Indice generale

0 Introduzione

Gli ebook sono un tipo di libro digitale, realizzato in modo da poter essere letto tramite un computer, un ebook reader (hardware creato appositamente per la lettura di ebook), tablet, cellulare o altri tipi di supporti fisici.
Un qualunque documento di testo non può essere paragonato ad un ebook, in quanto quest'ultimo tenta di replicare l'aspetto di un libro cartaceo. Il software di lettura utilizzato, di solito, tenta di emulare le azioni compiute dall'essere umano su un libro: la funzione più banale è lo sfogliare le pagine, che in alcuni ebook reader si limita all'aggiunta di alcuni pulsanti, mentre tablet più avanzati simulano il gesto, per rendere l'esperienza di lettura più vicina a quella di un libro vero.
Altri software o hardware permettono altre azioni utili all'utente come la sottolineatura del testo, l'utilizzo di segnalibri e l'inserimento di commenti personali.
Un ebook è preferibile a un libro in quanto un hardware per la lettura di ebook occupa, di solito, meno spazio rispetto ad un libro, è quindi meno pesante e più facilmente trasportabile (una persona che possiede un ebook reader può portare in viaggio un'intera biblioteca). L'ebook è meno costoso in termini di produzione, e costa meno di un libro cartaceo (sebbene vi siano ancora molte tasse sopra che non gli permettono di decollare come prodotto super-economico).
Si presta più facilmente ad aggiornamenti rispetto al rivale cartaceo: ad esempio un'edizione di un libro può presentare errori e, se tali da bloccare la normale distribuzione, è molto costoso e faticoso ritirare le copie cartacee messe in vendita: è invece possibile mettere online una versione aggiornata del prodotto, magari permettendo all'utente che ha comprato una vecchia copia dell'ebook di ottenere la nuova semplicemente presentando il codice d'acquisto della versione precedente, in modo pratico e veloce.
Il formato EPUB non è l'unico formato in cui può presentarsi un ebook: questo infatti è lo standard più diffuso, ma esistono anche il formato LIT, formato nativo Microsoft, che non viene più utilizzato; PDB, formato per Palm OS; Mobipocket (mobi).
Vengono usati, quasi impropriamente dal momento che non erano pensati per quest'utilizzo, formati come ASCII, CHM (appartenente a Microsoft, ormai non più utilizzato), HTML, Microsoft Word (.doc), Open Office Writer (.odt), PostScript, che è il linguaggio generalmente pensato per controllare le stampanti, RTF (di proprietà della

Microsoft, ormai obsoleto), TeX, utilizzato principalmente per la stesura di testi scientifici e matematici, e PDF, probabilmente il formato più utilizzato tra quelli impropriamente diffuso.

La maggior parte dei creatori di ebook in formato EPUB comunque fa utilizzo di programmi che non richiedono la conoscenza del formato o le competenze tecniche necessarie alla creazione di questo tipo di ebook; assomigliano agli elaboratori di testi come Word e Open Office, ma restituiscono in output ebook in formato EPUB invece che documenti di testo in formato .doc o .odt[1]. Dal momento che un ebook in formato EPUB altro non è che un archivio ZIP contenente documenti XML, (X)HTML e immagini, esistono altri metodi per la creazione di ebook in questo formato.

Dato che esistono molti testi codificati che utilizzano il formato TEI XML per la pubblicazione di testi nel web, e questo formato risulta così adattabile da permettere la creazione non solo di documenti HTML basati sul linguaggio XML, ma anche documenti in formato PDF e soprattutto EPUB, diviene di interesse fondamentale per coloro che si occupano di questi progetti, o che vogliono cimentarsi con essi, prendere in considerazione l'idea di utilizzare gli strumenti TEI per creare ebook in formato EPUB prendendo come punto di partenza i documenti TEI XML.

I fogli di stile TEI per la trasformazione da XML a EPUB producono un risultato soddisfacente: tuttavia, affinché sia garantito un buon risultato, è necessario che l'utente che li utilizza intervenga direttamente su di essi o su i parametri definiti all'interno di essi, e ne modifichi il risultato in base alle proprie necessità. Per effettuare dei cambiamenti su questi fogli di stile, è indispensabile conoscere il formato EPUB, per operare cambiamenti con cognizione di causa, e i fogli di stile stessi, che presentano della documentazione sul sito del consorzio TEI, ma sono privi di istruzioni specifiche sul loro funzionamento.

Per questo motivo nel primo capitolo prendo in esame la struttura e composizione del formato EPUB, e nel secondo capitolo descrivo il modo di utilizzo dei fogli di stile TEI per questa trasformazione (facendo uso del programma Oxygen, che permette di compiere questo procedimento in maniera automatica), e i vari documenti che ne fanno parte (quali sono questi fogli di stile e per cosa vengono utilizzati).

Nel terzo capitolo espongo vari esempi di personalizzazione dell'output che ho attuato

1 Alcuni esempi di programmi di questo tipo sono InDesign; Sigil, che permette di creare e modificare ebook in formato EPUB; Calibre, programma che prende in input documenti di testo in vari formati per trasformarli in ebook in formato EPUB tramite pochi passaggi da parte dell'utente; Feedbooks, sito dove è possibile inserire il proprio testo e specificare le caratteristiche dell'ebook desiderato.

modificando i fogli di stile TEI per la trasformazione da XML a EPUB.

1 Il formato EPUB[2]

Tra i vari formati disponibili per creare ebook vi è il formato EPUB. È un archivio contenente diversi file: di fatto, se ne viene cambiata l'estensione, da .epub a .zip, si mostra realmente come un archivio ZIP, aprendo il quale è possibile estrarne i file, modificarli o cancellarli.
Conoscerlo è importante per poter gestire con cognizione di causa i fogli di stile TEI, con la quale verrà creato un ebook in questo formato in maniera automatica.
Un documento EPUB è composto da un certo numero di file: alcuni di questi sono obbligatori, alcuni di essi possono venire omessi senza ripercussioni evidenti sulla visualizzazione finale (che dipende anche dal tipo di ebook reader utilizzato); alcuni di essi invece necessitano non solo di essere presenti nell'archivio per far sì che possa essere visualizzato, ma devono anche venire inseriti rispettando un determinato ordine.

1.1 La struttura di un epub

I documenti che obbligatoriamente devono trovarsi dentro un ebook in formato EPUB sono `mimetype`, `container.xml` all'interno di una cartella META-INF; altri due, `content.opf` e `toc.ncx`, sono strettamente necessari, ma la loro posizione può variare (è possibile trovarli dentro cartelle diverse, o nella stessa cartella, o anche nella root del'ebook). Gli altri documenti che possiamo trovare dentro un ebook in formato EPUB sono opzionali e a discrezione completa del creatore dell'ebook. Verranno quindi descritti adesso i singoli file nel dettaglio.

1.1.1 mimetype

Il contenuto di questo documento è standard, e non deve essere mai modificato:

```
application/epub+zip
```

2 Riferito al formato EPUB 2.0.1, definito dal International Digital Publishing Forum: http://idpf.org/epub/201.

Si tratta quindi di un documento di testo non compresso, e affinché l'ebook funzioni correttamente, è necessario che venga inserito nell'archivio come primo elemento.

1.1.2 META-INF

È una cartella che si trova sempre al livello base dell'archivio EPUB. Il nome di questa cartella non deve mai essere modificato. La cartella contiene un solo documento XML, `container.xml`.

1.1.3 container.xml

Anche il nome di questo documento XML non deve essere mai modificato, e deve sempre essere posizionato dentro al cartella META-INF. Il suo contenuto non varia molto da ebook a ebook; di solito il codice è questo o uno molto simile:

```
<container xmlns="urn:oasis:names:tc:opendocument:xmlns:container"
version="1.0">
    <rootfiles>
        <rootfile full-path="OPS/content.opf" media-
type="application/oebps-package+xml"/>
    </rootfiles>
</container>
```

Questo documento XML serve a indicare la posizione del documento `content.opf`, che è basilare per il funzionamento di tutto l'ebook. A seconda di come è chiamata la cartella che contiene `content.opf`, varia la riga di codice:

```
<rootfile full-path="OPS/content.opf" media-type="application/oebps-
package+xml"/>
```

In questo esempio `content.opf` è contenuto in una cartella chiamata “OPS”, che è la cartella standard dei fogli di stile TEI per contenere `content.opf`; altrimenti è uso comune chiamarla “OEBPS”, che sta per “Open eBook Publication Structure”.
La cartella che lo ospita può chiamarsi in qualsiasi modo: è importante però che sia

indicato correttamente, all'interno del documento XML, dove si trova `content.opf`.

1.1.4 content.opf

L'estensione di questo documento è un acronimo di "Opening Packaging Format": è un documento in formato XML che dichiara i contenuti che sono presenti all'interno dell'ebook. È necessario specificare all'interno di esso tutte le componenti dell'ebook che dovranno essere visualizzate, altrimenti non appariranno nell'ebook reader utilizzato, qualunque esso sia (anche se saranno fisicamente presenti). Le informazioni che fornisce sono importantissime, perché elenca gli elementi all'interno dell'ebook e in quale ordine devono essere visualizzati (e letti, in sostanza, dall'ebook reader). È divisibile in quattro sezioni, più una facoltativa:

<sezione di dichiarazione (1)>
  <sezione (2) metadata />
  <sezione (3) manifest />
  <sezione (4) spine />
  <sezione (facoltativa) guide />
</sezione di dichiarazione>

La prima sezione è la dichiarazione XML:

```
<?xml version="1.0" encoding="utf-8"?>
<package xmlns="http://www.idpf.org/2007/opf" unique-
identifier="dcidid" version="2.0">
```

Nell'elemento <package> l'identificatore univoco *dcidid* si ricollega a un elemento appartenente ai metadati, <dc:identifier>, e se non vi è corrispondenza tra i due l'ebook non verrà visualizzato correttamente. I metadati servono per fornire all'ebook reader informazioni sull'ebook stesso, come ad esempio il titolo e l'autore, e determinano anche in questo modo la sua visualizzazione.

```
<metadata xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <dc:title
```

```
xmlns:dc="http://purl.org/dc/elements/1.1/">Title</dc:title>
    <dc:language xmlns:dc="http://purl.org/dc/elements/1.1/"
xsi:type="dcterms:RFC3066">en</dc:language>
    <dc:identifier xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:opf="http://www.idpf.org/2007/opf" id="dcidid"
opf:scheme="URI">http://www.example.com/TEIEPUB/20120914194825</dc:ide
ntifier>
    <dc:description xmlns:dc="http://purl.org/dc/elements/1.1/">Insert
description here</dc:description>
    <dc:creator
xmlns:dc="http://purl.org/dc/elements/1.1/">Anonymous/Unknown</dc:crea
tor>
    <dc:publisher xmlns:dc="http://purl.org/dc/elements/1.1/"/>
    <dc:date xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:opf="http://www.idpf.org/2007/opf" opf:event="epub-publication"
xsi:type="dcterms:W3CDTF">2012-09-14</dc:date>
    <dc:rights xmlns:dc="http://purl.org/dc/elements/1.1/">Creative
Commons Attribution</dc:rights>
</metadata>
```

Sebbene non sia importante l'ordine in cui vengono inseriti, tre elementi in questo elenco sono obbligatori: <title>, <language> e <identifier>. Devono essere indicati forzatamente il titolo e il linguaggio per permettere una corretta visualizzazione dell'ebook (in particolare, per il linguaggio viene utilizzata la sigla identificativa, quindi sarà indicata come “en” e non “english”, nel caso della lingua inglese). L'identificativo deve necessariamente essere univoco: solitamente, per esserne sicuri, chi scrive l'ebook utilizza il codice ISBN del libro che sta creando.

Nel nostro esempio viene utilizzato un indirizzo HTML: essendo il codice soprastante estratto da un ebook creato utilizzando la trasformazione EPUB del programma Oxygen basata sui fogli di stile TEI, è evidente che l'uso di un codice a caso non è opportuno. Quindi la IANA, “Internet Assigned Numbers Authority”, mantiene una serie di domini chiamati “EXAMPLE.COM” e “EXAMPLE.ORG”, con il solo scopo della documentazione, non disponibili per la registrazione, e perciò sempre liberi. Se l'identificatore non viene quindi segnalato nell'input XML, verrà assegnato all'ebook in output un identificatore “esempio”. Altri elementi presenti sono facoltativi:

- <publisher>: l'editore;

- <rights>: tutte le informazioni sul copyright, nel caso dello dello standard TEI per la creazione di ebook viene automaticamente utilizzata la dicitura “Creative Commons Attribution”;
- <subject>: l'argomento del libro, che viene abbreviato in semplici parole chiave;
- <description>: la descrizione del contenuto dell'ebook (non abbreviata);
- <date>: la trasformazione EPUB applicata dal programma Oxygen applica automaticamente al giorno in cui è stato creato l'epub;
- <type>: la categoria alla quale appartiene l'epub (fantascienza, fantasy, biografia, …);
- <format>: informazioni tecniche supplementari riguardo l'ebook, ad esempio se è necessario un particolare software o hardware per permetterne la lettura;
- <source>: origine del contenuto, viene indicata nel caso in cui esista una versione dello scritto precedente alla creazione dell'ebook, sia su carta sia sul web: può essere costituito dall'ISBN, dall'indirizzo web della locazione originale, oppure descrittiva;
- <relation>: indica se vi sono scritti collegati a quello preso in esame: come sopra, gli scritti possono essere indicati tramite ISBN, un indirizzo web o da un testo descrittivo;
- <coverage>: indica a chi è destinato l'ebook: può essere una certa fascia di età, una famiglia di appartenenti a una certa cultura, i parlanti di una certa lingua, eccetera;
- <creator>: indica il creatore, persona singola o società, dell'ebook;
- <contributor>: indica le persone singole o gli enti che hanno contribuito alla creazione dell'epub.

Questi due elementi dispongono di ulteriori specificazioni di ruolo, tramite la stringa:

```
opf:role=”ruoloDaInserire”
```

I vari ruoli possono essere, ad esempio:

  - *aut* per indicare l'autore;
  - *cmm* per indicare il commentatore;
  - *drs* per indicare il designer;
  - *edt* per indicare l'editore;

- *ill* per indicare l'illustratore.

È importante separare i ruoli di ogni collaboratore scrivendo ognuno di essi in una riga di metadati separata.

Nella sezione <manifest> vengono specificati tutti i documenti, nei vari formati ammessi[3], utilizzati dall'ebook, quindi sia quelli visualizzati nello schermo (documenti (X)HTML che comprendono il testo proprio del libro elettronico) che quelli che servono a permettere una determinata visualizzazione (come ad esempio i fogli di stile). L'ordine della scrittura di questi elementi è irrilevante, ma è necessario indicarli tutti.
Quelli che non vengono utilizzati all'interno dell'ebook, anche se indicati nella sezione <manifest>, non provocano errore, invece se vengono utilizzati e non compaiono nell'elenco provocano un errore di visualizzazione. Non sono ammesse voci identiche all'interno di questa sezione, e ogni elemento deve comparire una sola volta. Ogni elemento deve avere questi attributi:

- un identificatore univoco (*id*), che sarà indicato nella sezione <spine> del documento;
- *href* che indica il nome e la posizione del documento, relativa alla posizione del documento `content.opf`. È permesso quindi inserire i documenti nelle sottocartelle, ma devono essere indicate anch'esse;
- la specificazione del tipo di file (*media-type*).

```
<manifest>
    <item href="stylesheet.css" id="css" media-type="text/css"/>
    <item href="titlepage.html" id="titlepage" media-
type="application/xhtml+xml"/>
    <item href="titlepageback.html" id="titlepageback" media-
```

3 In un ebook in formato EPUB non sono ammessi tutti i tipi di file, perciò possono esserci solo queste specificazioni disponibili dentro questo elemento:
- documenti html+xml o solo xml;
- fogli di stile text/css;
- file immagini, con la struttura image/type, dove “type” può corrispondere solo a “gif”, “jpeg”, “png”, “svg+xml”;
- x-dtbook+xml per i file audio libri.

```
type="application/xhtml+xml"/>
    <item id="print.css" href="print.css" media-type="text/css"/>
    <item id="apt" href="page-template.xpgt" media-
type="application/adobe-page-template+xml"/>
    <item id="start" href="index.html" media-
type="application/xhtml+xml"/>
    <item href="index-back.1_div.1.html" media-
type="application/xhtml+xml" id="section1"/>
    <item id="ncx" href="toc.ncx" media-type="application/x-
dtbncx+xml"/>
</manifest>
```

La sezione spine specifica l'ordine di visualizzazione dei file da parte dell'ebook reader, e deve, contrariamente ai documenti precedenti, essere nell'ordine desiderato.

```
<spine toc="ncx">
    <itemref idref="titlepage" linear="yes"/>
    <itemref idref="start" linear="yes"/>
    <itemref idref="section1" linear="yes"/>
    <itemref idref="titlepageback" linear="no"/>
</spine>
```

La sezione guide è facoltativa, e serve per indicare i documenti che hanno importanza nel disegnare una struttura dell'ebook.

```
<guide>
    <reference type="text" href="titlepage.html" title="Cover"/>
    <reference type="text" title="Start" href="index.html"/>
    <reference type="text" href="index-back.1_div.1.html" title=" "/>
    <reference href="titlepageback.html" type="text" title="About this
book"/>
</guide>
```

1.1.5 toc.ncx

Questo documento scritto in XML serve per costruire un indice dei contenuti del libro elettronico che viene creato; servono quindi solo gli elementi che devono venire inseriti nel sommario, e non è necessario che vi siano tutti gli elementi presenti nell'ebook.
Non tutti gli ebook reader ne fanno uso, e la visualizzazione di questo sommario può

variare, di aspetto e posizione, a seconda del software utilizzato.

```
<ncx xmlns="http://www.daisy.org/z3986/2005/ncx/" version="2005-1">
    <head>
        <meta name="dtb:uid"
content="http://www.example.com/TEIEPUB/20120914194825"/>
        <meta name="dtb:totalPageCount" content="0"/>
        <meta name="dtb:maxPageNumber" content="0"/>
    </head>
    <docTitle>
        <text>Titolo</text>
    </docTitle>
    <navMap>
        <navPoint id="navPoint-1" playOrder="1">
            <navLabel>
                <text>[Cover]</text>
            </navLabel>
            <content src="titlepage.html"/>
        </navPoint>
        <navPoint id="navPoint-2" playOrder="2">
            <navLabel>
                <text>[The book]</text>
            </navLabel>
            <content src="index.html"/>
        </navPoint>
        [...]
    </navMap>
</ncx>
```

L'output creato utilizzando il programma Oxygen mostra un sommario completo, con un collegamento diretto ad ogni parte dell'ebook realizzato. La struttura può essere nidificata: è possibile non mostrare solo i capitoli, ma arricchire ognuno di essi con le diverse parti dei capitoli, raggiungibili quindi con un semplice click. In questo documento la suddivisione logica è basilare, e viene preso in esame solo ciò che può interessare il lettore.

La struttura è composta da un'intestazione “xml/ncx”, una sezione <head>, una sezione <docTitle>, una sezione <navMap> con all'interno le necessarie sezioni <NavPoint>.

Il primo elemento della sezione <head> all'interno del documento `toc.ncx` è un

elemento <meta> con attributo ***name*** di valore *dtb:uid*, il cui valore ***content*** deve corrispondere esattamente al *dc:identifier* contenuto del documento `content.opf`. I due elementi <meta> seguenti, con l'attributo ***name*** uguale a *dtb:totalPageCount* e *dtb:maxPageNumber*, sono obbligatori anche se non vengono utilizzati: per questo motivo il valore che mostrano è settato a 0. Il <docTitle> mostra, come si evince sia dal nome sia dal codice soprastante, il titolo dell'ebook. Il <navMap> è l'elemento che contiene la struttura che verrà visualizzata nel prodotto finale. La struttura interna di questo elemento è standardizzata e deve essere rispettata in ogni caso: non è possibile quindi cambiare la posizione di questi elementi o ometterne alcuno. La struttura può essere semplificata in questo modo:

- <navPoint>, specifica la voce del sommario, con attributo ***playOrder***, che ne indica l'ordine di visualizzazione, e attributo ***id***, che deve essere univoco e non può essere utilizzato da altri <navPoint>;
  - <navLabel>, all'interno di <navPoint>, è la trascrizione del testo che verrà visualizzato nel sommario come collegamento a quella voce stessa (quello che sarà cliccabile dall'utente finale);
  - <content>, all'interno di <navPoint> (sullo stesso piano di <navLabel>), rappresenta il nome del documento relativo alla voce. Il suo attributo ***src*** ne indica la fonte, dove è possibile raggiungerlo, ed è equivalente all'attributo ***href*** dell'elemento <item> dentro la sezione <manifest> del documento `content.opf`.

I <navPoint> possono essere trovarsi uno dentro l'altro, dato che è permessa una struttura nidificata: l'importante è che l'ordine indicato sopra sia sempre rispettato. Il fatto che gli elementi <navPoint> vengano nidificati non influisce sull'attributo ***playOrder***, che sarà sempre ordinato in sequenza, senza considerare la nidificazione degli elementi.

1.1.6 page-template.xpgt

È un documento specifico per la visualizzazione nel software “Adobe Digital Editions”: non è obbligatorio inserirlo e gli altri software per la lettura di ebook lo ignorano.
L'output creato utilizzando la trasformazione TEI EPUB di Oxygen lo crea

automaticamente: al suo interno vi sono indicazioni per descrivere le pagine.
Le indicazioni sono scritte in XML:FO, e il docuemnto è diviso in pagine master, dove ognuna delle porzioni di codice definisce come deve essere visualizzata una determinata porzione dell'ebook (può specificare le caratteristiche dei caratteri, gli incolonnamenti del testo, …).

1.1.7 encryption.xml

È un documento XML opzionale che si trova all'interno della cartella “META-INF” qualora l'archivio EPUB sia protetto. Infatti la specifica di XML ENCRYPTION permette di cifrare i documenti XML o solo alcune parti di esso, utilizzando o una chiave simmetrica (è necessaria solo una chiave per codificare e decodificare il documento) o un algoritmo a chiave chiave pubblica (è necessaria una chiave pubblica e una privata, dove ogni codifica fatta con una chiave può essere decodificata solo con l'altra).

2 I fogli di stile per la conversione dal formato XML a EPUB

2.1 Cos'è la TEI

La Text Encoding Initiative (TEI) è un'associazione no-profit, composta da istituzioni accademiche, ricercatori e studiosi di tutto il mondo. Questa collettività di persone definisce gli standard della rappresentazione dei testi in forma digitale, in modo che tutti coloro che ne producono possano seguire gli stessi schemi standard: offre, nel suo sito, le linee guida che descrivono schemi di codifica sfruttabili da qualunque utente si voglia cimentare nel suo utilizzo, in modo, qualora lo desiderasse, possa modificarli e adattarli ai propri scopi. L'obiettivo che si prefigge è creare e mantenere degli standard per la rappresentazione di testi digitali, utilizzabili per la ricerca online, l'insegnamento o la preservazione degli stessi, e rendere più agevole l'interscambio e l'integrazione dei dati.

2.2 I fogli di stile TEI XSL e il foglio di stile EPUB

I fogli di stile TEI XSL offrono un set di fogli di stile XSL 2.0[4] per trasformare i documenti TEI XML in documenti di tipo HTML, LaTeX, XSL-FO, nel formato DOCX, nel formato ODT ed EPUB, in modo completamente gratuito. In questi fogli di stile non tutte le opzioni disponibili tra i moduli TEI vengono presi in esame: è possibile trovare fogli di stile vuoti, comunque inseriti all'interno dei fogli di stile per permettere a chi lo desiderasse di modificarli per adattarlo alle proprie esigenze. L'intero pacchetto di fogli di stile TEI è scaricabile all'indirizzo http://tei.sf.net , ma chi utilizza il programma Oxygen li trova integrati all'interno nel momento in cui lo scarica.

2.3 Perché utilizzare Oxygen

2.3.1 Quali sono i suoi vantaggi

La scelta ricade su questo programma perché, oltre a consentire elaborazioni standard su documenti XML, fogli di stile XSLT e di tanti altri formati, permette le trasformazioni dal formato XML al formato EPUB in maniera automatica, sfruttando i fogli di stile definiti dalla TEI.

I vantaggi rispetto ad altri programmi, come ad esempio InDesign (altro programma largamente utilizzato per la creazione e personalizzazione di ebook in formato EPUB), risiedono soprattutto nella facilità con cui è permessa la trasformazione utilizzando di default i fogli di stile definiti dalla TEI, e la loro personalizzazione. Dispone di messaggi di supporto automatici che indicano all'utente se sta commettendo un errore durante la scrittura/modifica del documento su cui sta lavorando e di verificarne la validità direttamente dal programma.

Aprendo un archivio EPUB con Oxygen è possibile navigare, attraverso il pannello "Browser degli Archivi", dentro di esso, visualizzandone la struttura e i documento all'interno: è possibile accedervi e modificarli, spostarli o cancellarli.

All'occorrenza, un ebook in formato EPUB può essere creato tramite template predefiniti; è anche possibile confrontare due archivi EPUB tra loro, visualizzandone le differenze di codice e permettendo di compiere operazioni su di essi durante il processo.

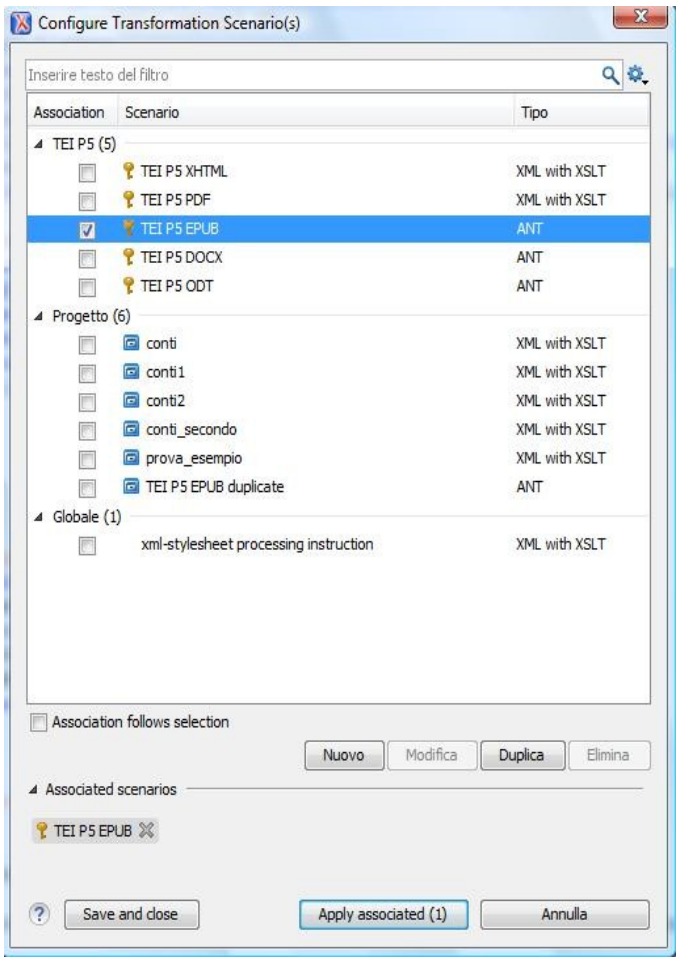
4 L'XSL è il linguaggio di descrizione dei fogli di stile per i documenti in formato XML. È l'acronimo di eXtensible Stylesheet Language.

2.3.2 Come utilizzare le trasformazioni in Oxygen

Oxygen permette di applicare le trasformazioni TEI su di un documento XML in maniera automatica. Il documento in input deve necessariamente essere un documento XML che rispetti gli standard TEI, altrimenti la trasformazione non può avvenire. Per configurare il proprio scenario di trasformazione, basta premere il pulsante apposito (segnato in rosso nell'immagine sottostante):

Premendo quel pulsante, appare questa schermata che mostra le trasformazioni possibili/consigliate da applicare ad esso:

Riguardo a queste operazioni, possiamo vedere:



- se il foglio di stile è associato al documento o meno;
- il nome dello scenario;
- il tipo dello scenario.

I tipi di trasformazioni applicabili in questo caso sono due: una trasformazione XML utilizzando un foglio di stile XSLT, oppure una trasformazione ANT (che è il tipo di trasformazione che viene eseguita nel caso di una trasformazione da XML a formato EPUB). Una trasformazione ANT utilizza il programma Apache Ant Java e permette di processare più fogli di stile XSLT insieme.

2.3.3 I fogli di stile presi in esame per modificare l'output

Per modificare l'output automatico, rendendolo personalizzato rispetto alle proprie esigenze[5], è opportuno guardare dentro la cartella:

```
Oxygen XML Editor 14[6]\frameworks\tei\xml\tei\stylesheet
```

La maggior parte dei fogli di stile che compiono le trasformazioni essenziali si trovano nella sottocartella “epub”, ma può essere utile attuare delle modifiche, a seconda delle necessità, in fogli di stile nelle sottocartelle elencate successivamente, che vengono comunque utilizzati nella trasformazione. Questo perché i fogli di stile nella cartella “epub” si occupano solo della trasformazione EPUB di un documento XML, ma un archivio EPUB comprende nel suo output anche documenti del tipo XHTML, HTML, CSS e altri. Perciò, per creare gli output, questi fogli di stile si appoggiano ad altri (utilizzati anche per compiere altre trasformazioni). I fogli di stile vengono richiamati dal foglio di stile tei-to-epub.xsl, che li importa per utilizzarli e compiere le operazioni necessarie alla trasformazione, tramite il codice:

```
<xsl:import href="../xhtml2/tei.xsl"/>
```

Viene così richiamato il foglio di stile tei.xsl che si trova nella cartella “xhtml2”, tramite

5 Permettendo così al creatore dell'ebook di non dover andare ad agire manualmente sui documenti all'interno del'archivio EPUB dopo che questo è stato creato.
6 Si riferisce alla cartella dove è installato il programma, che potrebbe variare a seconda del sistema operativo utilizzato o delle preferenze dell'utente.

il comando “import”, che serve a importare il contenuto di un foglio di stile in un altro. In totale, i fogli di stile che verranno utilizzati dalla trasformazione EPUB sono:

Cartella "epub"	Cartella "common2"	Cartella "xhtml2"	Cartella "profiles/default/epub"
build-to.xml	tei-param.xsl	tei-param.xsl	to.xsl
tei-to-epub.xsl	core.xsl	core.xsl	
epub-common.xsl	textstructure.xsl	corpus.xsl	
epub-preflight.xsl	header.xsl	drama.xsl	
	linking.xsl	figures.xsl	
	figures.xsl	header.xsl	
	textcrit.xsl	linking.xsl	
	i18n.xsl	namesdates.xsl	
	tei.xsl	textstructure.xsl	
	verbatim.xsl	textcrit.xsl	
	functions.xsl	tei.xsl	

2.4 Fogli di stile resi disponibili

2.4.1 Fogli di stile che si trovano all'interno della cartella “epub”

Questa cartella contiene i fogli di stile TEI di importanza maggiore per la trasformazione da XML a EPUB.

2.4.1.1 tei-to-epub.xsl

Si trova al seguente percorso:

```
Oxygen XML Editor 14\frameworks\tei\xml\tei\stylesheet\epub
```

È il foglio di stile che compie la maggior parte del lavoro e scrive la maggior parte dell'archivio EPUB in output. Include al suo interno altri fogli di stile, che richiama nei vari template utilizzati dentro il codice; questi verranno presi in esame nelle sezioni successive, e sono:

- `../xhtml2/tei.xsl`;
- `epub-common.xsl`;
- `epub-preflight.xsl`.

All'inizio del foglio di stile vengono definiti una serie di parametri, che possono essere modificati all'occorrenza dall'utente a seconda dell'output che desidera produrre, e che vengono utilizzati da diversi template sottostanti. Di particolare interesse per chi vuole modificare l'output di un ebook sono questi parametri:

autoHead

```
<xsl:param name="autoHead">true</xsl:param>
```

Serve a costruire una sezione <head> per ogni elemento che non la presenta nel documento XML di input; è un parametro con valore booleano, e di default è impostato come true.

autoToc

```
<xsl:param name="autoToc">true</xsl:param>
```

Crea all'inizio dell'ebook un indice automatico dei contenuti; se non se ne desidera la visualizzazione, il valore va impostato su “false”.

bottomNavigationPanel

```
<xsl:param name="bottomNavigationPanel">false</xsl:param>
```

Crea in fondo ad ogni pagina dell'ebook i collegamenti per andare direttamente alla home di questo, e i collegamenti per andare al capitolo precedente e a quello successivo. In origine è impostato come “false”, quindi per ottenere questo tipo di output è necessario cambiarne il valore in “true”.

footnoteBackLink

```
<xsl:param name="footnoteBackLink">true</xsl:param>
```

Crea dei link che collegano le note a piè di pagina ai relativi riferimenti.

linkPanel

```
<xsl:param name="linkPanel">false</xsl:param>
```

Crea, se impostato su “true”, dei collegamenti che rimandano alla pagina precedente e a quella successiva.

splitLevel

```
<xsl:param name="splitLevel">0</xsl:param>
```

Decide in che modo vengono divise le sezioni. Nel momento in cui processa un o un <div[0-5]> decide quando iniziare una nuova pagina HTML. Dal momento che lo standard TEI prevede una pagina HTML all'interno dell'ebook per ogni (numerato o non) che è presente nel documento di input XML, impostare questo parametro a un numero più alto di 0 permetterà di avere una pagina HTML singola per ogni . In origine infatti il parametro è 0, e comporta la scrittura da parte del programma di tutti gli elementi all'interno dello stesso documento. Se da un lato questo permetterà di avere ebook di grandezza minore, comporterà anche una minore navigabilità del testo: viene creato un collegamento tra il titolo della porzione di testo all'interno dell'elemento e la tabella di contenuti all'inizio dell'ebook, quindi se devono essere gestiti molte porzioni di testo divise in può essere possibile ottenere risultati che presentino collegamenti sbagliati se questo è 0, in quanto solo alla prima parte dei contenuti verrà aggiunto un collegamento diretto. Le successive, considerate parte della prima, rimanderanno al primo nel momento in cui vengono selezionate. È consigliato quindi adattare questo parametro alle proprie esigenze.

tocDepth

```
<xsl:param name="tocDepth">5</xsl:param>
```

Profondità a cui può arrivare, al massimo, la tabella dei contenuti; in origine è 5.

tocFront

```
<xsl:param name="tocFront">true</xsl:param>
```

Include nella tabella dei contenuti il peritesto iniziale, segnato nel documento XML di input da l'elemento <front>. Questo elemento può contenere qualsiasi contenuto peritestuale, come intestazioni, dediche, frontespizio, eccetera.

topNavigationPanel

```
<xsl:param name="topNavigationPanel">false</xsl:param>
```

Mostra un pannello di navigazione all'inizio di ogni pagina se impostato come “true”.

useHeaderFrontMatter

```
<xsl:param name="useHeaderFrontMatter">false</xsl:param>
```

Titolo, autore e data vengono ricavati dalla sezione <teiHeader> invece di essere ricavati dalle sezioni di peritesto se impostato su “true”.

Dopo che sono stati settati i parametri necessari per le operazioni svolte dai template, inizia <processTEI>, il template che comprende la maggior parte delle operazioni svolte dal foglio di stile: passa dall'applicare le trasformazioni standard, al sistemare i link che rimandano ai fogli di stile grafici e ad altri documenti relativi, eccetera.
Il template <processTei> parte verificando la presenza o meno una immagine di copertina già disponibile documento XML in input; qualora non sia presente, prende in esame la porzione di testo all'interno dell'elemento <titlePage>, per poter inserire un titolo che svolga la funzione di “copertina” del testo. Dopo di ciò, verifica che la variabile *$splitLevel* non sia impostata a -1, in modo tale che il programma possa decidere se deve iniziare a dividere il testo e in quale modo deve farlo. Viene richiamato il template <mainToc>, che crea il documento `toc.ncx`: nel caso in cui viene applicata la trasformazione da XML a EPUB per creare un ebook, il documento `toc.ncx` mostra i collegamenti alla sezione <header>, alle sezioni <front>, <body> e <back>. Per ognuno di essi viene creato un collegamento che permette, se l'ebook reader lo consente, di raggiungere cliccandolo la porzione di testo desiderata. Gli ebook in output che vengono definiti di seguito devono essere “puliti”, senza spazi non necessari (che

possono provocare errori): durante la scrittura dei documenti che fanno parte dell'ebook viene quindi verificato se l'attributo *$verbose* sia uguale a “true”, che determina se vi siano spazi superflui in una determinata parte di testo, e vengono normalizzati gli spazi in ogni porzione di testo che utilizza questo attributo (il numero di spazi eccessivi viene sostituito da un unico carattere di spazio). Esempio di codice che verifica se vi sono spazi superfluit:

```
<xsl:if test="$verbose='true'">
 <xsl:message>write file OPS/print.css</xsl:message>
</xsl:if>
```

Esempio di codice che viene utilizzato per normalizzare gli spazi:

```
<xsl:variable name="file" select="normalize-space(.)"/>
```

Iniziano quindi, utilizzando questo metodo, ad essere scritti uno per uno i file che si troveranno successivamente dentro l'ebook. Non è consigliabile modificare questo template se non in minima parte, in quanto utilizza tantissime variabili per gli utilizzi più svariati, richiamandole in più porzioni del codice ricorsivamente. Il rischio di compromettere l'ordine di scrittura dei documenti dell'archivio EPUB che viene creato è alto e questo è indispensabile per ottenere un ebook funzionante.
Dopo il template “processTei” inizia una serie di template che vengono richiamati all'interno di processTEI:

- per la gestione di elementi estratti da documento HTML;
- per la gestione di documenti javascript che possono essere presenti nell'output creato;
- per la creazione di sezioni all'interno del documento `content.opf` che viene creato.

2.4.1.2 epub-preflight.xsl

È il foglio di stile che comprende la maggior parte dei template in modalità *preflight*, che agiscono sul testo prima delle altre porzioni di codice. Il suo scopo è di pulire il

testo prima di creare l'output ebook. Viene ricopiato, per prima cosa, tutto il documento in input:

```
<xsl:template match="@*|text()|comment()|processing-instruction()"
mode="preflight">
  <xsl:copy-of select="."/>
</xsl:template>
```

Gestisce successivamente le interruzioni di pagine e le enfatizzazioni.

2.4.1.3 epub-common.xsl

Inizia con una serie di <xsl:key>, codice utilizzato per dichiarare chiavi, che specificano una coppia nome/valore da assegnare ad un elemento. Queste chiavi permettono di trovare più facilmente un nodo che corrisponda al suo match pattern se il valore della chiave specificata (se applicata a quel nodo) è conosciuta.

```
<xsl:key name="GRAPHICS" use="1" match="tei:graphic"/>
```

L'esempio precedente è la dichiarazione di una chiave: questa in particolare viene utilizzata dal foglio di stile `figures.xsl` che si trova all'interno della cartella "xhtml2". Vengono settati due parametri.

```
<xsl:param name="epubMimetype">application/epub+zip</xsl:param>
```

Questo parametro, tra i due, è di importanza rilevante, in quanto definisce ciò che deve essere inserito nel documento `mimetype`, che è standard e non deve mai essere modificato.
Le operazioni effettuate successivamente sono:

- gestisce i piè di pagina;
- sceglie la licenza, quella di default è Creative Commons Attribution;
- sceglie la lingua, se non trova specificazioni diverse viene considerato un testo input inglese di default;
- verifica se è presente uno o più generi (categoria) in cui può essere inserito il testo

(fantascienza, fantasy, ...);

- determina qual è il nome dell'editore (publisher);
- sceglie un identificatore univoco per l'ebook (se non è già presente, lo genera utilizzando questa espressione:

```
<xsl:text>http://www.example.com/TEIEPUB/</xsl:text>
 <xsl:value-of select="format-dateTime(current-dateTime(),'[Y][M02]
[D02][H02][m02][s02]')"/>
```

che crea un collegamento al sito "example.com/TEIEPUB/" e stampa alla fine la data esatta fino ai secondi (in questo modo viene resa univoca);

- rimuove gli elementi superflui dal foglio di stile CSS;
- stampa diversi tipi di titoli;
- gestisce l'output delle liste (scrive l'output all'interno di elementi <ul> se l'input XML è <list>, e all'interno di elementi <li> se l'input XML è <item>);
- vengono stampati i nomi di altri tipi di persone che possono aver lavorato alla creazione del documento di input opzionali (come il distributore), e di altri elementi opzionali (come il titolo alternativo);
- viene gestito l'output della sezione <head>;
- forza l'output a creare due fogli di stile CSS, uno chiamato `stylesheet.css` e uno chiamato `print.css`.

2.4.1.4 build-to.xml

Questo documento XML è importantissimo per la trasformazione che viene applicata, e il suo elemento radice è:

```
<property name="inputFile" value=""/>
```

Questo elemento radice può avere diversi attributi, che sono facoltativi (se non vengono aggiunti dall'utente, non sono presenti):

- nome (*name*): specifica il nome del progetto;
- predefinito (*default*): specifica la destinazione predefinita. Serve se non viene

specificata la destinazione nella riga di comando; qualora siano presenti entrambi, la destinazione specificata nella riga di comando è prevalente.

La stringa di codice successiva serve a indicare il percorso dove verrà salvato l'ebook:

```
<property name="outputFile" value=""/>
```

Questo accade ovviamente se non viene specificato il percorso nella riga di comando della trasformazione. La stringa di codice che compare subito dopo serve a determinare quale profilo utilizzare per la trasformazione:

```
<property name="profile" value="default"/>
```

Nel caso delle trasformazioni EPUB utilizzando lo standard TEI, i profili di default utilizzati si trovano nella cartella del programma Oxygen:

```
Oxygen XML Editor 14\frameworks\tei\xml\tei\stylesheet\profiles\default
```

Utilizzando la trasformazione automatica con il programma Oxygen, è utile questa stringa di testo:

```
<property name="oxygenlib" value=""/>
```

Infatti questa indica dove il programma Oxygen contiene il materiale utilizzato nella trasformazione. Dopo queste proprietà inizia il corpo che serve alla trasformazione: questo permette agli altri fogli di stile di richiamare i vari elementi da utilizzare durante il processo. Ad esempio:

```
<property name="outputTempDir" value="${outputDir}/temp-dir-for-ant"/>
```

Crea una cartella temporanea per l'output, che verrà successivamente cancellata, dove gli altri fogli di stile “depositano” i loro risultati che verranno utilizzati per la creazione dell'ebook. `build-to.xml` inoltre stampa dei messaggi nel testo che viene visualizzato dall'utente durante la trasformazione:

```
<echo level="info">XSLT generate ePub files</echo>
```

Gli altri parametri utilizzati sono di particolare importanza per la loro relazione con gli altri fogli di stile della trasformazione. Esempi:

splitLevel

```
<param name="splitLevel" expression="${splitLevel}" if="splitLevel"/>
```

Configura la divisione dei livelli.

subject

```
<param name="subject" expression="${subject}" if="subject"/>
```

Determina il genere dell'ebook.

nocompress

```
<param name="nocompress" expression="${nocompress}" if="nocompress"/>
```

Disabilita la compressione per l'output ZIP: serve per la creazione e l'inserimento del documento `mimetype` dentro l'ebook in formato EPUB, che non deve mai essere compresso.

Terminata la sezione dove vengono settati questi parametri, inizia una porzione di codice XML che salva i file temporanei in output dentro la cartella apposita (“temp-dir-for-ant”), scrive la trasformazione ed estrae i nomi dei fogli di stile grafici. Dopo di ciò vengono trasferiti tutti i file creati dalla cartella temporanea a quella definitiva,e la cartella temporanea viene cancellata:

```
<delete dir="${outputTempDir}"/>
```

2.4.2 Fogli di stile all'interno della cartella “common2”

2.4.2.1 tei.xsl

La sua funzione principale è quella di importare diversi fogli di stile della sua stessa cartella di appartenenza:

```
<xsl:import href="tei-param.xsl"/>
<xsl:import href="core.xsl"/>
<xsl:import href="textstructure.xsl"/>
<xsl:import href="header.xsl"/>
<xsl:import href="linking.xsl"/>
<xsl:import href="figures.xsl"/>
<xsl:import href="textcrit.xsl"/>
<xsl:import href="i18n.xsl"/>
<xsl:import href="functions.xsl"/>
```

Dopodiché, definisce una lista di elementi sulla quale deve essere applicato il comando *strip-space*, il comando che viene utilizzato per togliere gli spazi superflui, sostituendoli con uno spazio singolo. Gestisce la grafica delle citazioni, a seconda di come sono state definite nel documento di input:

```
[…] <xsl:when test="$quote='laquo'">«</xsl:when>
<xsl:when test="$quote='ldquo'">“</xsl:when> […]
```

Codice di esempio estratto dal foglio di stile, che specifica alcune delle operazioni compiute, come verificare se il valore *$quote* è uguale a 'laquo' oppure a 'ldquo', e decidere di conseguenza quale stile grafico deve essere adottato per quella citazione. I punti di sospensione tra le parentesi quadre indicano una porzione di codice saltata per sintesi.
Vengono infine definiti altri template, utilizzati poi richiamandoli in altri fogli di stile (ad esempio, *tei:makeText* viene utilizzato per stampare porzioni di testo).

2.4.2.2 tei-param.xsl

È un modulo per la personalizzazione per tutti i formati di output. Come si può evincere dal suo nome, imposta una lista di parametri utilizzati dagli altri fogli di stile. Gestisce anche la costruzione della sezione <head>, se questa non è presente nel documento di input; inoltre, gestisce template come quelli che si occupano della punteggiatura dopo un numero di sezione (se non presente), e altri tipi di numerazione che possono essere presenti in un documento, come la numerazione delle tabelle.

2.4.2.3 textstructure.xsl

Questo foglio di stile si occupa degli elementi del modulo TEI "textstructure". È il modulo che si occupa delle sezioni principali di alto livello di un documento TEI. In particolare si occupa di:

- stabilire la profondità delle sezioni (ammessa fino al livello <div6>);
- generare una descrizione della revisione (se presente).

2.4.2.4 header.xsl

Foglio di stile TEI che si occupa dell'intestazione TEI, nel dettaglio gestisce:

- come trovare un nome di autore corretto da utilizzare all'interno dell'ebook (cerca quindi tra le varie opzioni XML che potrebbero essere state utilizzate per definire il nome dell'autore: <author>, <docAuthor>, eccetera):
- come trovare un nome di editore corretto (come sopra);
- come trovare un nome corretto per colui che ha revisionato il testo in input;
- calcola la data dell'ultima revisione del documento;
- calcola la data della pubblicazione del documento;
- come trovare il titolo (togliendo gli elementi di marcatura che potrebbero essere inseriti con il titolo);
- come trovare il sottotitolo;
- omette la visualizzazione di <docAuthor> e <docDate> se trovati fuori dal peritesto

(<front>);

- ignora l'elemento <docTitle> se inserito in un elemento .

2.4.2.5 linking.xsl

Foglio di stile TEI che si occupa degli elementi del modulo “linking”.
Questo modulo gestisce i puntatori che servono a collegare elementi diversi in un unico documento o in diversi documenti. Gli elementi più importanti di questo foglio di stile, per coloro che fossero interessati a modificarlo, sono la gestione dei collegamenti interni all'ebook che viene creato, e la gestione dei collegamenti nel documento di input che utilizzano lo standard TEI P4, configurando come gestire le differenze rispetto allo standard TEI P5.
Per gestire i collegamenti locali, è norma utilizzare il carattere “#” all'inizio del collegamento stesso; dopo aver verificato che il collegamento preso in esame inizi con il carattere “#”, viene richiamato, dall'omonimo foglio di stile situato nella cartella “xhtml2”, il template corrispondente per crearne l'output. Diversamente, i collegamenti verranno considerati esterni, e verrà richiamato un altro template corrispondente. Se viene riconosciuto un collegamento TEI P4, viene automaticamente considerato locale al documento in questione.

2.4.2.6 figures.xsl

Foglio di stile TEI che si occupa degli elementi del modulo “figures”. Prende in esame tutte le figure e le tabelle, ne calcola il numero e ne analizza gli attributi.

2.4.2.7 textcrit.xsl

Foglio di stile TEI che si occupa degli elementi del modulo “textcrit”. Il modulo “textcrit” si occupa di edizioni critiche di testi, in particolare i testi di grande antichità o importanza; infatti l'abbreviazione “textcrit” sta per “testo critico”. Questi testi presentano differenze tra versione critica a versione critica: uno studio completo di un testo antico deve poter contenere ogni traduzione attribuita ad esso. Vengono quindi contemplate le varianti che possono trovarsi all'interno di un testo. Prende in esame gli elementi di marcatura <app> che serve a contenere la voce critica che viene aggiunta:

dentro di esso può essere collocato un elemento <lem>, facoltativo, che può contenere diversi elemento <rdg>. L'elemento <rdg> serve a contenere un modo di lettura all'interno di una variante testuale, e <lem> viene usato per contenere il lemma, o testo di base, di una variante testuale. L'elemento di marcatura<rdg> può anche trovarsi all'esterno di un elemento <lem>: per questo motivo il foglio di stile `textcrit.xsl`, quando processa un elemento <app>, se non trova elementi <lem> considera il primo <rdg> trovato come base del testo, e spedisce le informazioni a piè di pagina.

2.4.2.8 i18n.xsl

Si occupa di definire i fogli di stile TEI comuni per ogni output HTML, FO e LaTex. il18n è un'abbreviazione inglese che significa "internazionalizzazione e la localizzazione" e indica quando un determinato prodotto viene adattato, generalmente per essere utilizzato da tutte le persone appartenenti ad una certa cultura.
In generale:

- assegna una versione specifica di lingua di una parola o una frase da tradurre;
- dove necessario annulla in un sistema locale la parola da tradurre;
- determina la lingua da usare quando deve essere generato del testo, usando il sistema di codifica ISO 2[7];
- prefissa il testo prima di auto generare una tabella dei contenuti;
- genera automaticamente titoli per le sezioni <contenuti>, <feedback>, <ricerca>.

2.4.2.9 functions.xsl

Questo foglio di stile si occupa di definire le funzioni utilizzabili da tutti i formati possibili in output. In particolare si occupa di:

- verificare se una sezione è identificabile (per questo motivo distingue tra sezioni inserite all'interno dell'elemento di vario livello, sezioni <p> e sezioni <index>; il resto è considerato come non identificabile);
- verificare se una sezione è trascrivibile (sono considerate trascrivibili le sezioni <p>, <sp> e <l> di un testo XML);
- determina quando del testo in output va impostato come maiuscoletto;

7 Codice standard per la rappresentazione delle lingue: http://www.loc.gov/standards/iso639-2/php/code_list.php.

- determina quando un parola o frase va inserita tra virgolette;
- verifica quando deve rendere in ouput del testo in grassetto, corsivo, in stile codice (utilizzando un font che simula la macchina da scrivere);
- decide se un elemento debba essere reso *inline* o meno;
- prende in esame diversi elementi e decide se possono essere considerati sullo stesso livello o meno.

2.4.2.10 core.xml

È il foglio di stile che si occupa della gestione degli elementi in un documento XML che sono inclusi nel modulo "core". Questo modulo raggruppa gli elementi che potrebbero apparire in un qualunque tipo di testo, e include:

- paragrafi;
- evidenziazioni, enfasi e citazioni;
- modifiche redazionali semplici;
- numeri trascritti, date, indirizzi;
- collegamenti e riferimenti incrociati;
- liste, note, annotazioni, indicizzazioni;
- grafici;
- sistemi di riferimento, citazioni bibliografiche;
- poesie in versi.

Il foglio di stile inizia con un comando di rimozione degli spazi superflui, da applicare ai nomi proprio che potrebbero essere scritti nel documento di input:

```
<xsl:strip-space elements="tei:author tei:forename tei:surname tei:editor"/>
```

Vengono dichiarate due chiavi che permettono di distinguere vari elementi appartenenti alla sezione <biblStruct>:

```
<xsl:key name="MNAMES"
```

```
    match="tei:monogr/tei:author[tei:surname]|
tei:monogr/tei:editor[tei:surname]"
    use="ancestor::tei:biblStruct/@xml:id"/>
<xsl:key name="ANAMES"
    match="tei:analytic/tei:author[tei:surname]|
tei:analytic/tei:editor[tei:surname]"
    use="ancestor::tei:biblStruct/@xml:id"/>
```

MNAMES sceglie l'autore o l'editore nel caso in cui questi siano definiti all'interno degli elementi <monogr>, mentre *ANAMES* viene scelto se l'autore o l'editore vengono definiti all'interno dell'elemento <analytic>. Questo perché in XML entrambe le strutture sono ammesse, e a scelta di chi crea il documento. Siccome entrambe le chiavi hanno come parametro *use* la stessa espressione, non è importante per il programma se l'autore e l'editore fanno parte degli *MNAMES* o degli *ANAMES*: il risultato sarà in entrambi i casi la selezione degli elementi, nello stesso identico modo. Vengono processati tutti gli elementi per scoprire la loro profondità di annidamento, permessa fino al novantanovesimo livello. Dopodiché vengono elaborati tutti gli elementi in modalità normale:

```
<xsl:template match="tei:*" mode="plain">
      <xsl:apply-templates mode="plain"/>
</xsl:template>
<xsl:template match="tei:note" mode="plain"/>
```

Gli elementi processati sono molteplici, ma per questioni di sintesi qui viene inserito solo il template con `match=”tei:note` a scopo di esempio. Il risultato per questi elementi è di essere applicati dove indicato; questo perché se un elemento <xsl:apply-templates> ha come attributo ***mode***, applica ai <xsl:template> successivi con l'attributo “mode” equivalente solo quelle regole. Tutti gli elementi <item>, che di norma (secondo gli standard TEI) vengono utilizzati come elementi di un elenco, emettono in output un elenco puntato:

```
<xsl:template match="tei:item" mode="runin">
    <xsl:text> • </xsl:text>
    <xsl:apply-templates/>
    <xsl:text> </xsl:text>
</xsl:template>
```

Vengono presi in esame gli elementi <edition> ed <imprint>; entrambi possono trovarsi all'interno, nel documento di input XML, degli elementi <biblStruct> o <biblFull>. Nel primo template, <edition>, viene richiamato un altro template, <makeText>, per stampare qualsiasi cosa sia scritto all'interno del documento di input dentro gli elementi <edition>. Per gli elementi <imprint> invece, viene utilizzato un meccanismo di scelta:

```
<xsl:template match="tei:imprint">
    <xsl:choose>
        <xsl:when test="ancestor::tei:biblStruct or
ancestor::tei:biblFull">
            <xsl:apply-templates select="tei:date"/> <xsl:apply-
templates select="tei:pubPlace"/> <xsl:apply-templates
select="tei:publisher"/> <xsl:apply-templates select="tei:biblScope"/>
        </xsl:when>
        <xsl:otherwise> <xsl:apply-templates/> </xsl:otherwise>
</xsl:choose>
</xsl:template>
```

Se non vengono trovati gli elementi <date>, <pubPlace>, <publisher> e <biblScope>, viene dato in output ciò che si trova dentro <imprint>, senza ordine. Viene definito come deve essere stampato tutto ciò che è all'interno di <biblStruct>: si tratta di un template che fornisce molte alternative, dato che la sua struttura XML è molto variabile, e fa largo uso delle chiavi definite all'inizio del foglio di stile.

```
<xsl:template match="tei:biblStruct" mode="xref">
    <xsl:choose>
        <xsl:when test="count(key('ANAMES',@xml:id))=1">
            <xsl:value-of select="key('ANAMES',@xml:id)/tei:surname"/>
        </xsl:when> […] <xsl:when test=".//tei:author[tei:surname]">
            <xsl:value-of select=".//tei:author/tei:surname[1]"/>
        </xsl:when> […] </xsl:choose>
</xsl:template>
```

Il codice soprastante è un esempio di come viene scelto il cognome dell'autore che verrà inserito nell'output, che può essere presente a un qualsiasi livello. Le parentesi quadre con dentro i puntini di sospensione indicano delle porzioni di codice che sono state omesse. Vengono gestiti successivamente gli output per i nomi degli autori e degli

editori: viene verificato se, in entrambi i casi, si tratta di un autore o editore singolo, o di un gruppo di persone. Questo per poter stampare alla fine dei titoli “nome autore” e “nome editore” la “s” finale del plurale inglese, oppure per stampare degli “and” tra un nome di autore/editore. Vengono configurate altre porzioni del testo in input, come il titolo, la collana, la casa editrice, i capitoli...

2.4.2.11 verbatim.xsl

Anche questo foglio di stile tratta di elementi presenti nel modulo “core”. In generale:

- elabora un prefisso per il namespace degli elementi testuali;
- processa i commenti e i nodi di testo nella modalità “verbatim”, che inserisce interruzioni di pagina prima dell'elemento preso in esame (*linebreak*, il cui elemento è “lb”);
- prende in esame i caratteri di testo e i simboli commerciali, per determinare se tra questi vi sono ideogrammi di varie lingue orientali;
- processa altri elementi di default, sempre in modalità “verbatim”.

2.4.3 Fogli di stile all'interno della cartella “xhtml2”

2.4.3.1 tei.xsl

È il foglio di stile adibito a creare gli output HTML. Per prima cosa importa una serie di foglio di stile:

```
<xsl:import href="../common2/tei.xsl"/>
<xsl:import href="tei-param.xsl"/>
<xsl:include href="core.xsl"/>
<xsl:include href="corpus.xsl"/>
<xsl:include href="dictionaries.xsl"/>
<xsl:include href="drama.xsl"/>
<xsl:include href="figures.xsl"/>
<xsl:include href="header.xsl"/>
<xsl:include href="linking.xsl"/>
<xsl:include href="namesdates.xsl"/>
```

```
<xsl:include href="tagdocs.xsl"/>
<xsl:include href="textstructure.xsl"/>
<xsl:include href="textcrit.xsl"/>
<xsl:include href="transcr.xsl"/>
<xsl:include href="verse.xsl"/>
<xsl:include href="../common2/verbatim.xsl"/>
```

Questi foglio di stile quindi collegano il foglio di stile `tei-to-epub.xsl` ai foglio di stile citati sopra, che si trovano all'interno della cartella "common2" e della cartella "xhtml2". Esistono altri foglio di stile all'interno di queste cartelle, ma non fanno parte della trasformazione TEI da XML a EPUB.

2.4.3.2 tei-param.xsl

Questo foglio di stile si occupa della personalizzazione dell'output HTML, e inizia dichiarando una serie di chiavi, che specificano come deve essere trovato un elemento:

```
<xsl:key name="NOTES" use="1" match="tei:note[@place='foot' or @place='bottom' or @place='end' and not(parent::tei:bibl or ancestor::tei:teiHeader)]"/>
```

In questo caso le note sono identificate come testo che si trova nella sezione <foot>, <bottom> oppure <end>, e che non siano inserite nei gruppo di marcatura di <bibl> o <teiHeader>. Viene definita una serie di parametri per l'utilizzo del linguaggio CSS e HTML, e altri elementi adibiti alla personalizzazione dell'output.

2.4.3.3 core.xsl

Lo scopo di questo foglio di stile è lo stesso già analizzato nel foglio di stile `core.xsl` dentro la cartella "common2": processare tutti gli elementi che fanno parte del modulo core: per prima cosa definisce che ogni elemento con un <head> può essere scritto dentro il documento TOC che viene creato. Per tutti gli elementi presi in esame viene richiamato il template <rendToClass>, che converte l'attributo ***rend*** per quell'elemento in una classe HTML.
L'attributo ***rend*** indica come l'elemento in questione è stato reso o rappresentato nel

testo di partenza.

2.4.3.4 corpus.xsl

Si tratta del foglio di stile TEI che prende in esame gli elementi dal modulo “corpus”, creando un output HTML. Questo modulo prende in esame interi gruppi di corpus: dato che il modulo “corpus” tratta essenzialmente degli elementi di marcatura <teiCorpus>, <teiHeader> e <text>, il foglio di stile gestisce principalmente questi elementi. Per ogni corpus deve essere restituito un output HTML separato da quello degli altri testi dentro al corpus: ogni testo deve avere la sua intestazione <head> HTML, eccetera.

2.4.3.5 dictionaries.xsl

Definisce come si deve comportare il foglio di stile TEI con gli elementi del modulo <dictionaries>, che si occupa della codifica di risorse lessicali di tutti i tipi, in particolare dizionari monolingue e multilingue, glossari, e documenti simili. Il foglio di stile è vuoto: la gestione di eventuali elementi di questo modulo, se non sono indicati da altre parti, viene lasciata a chi desiderasse personalizzare il proprio output.

2.4.3.6 drama.xsl

Questo foglio di stile prende in esame gli elementi presenti nel modulo “drama” per creare output HTML. Il modulo “drama” si occupa di gestire gli elementi di marcatura dedicati ai drammi teatrali e testi simili: gestisce quindi elementi come <actor>, <castGroup>, <roleDesc>, <sound>, eccetera.

2.4.3.7 figures.xsl

Lo scopo di questo foglio di stile è lo stesso già analizzato nel foglio di stile `figures.xsl` dentro la cartella “common2”. Si occupa dell'utilizzo delle immagini, delle tabelle (gestione delle righe e delle colonne), delle funzioni matematiche e dei grafici.

2.4.3.8 header.xsl

Lo scopo di questo foglio di stile è lo stesso già analizzato nel foglio di stile `header.xsl` dentro la cartella “common2”. Definisce delle chiavi per poterle utilizzare in vari template, e genera un CSS utilizzato soltanto per la sezione <header>.

2.4.3.9 linking.xsl

Il modulo utilizzato da questo foglio di stile è lo stesso già analizzato nel foglio di stile `linking.xsl` dentro la cartella “common2”. Gestisce la creazione dei collegamenti, definendo un template per ognuno dei possibili collegamenti inseribili in un testo.

2.4.3.10 namesdates.xsl

Definisce come si deve comportare il foglio di stile TEI con gli elementi del modulo “namesdates”, che viene utilizzato per la codifica dei nomi e frasi descrittive di persone, luoghi o organizzazioni, in maniera più dettagliata di quella possibile con gli elementi già previsti per questi scopi nel modulo “core”. Prende in esame e gestisce l'output di soli tre elementi: <listPerson>, <person>, <affiliation>.

2.4.3.11 tagdocs.xsl

Si tratta del foglio di stile TEI che prende in esame gli elementi dal modulo “tagdocs”, creando un output HTML. Questo modulo serve a descrivere tramite marcatura elementi particolari che possono essere inseriti dentro un documento XML. Di questi il foglio di stile `tagdocs.xsl` prende in esame l'elemento <egXML>, che serve a marcare testo XML ben formato che può essere scritto all'interno di un altro documento XML (di solito a fini didattici); <ident>, usato per definire un identificatore o un nome per un oggetto di qualche tipo in un linguaggio formale di programmazione (nomi di classe, nomi di funzioni, eccetera); <gi> che contiene l'identificatore generico di un elemento.

2.4.3.12 textstructure.xsl

Si occupa di definire la struttura finale del testo che deve comparire nell'output HTML. Per impaginare correttamente il testo, deve richiamare il template <processTEI> definito nel foglio di stile `tei-to-epub.xsl`. Per determinare se sta posizionando le porzioni di testo nelle posizioni giuste, uso dei comandi XPath “preceding-sibling” e “following-sibling”: richiamando variabili definite nel foglio di stile `tei-to-epub.xsl`, riesce in questo modo a creare un output soddisfacente. Dal momento che questo foglio di stile viene già collegato, tramite i passaggi illustrati sopra, a `tei-to-epub.xsl`, non c'è bisogno che all'inizio del codice quest'ultimo venga importato.

2.4.3.13 textcrit.xsl

Lo scopo di questo foglio di stile è lo stesso già analizzato nel foglio di stile “textcrit.xsl” dentro la cartella “common2”. Definisce una serie di parametri che verranno utilizzati nelle trasformazioni, e verifica se nel documento di input è presente una sezione dedicata all'apparato critico, per poterla gestire nel foglio di stile omonimo all'interno della cartella “common2”.

2.4.3.14 transcr.xsl

Questo foglio di stile definisce come si deve comportare il foglio di stile TEI con gli elementi del modulo “transcr”, che viene utilizzato per rappresentare i manoscritti o altri materiali scritti. È vuoto: la gestione di eventuali elementi di questo modulo, se non sono indicati da altre parti, viene lasciata a chi desiderasse personalizzare il proprio output.

2.4.3.15 verse.xsl

Si tratta del foglio di stile TEI che prende in esame gli elementi dal modulo “verse”, creando un output HTML. Questo modulo si occupa interamente di testi o porzioni di testi in versi: se ne fa uso solo qualora il modulo “core” venga ritenuto inadeguato o

troppo povero di elementi per i propri scopi. Il foglio di stile è vuoto: la gestione di eventuali elementi di questo modulo, se non sono indicati da altre parti, viene lasciata a chi desiderasse personalizzare il proprio output.

3 Le modifiche apportate

Il documento XML utilizzato per testare questi cambiamenti fa parte del progetto di codifica del testo del “Pellegrino nell'Asia” di Angelo Legrenzi, a cura dagli studenti del modulo di Codifica di testi del corso di laurea in Informatica Umanistica.
Ai fini dello studio del testo del “Pellegrino nell'Asia” è stato deciso di modificare i fogli si stile TEI per creare un ebook in formato EPUB, in maniera automatica, che agevolasse la ricerca sul testo.

3.1 Rendere i nomi di luogo in grassetto

Uno dei problemi di personalizzazione sviluppati in questo studio è la resa grafica dei nomi di luogo in grassetto, per poterli evidenziare e permettere di visualizzarli a prima vista, agevolando la ricerca dei luoghi menzionati in questo testo. Il codice che è stato inserito, in questo caso nel foglio di stile `core.xsl` all'interno della cartella “common2”, è il seguente:

```
<xsl:template match="tei:rs[@type='luogo']">
```

In questo caso è stato scelto un match che prendesse in input tutti gli “rs” che avessero come attributo `type="luogo"`. Gli <rs> sono gli elementi utilizzati per esprimere nomi o dare un'etichetta generica nel documento di input XML:

```
<p> Fornito il giro per la <rs key="Persia" type="luogo">Persia</rs>
```

Affinché vengano richiamati nella maniera corretta deve essere specificato nel match che viene applicata la tipologia di <rs> che viene selezionata. All'inizio del match la porzione di codice `tei:` è obbligatoria in questo caso, per fare sì che il codice funzioni: essendo il documento di input che viene modificato un documento standard TEI,

presenta un'intestazione TEI che dichiara il namespace relativo, e perciò deve essere dichiarato anch'esso quando viene formulato un match., per evitare che venga ripetuto nell'output.
Potrebbe essere necessario, a coloro che desiderassero ottenere output simili, inserire più match in uno stesso template, invece di scrivere diversi template che si susseguono:

```
<xsl:template match="tei:rs[@type='luogo' or 'place']"> <b><xsl:value-
of select="."/></b> </xsl:template>
```

Al posto di scrivere un'operazione tramite l'uso di alternative (if) o altre soluzioni (comunque corrette), può essere un'operazione più semplice scrivere in questo modo, se l'obiettivo è di selezionare, come in questo caso, un determinato elemento o una serie di elementi, con attributi diversi, che devono restituire lo stesso output.

1.1. Della città di Surat

Fornito il giro per la **Persia**, e posto pur anco il piede nell'**India**, mi si presenta a prima faccia lo Stato del **Rè de Mogori**, divisi questi due Potentati al fiume **Indo** a confini del regno di **Candahar**. Quando **Alessandro il Grande** comparve in questi Regni fece passare le sue milizie per via di Terra, abbandonata quella del Mare, come meno pratticata a quei tempi, qual viaggio riuscigli così laborioso per la difficoltà del passaggio de Monti, e più de fiumi, l'**Indo**, e l'**Hidaspe**, che al riferir di **Quinto Curtio**, all'ottavo e nono Libro, stanche le militie non vollero proseguir più oltre. Covene dunque al Monarca far dogliosa ritirata con il rilascio di quanti regni, e Città haveva[Page 193] conquistato, e perche gli perì il suo famoso bucefalo, ordinò che fosse sepolto, nobilitando il luogo con la fondatione d'una Città insignita con il nome di **Bucefalia**. Da che si scorge che l'**India** è Terra ferma potendosi dall'**Europa** passar à quì senza traghetto di mare, vero ben è che per le ragioni sopraccennate per la grandezza de deserti, asprezza de monti, & eccesso de calori viene al giorno d'oggi abbandonato questo viaggio. Or per

Questo esempio è utile per i documenti in input che prevedono una mescolanza di lingua inglese e italiana, oppure se vogliamo trasformare, ottenendo gli stessi risultati, documenti XML di lingue differenti, senza dover modificare continuamente i fogli di stile.
In questo modo viene presa in esame ogni necessità futura, ad esempio se si desidera compiere degli studi su un ampio numero di testi in varie lingue, e adattarli tutti alle proprie esigenze, attribuendo ad ognuno di questi le stesse caratteristiche.

3.1.1 Dove è stato inserito e perché

Potrebbe anche essere inserito dentro il foglio di stile `to.xsl`; viene cambiata però la logica dell'operazione: in quest'ultimo caso viene cambiato l'intero profilo dell'elaborazione EPUB, mentre se viene inserito il codice dentro il foglio di stile `core.xsl` si rimane fedeli alla logica con la quale sono strutturati i moduli TEI, secondo cui il modulo chiamato core raggruppa tutti gli elementi per marcare un qualsiasi tipo di testo. Essendo questa modifica inserita una modifica di enfatizzazione del testo, il foglio di stile `core.xsl` è stato ritenuto in questo caso quello più opportuno modificare.

3.2 Creazione di un indice di luoghi automatico

Per una personalizzazione più completa del documento XML preso in esame, è stato deciso di introdurre un indice dei nomi dei luoghi citati nel testo. Per attuare questa modifica ai fogli di stile ho, in un primo tempo, ipotizzato di creare un foglio di stile che contenesse i comandi per ottenere in risultato le aggiunte che intendevo inserire nel testo; questo foglio di stile avrebbe dovuto essere aggiunto dentro la cartella contenente i fogli di stile per la trasformazione da XML a EPUB, e richiamato, tramite il comando <xsl:import>, nel foglio di stile `tei-to-epub.xsl`. L'obiettivo era forzare la scrittura del testo che volevo aggiungere nel documento `content.opf` all'interno dell'ebook ottenuto, modificando il template <processTEI>.
Ho quindi contattato la mailing list del consorzio TEI[8] per ottenere suggerimenti su come modificare i fogli di stile al fine di ottenere il risultato che desideravo. Mi ha risposto Sebastian Rahtz[9], il quale mi ha spiegato che è molto più facile, piuttosto che tentare le modifiche che avevo ipotizzato, creare una porzione di codice che aggiunga una nuova sezione in fondo al testo (dentro l'elemento <back>), e inserirla nel foglio di stile `epub-preflight.xsl`. In questo modo questo cambiamento viene applicato al

8 La mailing list TEI è messa a disposizione a qualunque utente voglia iscriversi, per poter usufruire del supporto dato direttamente dalla comunità TEI. Le domande poste vengono catalogate e inserite in una lista navigabile, in modo da agevolare eventuali utenti che desiderino effettuare una ricerca sulle domande poste. Qui è possibile navigare attraverso la lista delle mail: http://listserv.brown.edu/archives/cgi-bin/wa?A0=tei-l

9 Direttore e Ricercatore dell'Università di Oxford, ed anche scrittore dei fogli di stile TEI per la trasformazione da XML a EPUB.

documento XML prima che questo venga processato dagli altri fogli di stile TEI che si occupano della trasformazione. La modalità del template inserito deve essere necessariamente, in questo caso, *preflight*. Sfortunatamente si è verificato un errore nell'output ottenuto, a causa di un bug che ho segnalato al SRathz e che mi ha assicurato studierà per far sì che non si verifichi più. L'ebook creato infatti non presentava nessuna delle sezioni che dovevano essere visualizzate. Successivamente mi è stato suggerito di apportare quelle modifiche al foglio di stile `to.xsl`, che però ha prodotto lo stesso risultato. Il problema non si verificava omettendo l'uso della modalità *preflight*, perciò in `to.xsl` ho inserito questo codice:

```
<xsl:template match="tei:body">
    <xsl:copy>
        <xsl:apply-templates select="@*|node()"/>
    </xsl:copy>
    <xsl:element name="back" namespace="http://www.tei-c.org/ns/1.0">
        <xsl:element name="div" namespace="[come sopra]">
            <xsl:element name="head" namespace="[come sopra]">
                <xsl:element name="p" namespace="[come sopra]">
                    <xsl:element name="h1" namespace="[come
sopra]">Indice dei luoghi:</xsl:element>
                </xsl:element>
            </xsl:element>
            <xsl:for-each select="//tei:rs[@type='luogo']">
                <xsl:element name="ul" namespace="[come sopra]">
                    <xsl:value-of select="."/>
                </xsl:element>
            </xsl:for-each>
        </xsl:element>
    </xsl:element>
</xsl:template>
```

È stato indispensabile l'utilizzo del codice <xsl:element> per evitare la ripetizione nell'output del namespace, che avrebbe dato luogo a un documento HTML errato.

3.2.1 Difficoltà incontrate

Anche in questo caso l'output è stato inaspettato: l'indice dei nomi sperato, applicato al

documento di partenza, è stato inserito prima della porzione di testo centrale invece che dopo come avrebbe dovuto essere.


Il pellegrino nell'Asia

Table of contents

1. DEL VIAGGIO per L'INDIA. LIBRO TERZO.
   1.1. Della città di Surat

Indice dei luoghi:

Persia
Candahar
India
Europa
Guzarate
Golfo di Cambaia

Questo perché, in questo caso, la lista creata viene inserita nel documento dell'ebook `index.html`, dove viene creato un una sezione <body> vuota prima di essa.

3.2.2 La soluzione

Sebbene l'output finale sia comunque soddisfacente per lo scopo preposto, ovvero quello di creare una lista dei luoghi menzionati, se l'intenzione è quella di stampare del testo dopo il corpo centrale dell'ebook, è una possibile soluzione a questo problema applicare una trasformazione XSLT prima di utilizzare la trasformazione tramite i fogli di stile TEI, che restituisca in output un documento XML, da poter sottoporre al foglio di stile EPUB senza ottenere errori. Ho quindi creato un foglio di stile XSLT contenente il codice inserito sopra e con aggiunto, prima di esso, questo codice:

```
<xsl:template match="@*|node()">
    <xsl:copy>
        <xsl:apply-templates select="@*|node()"/>
    </xsl:copy>
</xsl:template>
```

così che tutto il documento in input venga ricopiato in ogni sua parte. L'output ottenuto ha tutti gli elementi di marcatura presenti nel documento XML di partenza, più il codice

per la creazione dell'indice in fondo al testo.
Dopo aver applicato la trasformazione al documento XML così ottenuto, il risultato sarà questo:

L'indice dei luoghi è stato quindi inserito nel punto previsto in partenza, ed è stato aggiunto un collegamento alla relativa tabella dei contenuti.

3.3 Resa grafica dell'ebook in formato EPUB più gradevole

L'output ottenuto da un documento XML in input che non prevede una copertina presenta di default, intorno al titolo, un box contornato di rosso che sfocia, a seconda della lunghezza del titolo, nella pagina successiva a quella destinata alla copertina.
Per rendere il risultato finale più gradevole all'utente, ho nuovamente chiesto un parere alla mailing list TEI. Mi è stato risposto che è possibile cambiare questo aspetto nel foglio di stile `tei-to-epub.xsl`:

```
<xsl:when test="$coverImageInside=''">
    <div>
        <xsl:attribute name="style">
            font-family: serif; height:auto; font-size:30pt; font-
weight: bold; padding-top: 15pt; margin: 12pt; border: solid red 1pt
text-align:center;
            </xsl:attribute>
        <xsl:call-template name="generateTitle"/>
    </div>
</xsl:when>
<xsl:otherwise>
    <div>
        <img width="1200" height="1700" alt="cover picture"
src="{$coverImageInside}"/> </div>
```

```
</xsl:otherwise> </xsl:choose> </body> </html>
```

Il codice presentato fa parte del template <processTEI>. Quando non viene trovata un'immagine di copertina, viene applicato un certo codice CSS intorno al titolo, altrimenti viene utilizzata l'immagine di copertina trovata. In questo caso, per migliorare l'aspetto grafico dell'ebook creato, è stato inserito questo codice, che elimina il margine rosso, e crea un riquadro di colore grigio, con all'interno la scritta "Copertina non pervenuta":

```
<xsl:when test="$coverImageInside=''">
    <div>
        <xsl:attribute name="style"> font-family: serif; height:auto;
font-size:30pt; font-weight: bold; padding-top: 15pt; margin: 12pt;
text-align:center; </xsl:attribute>
        <xsl:call-template name="generateTitle"/>
        <div style="position: Absolute; background-color: #DDDDDD;
top: 20%; height:15%; width:50%; margin: 0px, 0px, 0px, 0px; font-
size:10pt;"> Copertina non pervenuta</div>
    </div>
</xsl:when>
```

Il risultato ottenuto in questo caso è:

È ovviamente possibile inserire immagini e intervenire su altre porzioni di codice, in modo tale da modificare la resa grafica di porzioni di testo a propria scelta[10].

10 Nota: Sebastian Rathz ha deciso di cambiare la gestione del CSS, e verrà rilasciata presto una versione differente. Verrà introdotto un foglio di stile chiamato tei.css, e la gestione del titolo senza copertina verrà definita dal codice:

```
div.EpubCoverPage { font-family: serif; height:860; font-size:30pt;
```

Per inserire un'immagine, al posto del codice CSS, è stato utilizzato questo codice, che ho inserito subito dopo il comando di chiamata per il template che genera il titolo dell'ebook:

```
<img width="1200" height="1700" alt="cover picture"
src="http://i49.tinypic.com/r0qhs4.jpg"/>
```

Il risultato ottenuto è stato questo:

Copertina non pervenuta

---

font-weight: bold; padding-top: 15pt; margin: 12pt; border: solid red 1pt; text-align:center; }

4 Il passaggio finale

Dopo aver personalizzato il proprio output, e creato così un ebook in formato EPUB completo, è buona norma, se l'intento è la distribuzione, ma anche per verificare se vi sono modifiche da aggiungere, validare l'ebook creato, utilizzando un sito web come http://validator.idpf.org/. Raggiungendo questo indirizzo, è possibile immettere in questo sito il proprio ebook finale, e viene restituito all'utente in risposta quali sono gli elementi che non risultano standard nel proprio prodotto finale. Il sito in questione, in ogni caso, consiglia di installare il programma “EpubCheck” se l'obiettivo finale è la commercializzazione.

5 Conclusione

Nel corso della tesi sono state mostrate le caratteristiche di un ebook in formato EPUB, l'utilizzo della trasformazione TEI da un documento XML a un ebook in formato EPUB, tramite l'utilizzo del programma Oxygen, e sono stati illustrati i vari fogli di stile necessari a compiere queste trasformazioni. È stato inoltre mostrato come effettuare delle modifiche su questi fogli di stile, i vari problemi riscontrati nel momento della modifica di questi e le varie soluzioni trovate per poter attuare le personalizzazioni desiderate.

In eventuali sviluppi futuri, un utente interessato a creare ebook in formato EPUB o alla trasformazione da XML a EPUB potrà prendere visione di questo studio per comprendere in modo migliore come personalizzare gli ebook creati tramite questi fogli di stile per i propri scopi o studi. Uno studio interessante potrebbe anche riguardare il bug che ha impedito l'utilizzo della modalità *preflight* nella creazione dell'indice dei nomi di luogo in fondo alla pagina, e contemplare varie soluzioni alternative.

6 Bibliografia e sitografia

- Luca Ferrieri, *L'ebook in biblioteca: una sfida culturale*, in "Biblioteche Oggi", vol. XXVIII, n. 7, settembre 2010, pp. 5 – 14.
- Ivan Rachieli, *La pratica dell'ePub: quando il libro diventa software*, Apogeo, 2011.
- Massimo Canducci, *XML Pocket: conoscere il linguaggio XML significa poter comunicare veramente con tutti*, Apogeo, 2005.
- Documento del W3C sull'**XSLT 2.0** (23 gennaio 2007): http://www.w3.org/TR/xslt20/
- Traduzione in italiano della Raccomandazione W3C del 12 Maggio 1998 "Cascading Style Sheets, level 2 CSS2 Specification" (2006): http://www.diodati.org/w3c/css2/cover.html
- Dal sito guidaebook, informazioni sulla struttura di un ebook in formato EPUB: http://www.guidaebook.com/guida-epub/
- Da sito del Programma Oxygen, informazioni sul funzionamento del programma in generale: http://www.oxygenxml.com/
- Dal sito del Programma Oxygen, informazioni sulle trasformazioni EPUB: http://oxygenxml.com/epub.html
- Dal sito W3schools, lezioni e approfondimenti sull'utilizzo del linguaggio XSLT: http://www.w3schools.com/xsl/
- Dal sito dell'organizzazione TEI, informazioni generali sulle linee guida, gli obiettivi generali dell'organizzazione e i vari moduli: http://www.tei-c.org/index.xml.
- Tabella dei contenuti delle linee guida TEI: http://www.tei-c.org/release/doc/tei-p5-doc/en/html/index-toc.html
- Tabella dei contenuti utilizzati per la trasformazione EPUB: http://www.tei-c.org/release/doc/tei-xsl-common2/profiles/default/epub/to.html
- TEI Consortium, eds. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. 2.1.0. Ultima modifica il 17th Giugno 2012. TEI Consortium. http://www.tei-c.org/Guidelines/P5/