The DocBook Demystification HOWTO

Eric Raymond

esr@thyrsus.com

Questo HOWTO cercherà di dissolvere la nebbia e il mistero che circondano il sistema di markup DocBook e gli strumenti che ne fanno uso. È rivolto ad autori di documentazione tecnica di progetti open-source sotto Linux, ma dovrebbe essere utile anche a quanti producono altri generi sotto altri sistemi Unix.

Traduzione a cura di Luca Ferraro <`luca.ferraro(at)caspur.it`> e Giuseppe Briotti <`g.briotti(at)mclink.it`>, revisione a cura di Sebastiano Gazzola <`info(at)sebastianogazzola.it`>.

Sommario

1. Introduzione

Gran parte dei progetti open-source stanno convergendo sull'uso di DocBook come formato standard per la loro documentazione — come ad esempio i progetti Kernel Linux, GNOME, KDE, Samba e lo stesso Linux Documentation Project. I sostenitori del linguaggio "structural markup" basato su XML (opposto al vecchio "presentation markup" usato ad esempio da Troff, TeX e Texinfo) sembrano aver vinto la battaglia. Si può generare un "presentation markup" da un "structural markup", ma fare il contrario è ancora molto difficile.

Nonostante questo, c'è una grande confusione riguardo a DocBook ed ai programmi che lo gestiscono. I suoi fedeli devoti parlano un gergo vasto e inaccessibile anche per gli standard linguistici dell'informatica, sputando acronimi che sembrano aver nulla a che vedere con quello che serve per scrivere documenti in markup language e ottenere da questi pagine HTML o Postscript. Gli standard e la documentazione tecnica XML sono notoriamente molto oscuri.

Questo HOWTO cercherà di far luce su gran parte dei misteri del formato DocBook e sul suo impiego per la produzione di documentazione open-source, considerando sia gli aspetti tecnici che quelli politici. Il nostro obiettivo è quello di rendervi in grado di capire non solo ciò di cui avrete bisogno per creare dei documenti, ma anche perché il processo di creazione è così complesso e cosa c'è da aspettarsi possa cambiare con l'arrivo di nuovi strumenti DocBook.

2. Perché interessarsi di DocBook?

Ci sono due caratteristiche che rendono DocBook estremamente interessante: la prima è il *multi-mode rendering*, la seconda è il *searchable documentation databases*.

Multi-mode rendering è il termine inglese più semplice per spiegare la cosa; in altre parole si tratta della possibilità di scrivere un documento in un unico formato che possa poi essere rielaborato per diverse modalità di visualizzazione (in particolare, sia per pagine HTML ottimizzate per la consultazione on-line, sia per pagine Postscript ad alta qualità di stampa). Questa caratteristica è già stata implementata molto bene in DocBook.

Searchable documentation databases è il termine col quale indichiamo la potenzialità che DocBook ha di portarci verso un mondo in cui tutta la documentazione del nostro sistema operativo open-source sarà raccolta in un unico e ricco database, consultabile con indici incrociati e hyperlink (invece di essere sparpagliata in vari luoghi e formati come lo è ora).

Idealmente, non appena installate un pacchetto software sulla vostra macchina, dovrebbe essere registrata la documentazione DocBook nel vostro catalogo di sistema. Potrebbero così venire aggiunte delle pagine di HTML indicizzate e propriamente collegate al resto della documentazione HTML già esistente sul vostro catalogo. La documentazione del nuovo pacchetto potrebbe quindi essere accessibile direttamente dal vostro browser e consultabile attraverso una interfaccia molto simile a quella di un buon motore di ricerca Web.

L'HTML da solo non è tuttavia un formato sufficientemente ricco e adatto per realizzare una cosa simile. Giusto per citare una sua mancanza, in HTML non potete dichiarare esplicitamente degli elementi di tipo indice. DocBook invece *ha* la ricchezza semantica per la creazione di documenti strutturati come database. Questo, fondamentalmente, è il motivo per cui molti progetti lo stanno adottando.

Ma insieme ai suoi pregi, DocBook ha anche i suoi difetti. Molte persone lo trovano sgradevolmente pesante, eccessivamente prolisso per essere considerato un agevole formato di composizione. Fin tanto che gli strumenti di markup (come il Perl POD o GNU Texinfo) riescono a generare DocBook, possiamo ritenerci tutti soddisfatti. Non ha importanza se non tutti scrivono in DocBook: quando diverrà il formato comune di interscambio adottato da tutti, si riuscirà comunque a realizzare una base dati unificata per la documentazione.

3. L'Abc dello structural markup language

I vecchi programmi a linguaggio di formattazione come Tex, Texinfo, e Troff utilizzavano il *presentation markup*. In questi sistemi, le istruzioni inserite nel codice riguardavano la disposizione e la visualizzazione fisica degli elementi del testo (i cambiamenti di font, le indentazioni e cose simili).

Il presentation markup era un linguaggio di formattazione adeguato fintanto che l'obiettivo era quello di stampare su un unico ambiente o tipologia di dispositivo di visualizzazione. Si incappa nei suoi limiti non appena si vuole formattare un documento in modo che (a) questo possa essere visualizzato su una più vasta gamma di dispositivi di output (stampanti, Web, ecc.) oppure (b) si voglia che il documento consenta la ricerca e l'indicizzazione basate sulla sua struttura logica (come vorreste fare se, ad esempio, decideste di inserirlo in un sistema ipertestuale).

Per essere in grado di fornire queste capacità, bisogna utilizzare un sistema di tipo *structural markup*. Con lo "structural markup" non si descrive il layout fisico del documento, bensì le proprietà logiche delle sue singole parti.

Un esempio: in un linguaggio di tipo presentation markup, se volete enfatizzare una parola, dovrete istruire il formattatore per renderla in grassetto. In troff(1) dovreste scrivere qualcosa del genere:

```
All your base
.B are
belong to us!
```

In un linguaggio di tipo structural markup, quello che direste al formattatore è semplicemente di evidenziare la parola:

```
All your base <emphasis>are</emphasis> belong to us!
```

I comandi "<emphasis>" e "</emphasis>" della riga sopra sono chiamati *markup tags* o, semplicemente, *tags* [NdT: in italiano "marcatori"]. Queste sono le istruzioni per il vostro programma di formattazione.

In un linguaggio structural markup, la formattazione finale del documento viene controllata da un foglio di stile [NdT: in inglese *stylesheet*]. È il foglio di stile che dirà al vostro programma di formattazione "visualizza gli emphasis con il carattere grassetto". Un vantaggio del linguaggio "structural markup" è che cambiando il foglio di stile cambiate la visualizzazione totale del documento (per usare font diversi, ad esempio) senza dover mettere mano ad ogni singola istruzione (come .B nell'esempio precedente) nel codice del documento.

4. Definizioni Tipologia Documento (Document Type Definitions)

(Nota: per mantenere semplice la spiegazione, in questa sezione dirò qualche bugia, principalmente tralasciando un sacco di roba. La verità verrà ripristinata in una della sezioni successive.)

DocBook è un linguaggio di tipo structural-level markup. Più precisamente, è un dialetto di XML. Un documento DocBook è quindi un pezzo di XML che usa tag XML per le istruzioni structural markup.

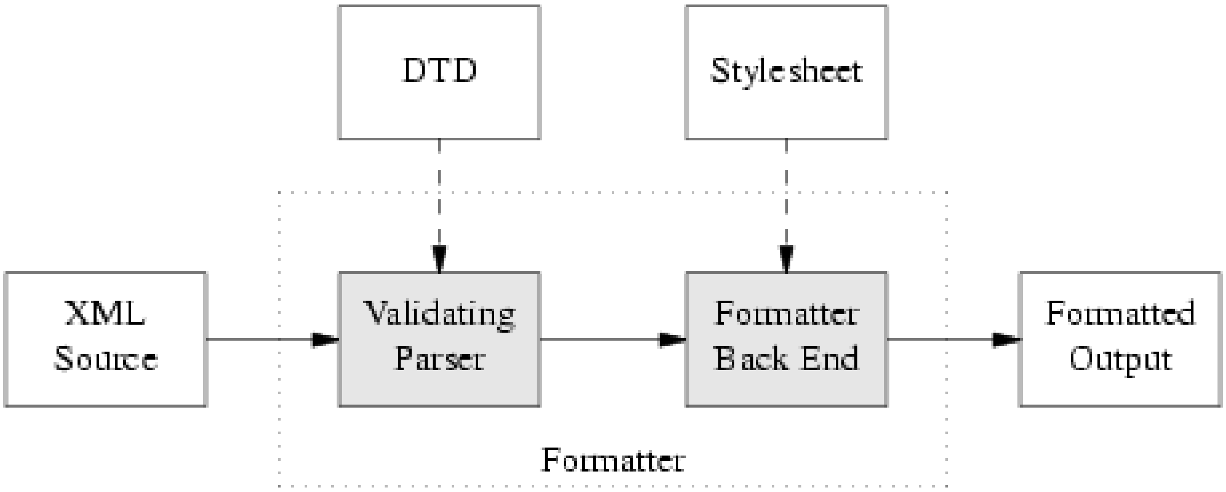
Affinché un programma per la composizione ed impaginazione dei documenti (document formatter) applichi un foglio di stile al vostro documento per renderlo di bell'aspetto, deve sapere alcune cose sulla struttura generale del vostro documento. Per esempio, dovrà sapere che un manoscritto generalmente consiste in una prima di copertina, una sequenza di capitoli e un'ultima di copertina, in modo da comporre fisicamente i titoli dei capitoli in modo appropriato. Per fargli sapere tutte queste cose, dovrete fornire al compositore di documento una Definizione del Tipo di Documento o *Document Type Definition*() (DTD). Il DTD informa il compositore sul tipo di elementi che può contenere il vostro documento e in quale ordine possono apparire.

Quando diciamo che DocBook è una "applicazione" (una realizzazione) di XML intendiamo dire che DocBook è un DTD XML, molto esteso, che comprende quasi 400 tag.

Dietro DocBook c'è un tipo di programma chiamato *validating parser*. Quando componete un documento di tipo DocBook, il primo passo da fare è quello di darlo in pasto a un validating parser (il front-end di un compositore DocBook). Questo programma controlla il vostro documento in base al DocBook DTD per assicurarsi che non abbiate violato le regole e le strutture del DTD (in caso contrario, il software di composizione potrebbe confondersi al momento di applicare il foglio di stile).

Il validating parser potrà darvi dei messaggi di errore, indicando in quali punti del documento non è corretta la struttura, oppure tradurre il documento in un flusso di *formatting events* che la parte finale del parser combina con le informazioni del vostro stylesheet per produrre un output formattato correttamente.

Ecco un diagramma dell'intero processo:



La parte del diagramma all'interno del rettangolo punteggiato rappresenta il vostro software di composizione, realizzato con una serie di strumenti in sequenza (*toolchain*). Bisogna tenere a mente, per capire ciò che seguirà, che vengono passati al compositore, oltre all'ovvio e ben visibile sorgente del documento, anche due input invisibili: il DTD ed il foglio di stile.

5. Altri DTD

Una breve digressione su altri DTD può aiutare a chiarire quali parti dei precedenti paragrafi riguardano esclusivamente DocBook e quali sono invece più generali e comuni a tutti i structural-markup language.

TEI (http://www.tei-c.org/) (Text Encoding Initiative) è un ampio ed elaborato DTD usato principalmente nelle università per la trascrizioni di testi letterari in formato elettronico. La toolchain Unix di TEI utilizza molti degli strumenti utilizzati da DocBook, ma con stylesheet differenti e (ovviamente) con un DTD diverso.

XHTML, l'ultima versione di HTML, è anch'essa una applicazione XML descritta da un DTD, che esprime la somiglianza tra XHTML e i tag di DocBook. La toolchain di XHTML consiste in un web-browser e un certo numero di utilità di stampa HTML costruite ad-hoc.

Molti altri DTD XML sono disponibili per aiutare la gente a scambiare documenti strutturati in diversi campi, dalla bioinformatica alla gestione bancaria. Potete guardare nella lista di repository (http://www.xml.com/pub/rg/DTD_Repositories) per farvi un'idea della loro varietà.

6. La toolchain di DocBook

Il modo più semplice per impaginare documenti XML-DocBook è utilizzare gli strumenti di xmlto. Questi strumenti sono forniti con Red Hat; gli utenti di Debian possono ottenerli con il comando **apt-get install xmlto**.

Normalmente, quello che dovreste fare per creare un documento XHTML da un sorgente DocBook è qualcosa di simile:

```
bash$ xmlto xhtml foo.xml
bash$ ls *.html
ar01s02.html ar01s03.html ar01s04.html index.html
```

In questo esempio abbiamo trasformato un documento XML-DocBook chiamato `foo.xml`, composto da tre sezioni principali, in una pagina di indice e altre due parti. Ma mettere tutto in una sola pagina è altrettanto facile:

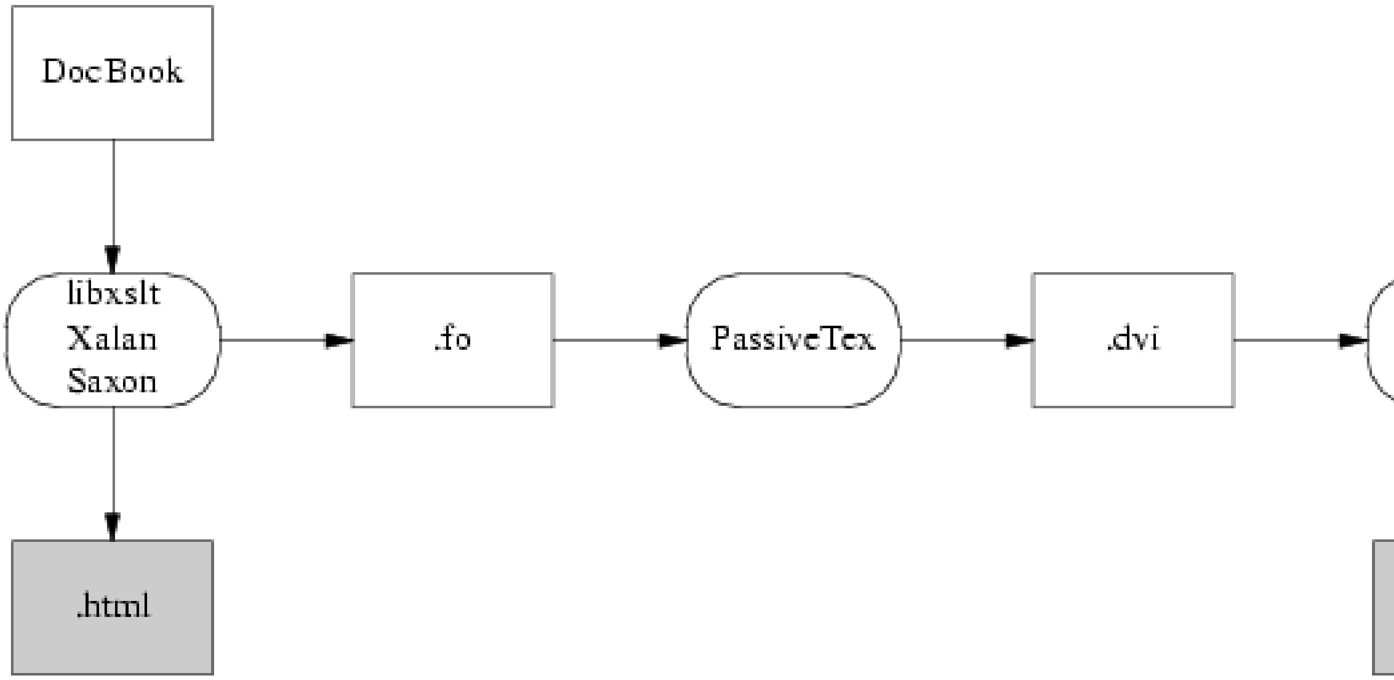
```
bash$ xmlto xhtml-nochunks foo.xml
bash$ ls *.html
foo.html
```

Infine, ecco come fare per creare un documento Postscript:

```
bash$ xmlto ps foo.xml       # To make Postscript
bash$ ls *.ps
foo.ps
```

Alcune vecchie versioni di **xmlto** potrebbero essere più verbose, riportando messaggi quali "Coverting to XHTML" e così via.

Per trasformare i vostri documenti in HTML o Postscript avrete bisogno di uno strumento che possa applicare al vostro documento le direttive di DocBook DTD e del foglio di stile adatto. Ora mostreremo come combinare gli strumenti open-source per compiere questo lavoro:



Attuale catena XML-DocBook

L'analisi del documento e l'applicazione del foglio di stile possono essere realizzati tramite uno di questi tre programmi: xsltproc, il parser distribuito con la RedHat 7.3 e versioni successive, oppure Saxon o Xalan, due programmi scritti in Java.

Generare un documento XHTML di alta qualità con DocBook è abbastanza semplice; il fatto poi che XHTML sia semplicemente un altro tipo di XML DTD è di grande aiuto. La traduzione in HTML si ottiene applicando un semplice stylesheet, e fine della storia. Generare un RTF in questo modo è altrettanto semplice e da un XHTML o RTF è facile generare un testo ASCII in un attimo.

Caso delicato è la stampa. È difficile generare un documento di alta qualità per la stampa (che in pratica significa un documento tipo Adobe o Portable Document Format, una forma compressa di PostScript). Farlo bene comporta dover automatizzare il giudizio attento di un tipografo nel passaggio dal livello di contenuto a quello di presentazione.

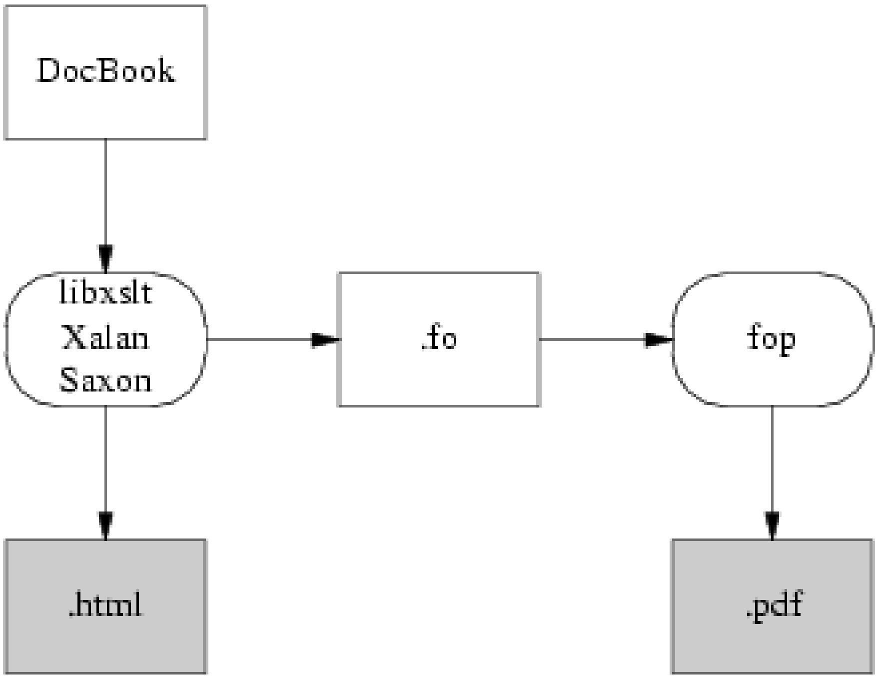
Prima di tutto, uno stylesheet trasforma e traduce un documento strutturato DocBook DTD in un altro documento scritto in un dialetto di XML, chiamato FO (Formatting Objects). Il markup di FO è molto più vicino ad una struttura tipo presentation-level; potete pensarla come ad una sorta di XML funzionalmente equivalente a Troff (NdT. la struttura delle pagine man per intenderci). Subito dopo deve essere trasformato in Postscript per poi essere inpacchettato in un PDF.

Nella toolchain distribuita con RedHat, questo lavoro è fatto da una macro di TeX chiamata PassiveTeX. Questa traduce il documento in FO prodotto da **xsltproc** nel linguaggio TeX di Donald Knuth. TeX è stato uno dei primi progetti open-source, un vecchio ma potentissimi linguaggio di formattazione presentation-level molto amato dai matematici (ai quali fornisce una serie di strumenti particolarmente adatti per scrivere documenti contenenti formule matematiche). TeX è anche molto utile per compiti di composizione di base, quali il kerning, il riempimento di linee e la gestione delle ifenazioni. Il formato di output del TeX, chiamato DVI (DeVice Independent), viene poi trasformato in un PDF.

Se pensate che questa lunga e tortuosa catena che va da XML alle macro TeX, per poi passare al DVI ed infine al PDF sia delicatissimo, beh, avete perfettamente ragione: sferraglia, ansima ed ha delle orribili verruche. I font sono un grosso problema, dal momento che XML, TeX e PDF fanno uso di modelli molto diversi per la gestione dei font. Inoltre, la gestione della localizzazione e dell'internazionalizzazione è un vero incubo. Forse, l'unico punto a favore di questo percorso è che funziona.

Una soluzione più elegante sarebbe FOP, un traduttore che passa direttamente da FO a Postscript, sviluppato con il progetto Apache. Con FOP il problema dell'internazionalizzazione è, se non completamente risolto, almeno ben delimitato; gli strumenti XML utilizzano tutti l'Unicode attraverso FOP. La mappatura del singolo carattere nel font è anche un problema importante di FOP. L'unico problema in questo approccio è che non funziona, almeno per il momento. Ad Agosto 2002, il progetto FOP è ad uno stato alpha, quindi utilizzabile, ma ancora non rifinito e con diverse caratteristiche mancanti.

Ecco quello che fa la toolchain FOP:



Futura catena XML-DocBook con FOP.

FOP ha già della concorrenza. C’è un altro progetto, chiamato xsl-fo-proc, che aspira agli stessi obiettivi di FOP, ma è scritto in C++ (è quindi più veloce di Java ed è indipendente dall’ambiente Java). Nell’Agosto 2002, il progetto xsl-fo-proc era allo stato alpha, non molto distante da FOP.

7. Quali sono i progetti e gli sviluppatori?

Il DTD DocBook è mantenuto dal DocBook Technical Committee, diretto da Norman Walsh. Norman è l’autore principale degli stylesheet DocBook, un uomo che ha impiegato per diversi anni le sue energie ed il suo talento per risolvere i complessi problemi di realizzazione di DocBook. Nella comunità dei DocBook/SGML/XML gode della stessa considerazione e popolarità che Linus Torvald ha nel mondo di Linux.

libxslt (http://xmlsoft.org/XSLT/) è una libreria C che interpreta XSLT, applicando gli stylesheet ai documenti XML. La libreria include un wrapper, **xsltproc**, che può essere utilizzato come un formattatore XML. Il codice di libxslt è stato scritto da Daniel Veillard sotto i buoni auspici del progetto GNOME, ma non richiede GNOME per essere utilizzato. Ho sentito dire che è mostruosamente veloce rispetto alle alternative in Java, il che non mi sorprende affatto.

xmlto (http://cyberelk.net/tim/xmlto/) è l’interfaccia della toolchain XML fornita da RedHat. È scritta e mantenuta da Tim Waugh.

Saxon (http://users.iclway.co.uk/mhkay/saxon/) e Xalan (http://xml.apache.org/xalan-j/) sono dei programmi Java che interpretano XSLT. Saxon sembra esser stato pensato per girare sotto Windows. Xalan invece fa parte del progetto Apache XML, nativo per Linux e BSD; è stato progettato per lavorare con FOP.

FOP (http://xml.apache.org/fop/) è un programma che trasforma FO-XML in PDF. Fa parte del progetto Apache XML ed è pensato per poter lavorare con Xalan.

8. Migration tools

Il secondo grande problema con DocBook è dato dagli sforzi necessari per convertire il vecchio presentation markup nel nuovo markup di DocBook. Un essere umano non avrebbe alcuna difficoltà a riconoscere in un documento le sue diverse strutture logiche, questo perché può capire dal contesto quando, ad esempio, il corsivo è utilizzato per evidenziare qualcosa e quando invece serve a riportare una citazione del tipo 'questa è la frase di qualcuno'.

In qualche modo, quando vogliamo convertire i documenti in DocBook, queste distinzioni devono essere rese esplicite. Qualche volta sono già prensenti nel vecchio markup, ma spesso non lo sono, e la mancanza di questa informazione strutturale deve essere quindi dedotta da algoritmi euristici o introdotta a mano dall'uomo.

Di seguito è riportato un riepilogo sullo stato dei vari strumenti di conversione utilizzati per i vari formati:

GNU Texinfo

La Free Software Foundation ha deciso di utilizzare DocBook come un formato di interscambio. Texinfo ha strutture a sufficienza per realizzare conversioni automatizzate di buona qualità, e la versione 4.x di **makeinfo** mette a disposizione l'opzione `--docbook` che genera direttamente documenti in DocBook. Per avere maggiori informazioni, consultate la pagina del progetto makeinfo (http://www.gnu.org/directory/texinfo.html).

POD

Esiste un modulo POD::DocBook (http://www.cpan.org/modules/by-module/Pod/) che traduce i documenti scritti nel markup POD (Plain Old Documentation) in DocBook. Ufficialmente implementa tutti i tag POD eccetto il tag L<> italic. La pagina man dice anche "liste del tipo nested-over e nested-bask inserite nei documenti non sono gestite in Docbook", ma si noti che il modulo è stato ampiamente testato.

LaTeX

LaTeX è, principalmente, un insieme di macro in linguaggio structural markup costruito sul formatter TeX. Esiste un programma, chiamato TeX4ht (http://www.lrz-muenchen.de/services/software/sonstiges/tex4ht/mn.html), che riesce a generare DocBook a partire da documenti LaTeX (questo secondo l'autore di PassiveTeX).

Pagine man e altri markup tipo troff

Questo è generalemente considerato il più grosso e difficile problema di conversione. Infatti troff(1) è un markup di così basso livello di presentazione che è molto difficile ottenere da strumenti automatici una conversione dignitosa. Tuttavia, la visione così buia della cosa può essere schiarita pensando a sorgenti di documentazione scritti con le macro del pacchetto man(7). Questi infatti utilizzano caratteristiche di struttura sufficienti per ottenere conversioni in modo automatico.

Ho scritto un programma per fare proprio questo, anche perché non sono riuscito a trovare nient'altro che facesse un lavoro almeno decente (il problema è molto interessante). L'ho chiamato doclifter (http://www.catb.org/~esr//doclifter/). Dovrà riuscire a produrre DocBook XML o SGML a partire da sorgenti scritti per man(7), mdoc(7), ms(7), o me(7). Leggete la documentazione per maggiori informazioni.

9. Strumenti di Editing

Quello che al momento non abbiamo è un buon programma open-source per scrivere documenti SGML/XML.

Il progetto Conglomerate (http://conglomerate.org/) punta specificatamente a produrre un buon editor per DocBook. Agli inizi del 2004 è ancora allo stato alfa.

Il progetto MlView (http://www.freespiders.org/projects/gmlview/) è un editor XML, non indirizzato specificatamente a DocBook. Agli inizi del 2004 è carente di documentazione e sembra essere ancora allo stato alfa.

LyX (http://www.lyx.org/) è un word processor tipo GUI che utilizza LaTex per la stampa e consente le modifiche strutturali di LaTeX markup. Esiste un pacchetto LaTeX che genera documenti DocBook, e un how-to document (http://bgu.chez.tiscali.fr/doc/db4lyx/) che descrive come redigere documenti SGML e XML con LyX .

GeTox (http://idx-getox.idealx.org/), l'editor XML di GNOME, è rivolto a utenti non-tecnici, ma il software (Agosto 2001) è ancora in fase alpha di sviluppo: poco più che una serie di concetti che qualcosa di utile, inoltre il gruppo non sembra essere troppo attivo sul progetto. Non ci sono stati aggiornamenti sul sito da Marzo 2001 e ora siamo ad Agosto 2002.

GNU TeXMacs (http://www.math.u-psud.fr/~anh/TeXmacs/TeXmacs.html) è un progetto rivolto alla realizzazione di un editor orientato alla produzione di materiale tecnico e matematico, inclusa la gestione delle formule. In Aprile 2002 è stata rilasciata la versione 1.0. Gli sviluppatori pensano di poter implementare XML in futuro, ma per ora non è gestito.

ThotBook (http://www.freesoftware.fsf.org/thotbook/) è un progetto che vuol mettere insieme un editor GUI per DocBook e il toolkit Thot. Ma sembra ormai morto: la pagina web non è aggiornata da Novembre 2001 e ora siamo ad Agosto 2002.

Tutt'oggi, la maggior parte della persone deve modificare a mano i tag con Vi o con Emacs e usare psgml per controllare la validità delle modifiche.

10. Suggerimenti e trucchi

È possibile creare un indice inserendo un tag <index/> vuoto nel punto del documento dove si vuole che appaia. Si faccia attenzione al fatto che, agli inizi del 2004, questo strumento è in qualche modo ancora primitivo. Non riunisce i blocchi selezionati e l'output prodotto per PostScript non è ancora di alta qualità.

Questo spazio è riservato per ulteriori suggerimenti e trucchi.

11. Usi e standard correlati

Gli strumenti per scrivere e maneggiare documenti DocBook stanno, anche se lentamente, venendo fuori. Ma lo stesso progetto DocBook rappresenta solo una tappa intermedia, non il traguardo. Avremo bisogno di altri standard, oltre a quello di DocBook, per raggiungere l'obiettivo di un searchable-documentation-database di cui ho parlato all'inizio di questo documento. Ci sono ancora due grandi questioni da affrontare: la catalogazione dei documenti e i metadata.

Il progetto Scrollkeeper (http://scrollkeeper.sourceforge.net/) punta a raggiungere questi scopi. Mette a disposizione una serie di semplici script che possono essere usati durante l'installazione o la disinstallazione dei pacchetti per registrare o eliminare la relativa documentazione da un database che consenta ricerche e sia condiviso da tutto il sistema.

Scrollkeeper utilizza Open Metadata Format (http://www.ibiblio.org/osrt/omf/), una struttura standard per l'indicizzazione della documentazione open-source, molto simile a quella usata nei cataloghi a schede delle biblioteche. L'idea è quella di fornire un servizio di ricerca basato sia sui metadati delle schede sia sul testo stesso contenuto nella documentazione.

12. SGML and SGML-Tools

Nei paragrafi precedenti, ho tralasciato gran parte della storia su DocBook. XML infatti ha un fratello maggiore, chiamato SGML o Standard Generalized Markup Language.

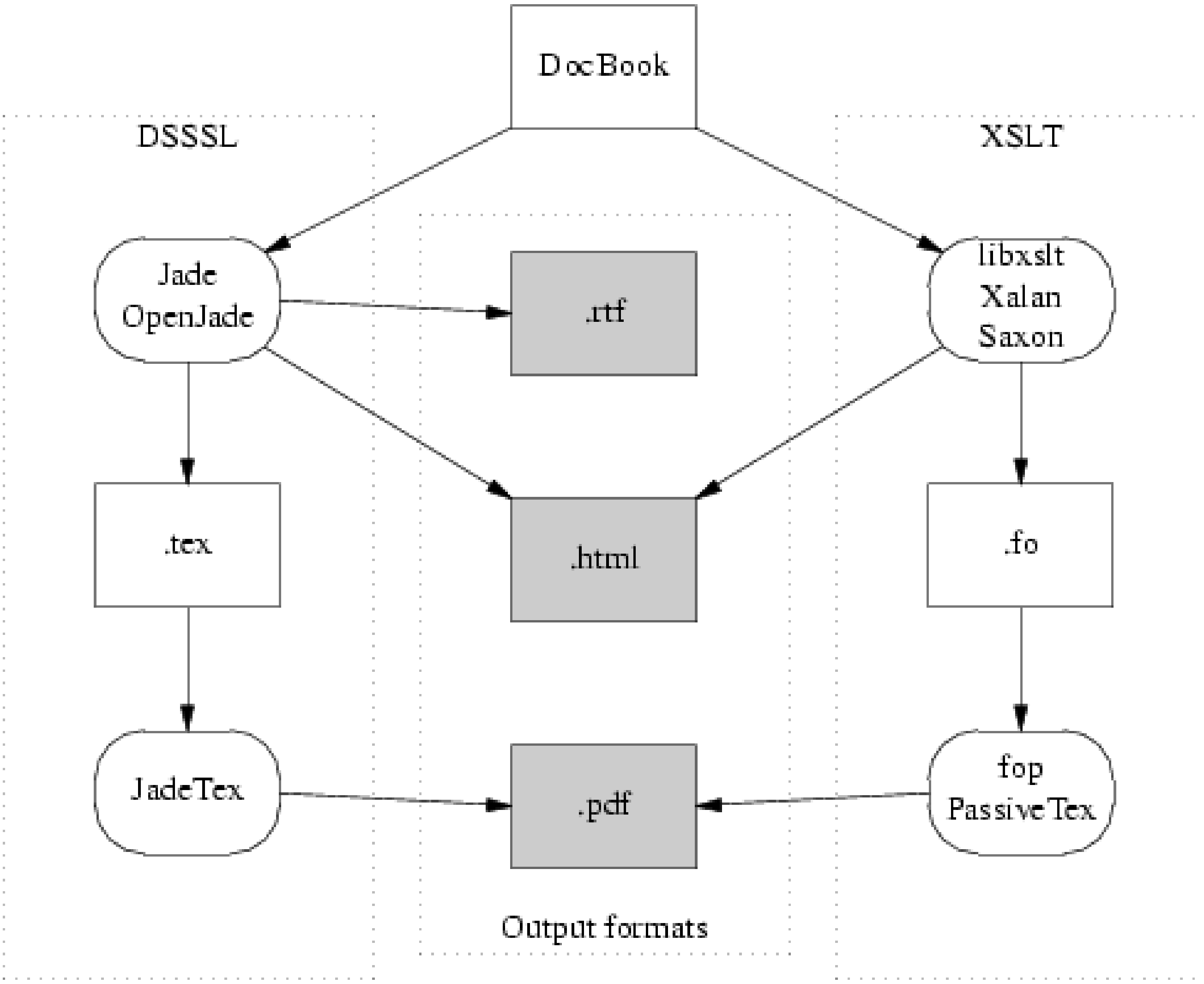
Fino a metà del 2002, nessuna discussione su DocBook sarebbe stata completa senza una lunga escursione su SGML, le differenze tra SGML e XML e una accurata descrizione della toolchain SGML DocBook. Ora la vita è molto più semplice; una toolchain XML DocBook è ora disponibile come open-source, funziona bene come la toolchain di SGML ed è molto più facile da usare. Se pensate che non avrete mai a che fare con vecchi documenti redatti in SGML DocBook, potete tranquillamente saltare il resto di questa sezione.

12.1. DocBook SGML

Originariamente DocBook nasce come una applicazione SGML, e c'era anche un progetto di toolchain SGML DocBook, che ora è praticamente morto. Ci sono alcune piccole differenze tra i DTD DocBook SGML e i DTD DocBook XML, ma per una discussione introduttiva possiamo tranquillamente ingorarle. L'unica differenza evidente all'utente è che i tag SGML senza contenuto non hanno bisogno del trailing slash (/) in chiusura del tag. (L'utilizzo del trailing slash in XML rende il lavoro del parser molto più facile perché non c'è bisogno che il programma conosca il DTD e quindi quali tag aperti hanno bisogno di chiusura.)

Tutte le versioni di HTML fino alla 4.01 (quindi antecedenti a XHTML) erano esse stesse applicazioni di SGML. Anche TEI era nato originariamente come applicazione SGML. I gruppi che lavoravano su questi tre DTD sono passati all'XML per lo stesso motivo degli sviluppatori di DocBook: è decisamente molto più semplice. L'SGML era estremamente complesso, sostanzialmente ingestibile. Le specifiche sono raccolte in ben 150 pagine dense di regole e non c'è ancora un software che le abbia mai implementate tutte.

La toolchain che avevo mostrato prima era molto semplificato: mostrava solo la toolchain di XML. Ecco qui riportata la versione storicamente corretta:



La toolchain DSSSL è quella che elabora i documenti SGML DocBook. È sotto il suo controllo che un documento in formato SGML passa attraverso uno o due motori per i fogli di stile, Jade e OpenJade. Questi lo trasformano in un formato tipo TeX-macro markup, che verrà successivamente elaborato dal pacchetto JadeTeX per passarlo in DVI e successivamente in Postscript.

12.2. SGML tools

Il progetto docbook-tools (http://sources.redhat.com/docbook-tools/) fornisce strumenti in open-source per convertire DocBook SGML in HTML, Postscript ed altri formati. Questo pacchetto è fornito con Red Hat ed altre distribuzioni Linux. È mantenuto da Mark Galassi.

Jade (http://www.jclark.com/jade/) è un motore utilizzato per applicare fogli di stile DSSSL a documenti SGML. È mantenuto da James Clark.

OpenJade (http://openjade.sourceforge.net/) è un progetto comunitario iniziato poiché i fondatori ritenevano che la manutenzione di Jade da parte di James Clark fosse troppo saltuaria. I programmi di docbook-tools utilizzano OpenJade.

PassiveTeX (http://users.ox.ac.uk/~rahtz/passivetex/) è il pacchetto di macro di LaTeX che xmlto utilizza per produrre DVI da XML-DocBook. JadeTex (http://jadetex.sourceforge.net/) è il pacchetto di macro di LaTeX che OpenJade utilizza per produrre DVI da SGML-DocBook.

12.3. Ecco perché l'SGML DocBook è morto

La toolchain DSSSL è praticamente morta, a meno di nuovi sviluppi futuri. La toolchain XSLT ha raggiunto lo stato di produzione a metà del 2002; una versione operativa è stata distribuita con la RedHat 7.3. È su questo che gli sviluppatori di Docbook stanno spendendo la maggior parte dei loro sforzi.

Sono tre le ragioni per cui si è passati a XML. Primo, ci si è resi conto che l'SGML è troppo complicato da usare; secondo, il DSSSL ha finito per essere insostenibile da utilizzare; terzo, alcune parti fondamentali della toolchain DSSL si sono rivelate troppo deboli e irrimediabilmente ingarbugliate.

Rispetto all'SGML, l'XML ha un insieme di funzioni più ridotto, ma sufficiente per la maggior parte dei suoi scopi e sicuramente più facili sia da capire e sia da implementare nei parser che le interpretino. Gli strumenti di elaborazione SGML (come ad esempio i validatori parser) dovevano contenere regole per la gestione di numerose caratteristiche che i DocBook e gli altri sistemi di testo a markup non utilizzano affatto. La rimozione di tutte queste caratteristiche supplementari ha reso molto più semplice l'XML e molto più veloci i relativi strumenti di elaborazione.

Il linguaggio utilizzato per dedscrivere i DTD SGML è talmente spinoso e proibitivo da rendere la scrittura di un DTD SGML una sorta di arte oscura. I DTD XML invece possono essere descritti in un dialetto di XML stesso. Non c'è quindi alcun bisogno di avere un linguaggio diverso per scrivere DTD. Una descrizione XML di un DTD XML è detta *schema*. Il termine DTD verrà probabilemente soppiantato proprio da quest ultimo, col modificarsi degli standard.

Al di là di tutto, dobbiamo dire che la toolchain DSSSL è morta proprio perché il DSSSL e il linguaggio di descrizione dello stylesheet SGML della toolchain, sono troppo arcani per molti, e ha reso i fogli di stile troppo difficili da scrivere e modificare. (Era un dialetto Scheme. Il vostro autore, un conoscitore di LISP da lungo tempo, ha scosso la testa in un attimo di stupore, al pensiero che questo potesse allontanare la gente)

I fan di XML amano riassumere tutto questo con una frase: "XML: è più buono e appesantisce meno".

12.4. SGML-Tools

SGML-Tools era il nome di un DTD utilizzato dal Linux Documentation Project (http://www.linuxdoc.org), sviluppato quale anno fa quando le attuali toolchain DocBook non esistevano. Il linguaggio SGML-Tools markup era più semplice di DocBook, ma anche meno flessibile. La toolchain formatter/DTD/stylesheet originale di SGML-Tools è stata abbandonata per molto tempo, ma il suo successore SGML-tools Lite (http://sourceforge.net/projects/sgmltools-lite/) è ancora mantenuto.

Il LDP sta abbandonando l'SGML-Tools in favore di DocBook, ma è possibile ancora incappare in qualche vecchio HOWTO. Questi possono essere riconosciuti dalla voce nell'intestazione "<!doctype linuxdoc system>". Se vi capita tra le mani, convertitelo in un XML DocBook e date alla vecchia versione una veloce sepoltura.

13. Riferimenti

Una delle cose che rende difficile imparare DocBook è che i siti che lo trattano tendono a sommergere il principiante con lunghe liste di standard W3C, dosi massicce di esercizi teologici sul markup ed una densa foresta di terminologia astratta. Cercheremo di evitare questa trafila dandovi qualche riferimento selezionato:

Take My Advice: Don't Learn XML (http://xml.oreilly.com/news/dontlearn_0701.html) di Michael Smith analizza il mondo dell' XML da un punto di vista molto simile a quello presentato in questo documento.

DocBook: The Definitive Guide di Norman Walsh è disponibile a stampa (http://www.oreilly.com/catalog/docbook/) e sul web (http://www.docbook.org/tdg/en/html/docbook.html).

Questo dovrebbe essere il documento fondamentale, ma se letto come tutorial è un disastro. Leggetevi dunque prima:

Writing Documentation Using DocBook: A Crash Course (http://www.bureau-cornavin.com/opensource/crash-course/index.html). Questo è un tutorial eccellente.

Esiste anche una ottima DocBook FAQ (http://www.dpawson.co.uk/docbook/) con molto materiale sui formati di stile per formati HTML. C'è anche un Docbook wiki (http://docbook.org/wiki/moin.cgi).

Se state scrivendo documentazione per il Linux Documentation Project, allora leggetevi la LDP Author Guide (http://www.linuxdoc.org/LDP/LDP-Author-Guide/index.html).

La migliore guida introduttiva a SGML e XML, che ho personalmente letto, è quella di David Megginson: Structuring XML Documents (http://vig.pearsoned.com/store/product/0,,store-562_banner-0_isbn-0136422993,00.html) (Prentice-Hall, ISBN: 0-13-642299-3).

Per il solo XML invece, XML In A Nutshell (http://www.oreilly.com/catalog/xmlnut2/) di W. Scott Means e Elliotte "Rusty" Harold è molto buona.

The XML Bible (http://www.ibiblio.org/xml/books/bible/) assomiglia ad una guida di riferimento omnicomprensiva sull' XML e sui relativi standard (compreso Formatting Objects).

Infine, se proprio siete degli avventurosi, The XML Cover Pages (http://xml.coverpages.org/) vi farà fare un viaggio nella giungla degli standard XML.