON, *Leggere: il futuro digitale*, in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=13>
Trad. it. a cura di Delfina Vezzoli, l'articolo era già apparso nel The New York Review of Books n° 11 vol. 48 del 5 luglio del 2001 con il titolo *Reading: The Digital Future.*

EVOLUTIONBOOK, *Acrobat eBook Reader* in *Evolutionbook*
<http://www.evolutionbook.com/Mod_02.php?data_dir=Tech&data_file=Articolo_SW&ID=9 >
29 marzo 2002.

EVOLUTIONBOOK, *Aumentano ancora le vendite di eBook*, in *Evolutionbook*
<http://www.evolutionbook.com/Mod_02.php?data_dir=News&data_file=Articolo&data=2003-09-24&numero=02>
24 settembre 2003.

EVOLUTIONBOOK, *La Pianta ha dato ottimi frutti*, in *Evolutionbook*
<http://www.evolutionbook.com/Mod_02.php?data_dir=News&data_file=Articolo&data=2001-02-19&numero=04>
19 febbraio 2001.

EVOLUTIONBOOK, *SPECIALE TECNOLOGIA ClearType*, in *Evolutionbook*
<http://www.evolutionbook.com/Mod_02.php?data_dir=Tech&data_file=Speciale_tecnologia&data=2001-06-01&numero=00>
01 giugno 2001.

GSMBOX S.P.A., "Tablet PC: *una rivoluzione* tecnologica, in *GSMBOX.IT mobile news*
<http://it.gsmbox.com/news/mobile_news/all/96723.gsmbox>
09 luglio 2003.

HARNAD STEVAN, *Lettura e scrittura celeste per ricercatori: Un'anomalia post-Gutenberg e la sua soluzione* in *Text-e* convegno virtuale "*Schermi e reti, verso una trasformazione della scrittura?*"
<http://www.text-e.org/conf/index.cfm?fa=printable&ConfText_ID=7>.

KOK HARM, *Comparison of XSL-FO Render software* in *diderot track*
<http://www.diderottrack.nl/nl.articles.xslfo.html>
2002.

LOVETT CHRIS, *Come codificare i dati XML*, in *MSDN Library*
<http://www.microsoft.com/italy/msdn/library/default.asp?url=/italy/msdn/library/xmlsoap/xmlencodings

.asp?frame=true>
marzo 2000.

MAH CAROLE E., FLANDERS JULIA H., *The TEI Header:A Tutorial with Examples* in *Brown University Women Writers Project*
<http://www.wwp.brown.edu/encoding/training/teiheader/teiHeader.html >
29 ottobre 1998

MASSACRA LAURA, *Come sarà il cybermondo del 2001. Bill Gates. Bilanci e previsioni sul futuro di Internet*, in *Mediamente Biblioteca Digitale*
<http://www.mediamente.rai.it/biblioteca/prov/001229gates.asp>.

MELLINI MANUELA, *IL MAGICO MONDO DELL'OEB in Words on-line*
<http://www.wordson-line.it/oeb/articolo2.htm>
2002.

MICROSOFT CORPORATION, *Microsoft Reader – Support for PC*, in *Microsoft.com*
<http://www.microsoft.com/reader/it/support/faq/general.asp>.

MORGOGLIONE CLAUDIA, *Per il libro in Rete ecco i lettori elettronici* in *Repubblica.it*
<www.repubblica.it/online/cultura_scienze/fieralibro/ebook/ebook.html>
11 maggio 2000.

NEGROPONTE NICHOLAS, durante un'intervista realizzata dallo staff di Mediamente *La rivoluzione digitale*, in *Mediamente Biblioteca digitale*
<http://www.mediamente.rai.it/home/bibliote/intervis/n/negrop02.htm>
Venezia - Ca' Foscari, 3 giugno 1995).

OPEN EBOOK FORUM, *Open eBook Publication Structure 1.2*l, in *Open eBook Forum*
<http://www.openebook.org/oebps/oebps1.2/index.htm>
27 agosto 2002.

ORFEI FABIO, *Il futuro del libro e l'avvento degli eBook*, in *Ebook Italia Forum*, *Italianistica Online*
< http://www.italianisticaonline.it/e-book/forum_2002/relazioni/index.htm>
30 settembre 2002.

PASCUZZI DOMENICO, *E-book, il libro diventa digitale* in *ITportal*,
<http://www.itportal.it/special/multimedia/e-book/default.asp>
04 giugno 2001.

PORRO STEFANO, *Il libro elettronico? Bocciato*, in *Clarence Cultura e Spettacolo*
<http://www.clarence.com/contents/culturaspettacolo/societamenti/speciali/000712ebook/001.html>
13 luglio 2000.

PROJECT GUTENBERG, *What is PG? HISTORY AND PHILOSOPHY OF PROJECT GUTENBERG*, in *Project Gutenberg official website*
<http://promo.net/pg/history.html>
13 maggio 2002.

REGALZI GIUSEPPE, *Vino vecchio in otri nuovi. La letteratura scientifica nell'era dell'eBook*, in *Ebook Italia Forum*, *Italianistica Online*
<http://www.italianisticaonline.it/e-book/forum_2002/relazioni/index.htm>
30 settembre 2002.

ROMAGNOLO SALVATORE, *Palm inventa l'e-book da dare in prestito*, in *Romagnolo.com*
<www.romagnolo.com/e-book/palm-ebook.htm>
06 dicembre 2002.

ROMAGNOLO SALVATORE, *Prestare i libri elettronici: adesso si può*, in *Romagnolo.com*
<www.romagnolo.com/e-book/adobe.htm>
25 giugno 2002.

ROMAGNOLO SALVATORE, *Quale futuro per la lettura?*, in *Romagnolo.com*
<http://www.romagnolo.it/frontiere/lettura.htm>
2002.

RONCAGLIA GINO, *Libri elettronici: problemi e prospettive*, a cura di Anna Galluzzi, in *Associazione italiana biblioteche. BollettinoAIB*
<http://www.aib.it/aib/boll/2001/01-4-409.htm>
25 febbraio 2002.

SGML USERS' GROUP,*A Brief History of the Development of SGML*, in *The CoverPages*
<http://www.oasis-open.org/cover/sgmlhist0.html>
11 giugno 1990.

STAGLIANÒ RICCARDO, *Arrivano i manuali digitali, economici e sempre aggiornati*, in Repubblica.it
<http://www.repubblica.it/online/tecnologie_internet/universita/libritesto/libritesto.html>
16 agosto 2000.

STAGLIANO RICCARDO, *King: "Perché ho smesso di pubblicare online"*, in *Repubblica.it*
<http://www.repubblica.it/online/tecnologie_internet/king/spiega/spiega.html>
13 dicembre 2000.

TAVOSANIS MIRKO, *La Biblioteca diventa virtuale. Il Cibit mette in rete la cultura italiana*, in *Athenet On Line, notizie e approfondimenti dall'Università di Pisa N°2 settembre* 2000
<http://www.unipi.it/athenet/02/articoli/0002tavosanis_bibliotecaA.html>
19 settembre 2002.

TERMINFORMATICI, *Grafica Vettoriale e Grafica Bitmap*, in *terminiformatici*
<http://www.termininformatici.com/grafica/leggi/quattro.asp>
2002.

UNICODE CONSORTIUM, *The Unicode® Standard: A Technical Introduction*, in *Unicode Home Page*
<http://www.unicode.org/standard/principles.html>
23 aprile 2003

UTILI FABIO, *I caratteri di testo e internet*, in *Home page of Fabio Utili*
<http://digilander.libero.it/fabioutili/tavolASCII.html>
11 luglio 2003.

WILSON RUTH, *The Problem of Defining Electronic Books*, documento sviluppato nell'ambito del progetto *EBONI*: <http://eboni.cdlr.strath.ac.uk/documents/definition.html>
14 novembre 2000.

WORLD WIDE WEB CONSORTIUM, *Extensible Markup Language (XML) 1.0*, traduzione ufficiale delle specifiche XML 1.0 effettuata da Andrea Marchetti
<http://www.w3c.it/traduzioni/xml-19980210/REC-xml-19980210-it.html>.

ZAMPARELLI MARCO, QUAGLIARIELLO SUSANNA, *La Carta Elettronica*, in *Sottoachitocca.it Computer e Tecnologia*
<http://www.sottoachitocca.it/computertecnologia/cartaelettronica.htm>.

ZONCA EMANUELA, *Cortocircuito tra passato e futuro: il ritorno dello scrittore-artigiano*, in *Ebook Italia Forum*, *Italianistica Online*
<http://www.italianisticaonline.it/e-book/forum_2002/relazioni/index.htm>
30 settembre 2002.

INDICE

Premessa *1*

PARTE PRIMA

I.1.1. Immaterialità **2**
I.1.2. Trasmissibilità **3**
I.1.3. Conservazione **3**
I.1.4. Riproducibilità **3**
I.1.5. Duttilità **4**
I.2. CODIFICA INFORMATICA DEI TESTI *6*
I.3. LA CODIFICA DEI CARATTERI *8*
I.3.1. Le tavole dei caratteri **8**
I.3.2. Un po' di storia **9**
I.3.3. Prospettive future **11**
Tavola dei caratteri ASCII **14**
I.4. I LINGUAGGI DI CODIFICA *15*
I.4.1. I programmi di text processing WYSIWYG **16**
I.4.2. Codifiche procedurali e codifiche dichiarative **17**
I.4.3. Linguaggi di codifica procedurali **17**
I.4.4. Linguaggi di codifica dichiarativi **19**
I.4.5. SGML **20**
I.4.6. HTML **26**
I.5. XML *30*
I.5.1. XSL **35**
I.5.2. XSL-FO **37**
I.5.3. XPath **37**
I.5.4. XSLT **38**
I.5.5. CSS **42**
I.5.6. Tecnologie correlate a XML **42**

PARTE SECONDA

II.1. E-BOOK *45*
II.1.1. E-book e codifica elettronica **47**

II.1.2. E-text 48
II.2. STORIA DELL'E-BOOK 49
II.2.1. Alan Kay 49
II.2.2. Project Gutenberg 49
II.2.3. The Random House Electronic Thesaurus 51
II.2.4. Franklin Electronic Publishers 51
II.2.5. Il Sony Data Discman 51
II.2.6. Rocket e Softbook 52
II.2.7. Il 2000 anno zero 53
II.2.8. Il caso King 54
II.2.9. La scesa in campo dei giganti dell'editoria 56
II.2.10. Le grandi software house 56
II.2.11. Il declino 57
II.3. FORMATI E-BOOK 61
II.3.1. PDF 61
II.3.2. OEBPS 65
II.3.3. Il formato LIT 70
II.4. HARDWARE PER IL LIBRO ELETTRONICO 76
II.4.1. Palmari 76
II.4.2. Palm OS 77
II.4.3. Pocket PC 79
II.4.4. Considerazioni sulla piattaforma palmare 80
II.4.5. My Friend 81
II.4.6. Tablet PC 82
II.4.7. Prospettive future 85
II.5. STAMPA SU RICHIESTA 88
II.6. CRITTOGRAFIA E FIRME DIGITALI 93
II.6.1. La guerra degli Mp3 93
II.6.2. DRM e crittografia 97
II.6.3. Firma digitale 99
II.7. VANTAGGI E LIMITI DEL LIBRO ELETTRONICO 103
II.7.1. Immaterialità 103
II.7.2. Trasmissibilità 104
II.7.3. Conservazione 104
II.7.4. Riproducibilità 106

II.7.5. Duttilità 107
II.8. COME ABBIAMO LAVORATO*.....*109

PARTE TERZA

III.1. INTRODUZIONE*.....*114
III.1.2. Scelte 114
III.1.2. Sul concetto di biblioteca digitale.....115
III.2. TEI*.....*119
III.2.1. Lo schema di codifica TEI 120
III. 3. ESPERIENZE DI CODIFICA IN ITALIA*.....*125
III.3.1. TIL 125
III.3.2. GriseldaOnLine 126
III.3.3. CIBIT.....126
III.4. DIGITALIZZAZIONE* *128
III.5. CODIFICA*.....*130
III.5.1. I livelli di codifica.....130
III.5.2. Il set di caratteri.....132
III.5.3. Gli strumenti per la codifica 134
III.5.4. Struttura dei documenti TEI 135
III.6. LA CODIFICA DI BALTICO*.....*138
III.6.1. <front> 138
III.7. LA CODIFICA DEL CORPO DEL DOCUMENTO (<body>)* *144
III.7.1. <div1> 145
III.7.2. <head> 148
III. 7.3. Paragrafi.....149
III.7.4. Salti di pagina.....149
III.7.5. Elementi intralineari 150
III.7.6. Nomi ed espressioni referenziali.....151
III.7.7. Nomi propri.....151
III.7.8. Nomi di luogo ed altre espressioni referenziali 152
III.7.9. Riferimenti temporali.....153
III.7.10. Discorso diretto e citazioni.....153
III.8. I MATERIALI CHE SEGUONO IL TESTO (<back>)*.....*160
III.9. IL FRONTESPIZIO ELETTRONICO*.....*162
III.9.1. Descrizione del file 163

III.9.2. Dichiarazione del titolo 163
III.9.3. Dichiarazione dell'edizione 165
III.9.4. Dichiarazione della dimensione 166
III.9.5. Dichiarazione della pubblicazione 166
III.9.6. Dichiarazione della fonte 170
III.9.7. Descrizione della codifica 172
III.9.8. Descrizione del profilo 174
III.9.9. Descrizione della revisione 176
III.10. REVISIONE DELLA CODIFICA 178
III.11. LA PRODUZIONE DEGLI OUTPUT 182
III.11.1. L'output HTML dell'Header 182
Appendice 1 194
Appendice 2 198
Appendice 3 200
III.12. OUTPUT HTML 202
Appendice 4 211
III.13. ACCESSIBILITÀ 216
Appendice 5 222
III.14. XML DATA ISLAND 227
Appendice 6 232
Appendice 7 233
III.15. OEBPS e LIT 235
Appendice 8 244
Appendice 9 247
Appendice 10 249
Appendice 11 252
Appendice 12 253
Appendice 13 257
III.16. PDF E PRINT ON DEMAND 258
Appendice 14 276
Appendice 15 282
Appendice 16 288
Appendice 17 335
III.17. ANALISI ED INTERPRETAZIONE 340
Appendice 18 350

Appendice 19.....**351**
Appendice 20.....**355**
Appendice 21.....**356**
Appendice 22.....**357**
BIBLIOGRAFIA.....***358***
WEBLIOGRAFIA.....***360***
INDICE.....***366***