Diseño de un procedimiento de marcado para la automatización del procesamiento de textos digitales usando XML y TEI

Alejandro Bia y Manuel Sánchez-Quero

Biblioteca Virtual Miguel de Cervantes,
Universidad de Alicante,
Apdo. de correos 99, E-03080, Alicante, España
{alexbia, manuel.sanchez}@cervantesvirtual.com
http://cervantesvirtual.com/

Abstract. En este artículo se pretende explicar la investigación llevada a cabo por la Biblioteca Virtual Miguel de Cervantes [1] en el desarrollo de una política de marcado de textos literarios y la automatización de su procesamiento. También se intenta explicar y argumentar algunas de las decisiones de diseño tomadas y las soluciones de compromiso a que hemos llegado. Entre estas decisiones, cabe destacar la elección del lenguaje de marcado y la política de marcado.

1 Introducción

"Una biblioteca digital es la combinación de una colección de objetos digitales (repositorio); las descripciones de esos objetos (metadatos); un conjunto de usuarios (clientes o público o usuarios); y sistemas que ofrecen una serie de servicios como la captura, indexación, catalogación, búsqueda, navegación, recuperación, entrega, archivo y preservación." [1]

Los pasos iniciales en la creación de un proyecto de biblioteca digital consisten, desde un punto de vista global, en la definición de los métodos de la producción que conformarán el flujo de trabajo (workflow, ver fig. 1) y, desde una perspectiva de más bajo nivel, en la selección de la tecnología, formatos y herramientas para procesar los textos e imágenes. Este artículo trata de esto último, dando especial énfasis a la tecnología de marcado. En palabras de Sperberg-McQueen y Huitfeldt, "parecen generalmente estar de acuerdo aquellos que emplean ordenadores para los problemas de la investigación humanística en que un lenguaje de marcado bien diseñado es fundamental en la tarea de representar los materiales textuales en una forma electrónica adecuada a las necesidades de la investigación. El interés actual se centra en los lenguajes de marcado declarativos basados en el Standard Generalized Markup Language (SGML) o en su subconjunto el Extensible Markup Language (XML)." [2]

[1] http://cervantesvirtual.com

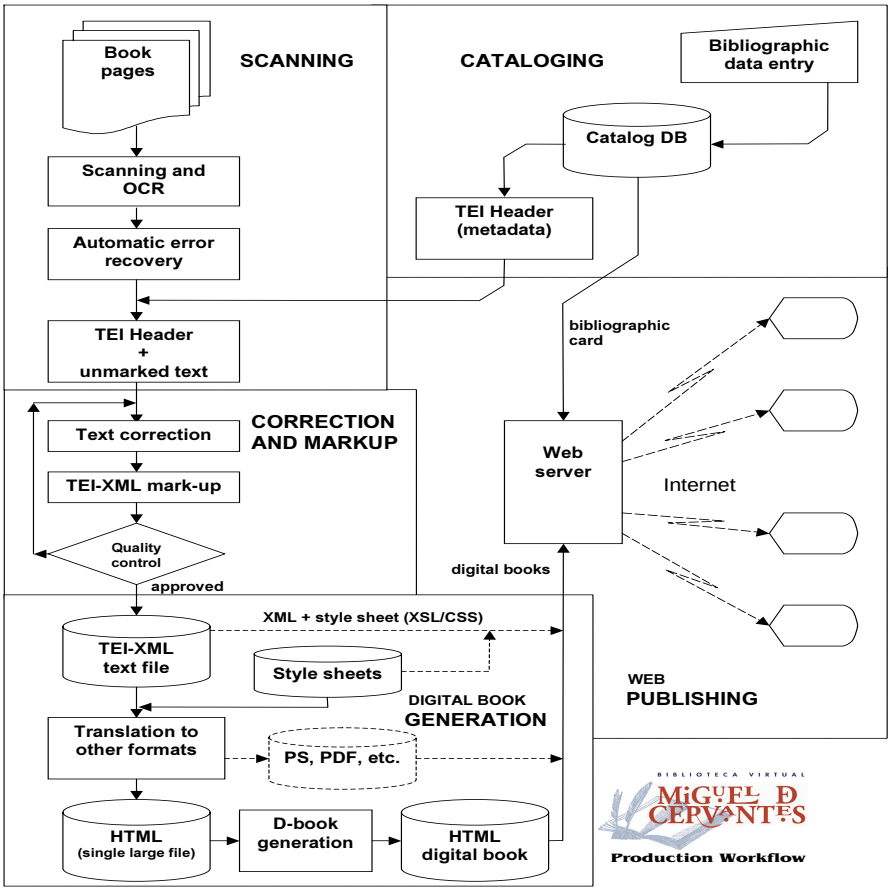

Fig. 1. Modelo de producción de libros digitales usando XML y TEI .

2 Una clasificación del marcado

Coombs, Renear y DeRose sostienen que los procesadores de textos disponibles "distraen a los autores de sus tareas de investigación y composición, centrando su atención en tareas tipográficas y de otros tipos" [3]. Defienden que una preocupación excesiva en la tecnología WYSIWYG [2] provoca que obviemos otra tarea, más importante, que es la de representar la estructura del documento. Por otra parte, presentan una interesante clasificación del marcado en cuatro categorías:

– Puntuacional (espacios, puntuación,...)

[2] *What Your See Is What You Get*, es una abreviatura popularizada en los años 90 que significa que lo que se ve en pantalla es una visión realista de lo que finalmente se obtiene.

– Presentacional (disposición, tipos de letra,...)
– Procedimental (comandos de formato)
– Descriptivo (etiquetas mnemónicas para los elementos del documento).

Coombs, Renear y DeRose enfatizan que únicamente algunos manuscritos antiguos no tienen ningún marcado en absoluto.

3 Beneficios del marcado descriptivo

El marcado puntuacional puede considerarse parte del propio texto. Los marcados presentacional y procedimental se centran básicamente en la apariencia. El marcado descriptivo es el nivel más alto de abstracción, reflejando la función que cada fragmento de texto desempeña en todo el documento. En la mayoría de los casos, el marcado presentacional y procedimental pueden aplicarse automáticamente basado en el marcado descriptivo y en un conjunto de reglas, normalmente recogidas en una hoja de estilo o plantilla.

Por lo tanto, si el marcado descriptivo por sí solo es suficiente para describir primero y procesar después un documento, la tarea del autor se reduce a asignar las etiquetas adecuadas (seleccionadas de un menú) a las partes estructurales del texto. Actualmente, la mayoría de los editores de XML o SGML utilizan este método de elección guiada por un menú. Estos son editores inteligentes que guían y ayudan a los autores a agregar el marcado descriptivo a sus textos.

Para lograr, por un lado, consistencia en el marcado dentro del mismo documento o, por otro lado, la estandarización de un conjunto entero de documentos, o simplemente para simplificar la tarea, se debe dar de antemano una definición estructural genérica del tipo de documento a crear. A estos efectos, existen hoy en día varios métodos para describir los posibles elementos estructurales que un documento puede contener. La DTD (definición del tipo de documento), aunque limitada en algunos aspectos, es el tipo más tradicional de descripción estructural no propietaria. Han aparecido recientemente diferentes formas de esquemas (Schemas) qué permiten una definición más exhaustiva de las propiedades de los elementos estructurales, como el concepto de tipo de datos (del modo en que se usa en los lenguajes de programación), qué permite un control más exhaustivo en la entrada de información (por ejemplo, un elemento fecha (date), definido como de tipo fecha, puede aceptar sólo una fecha válida como contenido), ver Morrison [4], capítulo 2. Yendo un paso más allá, Prescod sugiere el uso de los Autómatas de Bosque (Forest Automata) como una alternativa a las DTDs [5].

Asignar un código mnemotécnico, seleccionado de un menú, a un fragmento de texto es más sencillo que decidir cómo debe aparecer y luego recordar cómo conseguir esa apariencia. No digamos la necesidad de aplicar estos formatos repetidamente y de una manera uniforme a lo largo de todo el documento. Como Coombs, Renear y DeRose concluyen, “el marcado descriptivo permite a los autores centrarse en la creación literaria.”

4 Separar formato y estructura

Pero, ¿hasta qué punto se puede separar la estructura del marcado descriptivo?

Según John Bradley [6], "el SGML se diseñó para separar completamente el formato del texto de su estructura. El marcado SGML no debe decir nada en absoluto sobre el formato. El objetivo es claramente que parte del trabajo de cualquier software que finalmente tenga que mostrar algún texto sea obtener la información de formato de otra parte (como por ejemplo una hoja de estilo que diga al programa cómo mostrar cada elemento, pero que en realidad no sea parte del SGML), y usar como una guía la información de marcado estructural que encuentra en el texto más la información de la hoja de estilo para determinar cómo darle formato al texto que se debe mostrar."

Hemos dicho que la tarea de dar formato a un documento puede separarse del marcado descriptivo por medio de un conjunto genérico de reglas declarado aparte en una hoja de estilo. Esto es cierto en la medida en que esta información de formato pueda establecerse como una regla, es decir, como un procedimiento que pueda aplicarse a todos los elementos para los que se ha definido.

Pero hay algunos casos dónde hay información específica de un elemento que afecta el modo en que debe mostrarse y que no es aplicable a todos sus hermanos. Expliquemos esto con un ejemplo: un párrafo, que puede estar tabulado y justificado de muchas maneras. Nosotros podemos establecer tres opciones de tabulado, por ejemplo: primera línea, francesa o ninguno, y tres opciones para la justificación: izquierda, derecha, centrado y completa. Para separar completamente estos rasgos de formato del marcado descriptivo podríamos definir doce tipos de párrafo para cubrir todas las posibles combinaciones de tabulación y justificación mencionadas: parraf-primeralinea-izquierda, parraf-primeralinea-derecha, etc. Pero, ¿qué pasaría si agregáramos otro rasgo? Habría que multiplicar 12 por los posibles valores de este nuevo rasgo para obtener el total de posibles párrafos distintos que podrían ser usados. Queda claro que no es práctico definir un tipo diferente de párrafo para cada posible combinación de formatos de párrafo. Es preferible tener sólo un elemento párrafo con varios atributos que definan los rasgos específicos. De este modo, al diseñar un esquema de marcado debemos declarar todos los aspectos generales de formato en una hoja de estilo, pero debemos dejar los rasgos específicos para que sean asignados como valores de atributo. Si omitimos esta información específica del marcado declarativo la perderemos y no estará disponible en el momento de dar formato al texto. No podemos omitir información que puede ser necesitada más tarde, pero tampoco debemos agregar información que puede declararse como reglas generales para el elemento en una hoja de estilo.

5 Decisiones de marcado

5.1 Fuente única y transformación automática

Como quedó claro desde el inicio de nuestro proyecto de biblioteca digital, "un principio de buena práctica en la tecnología de la información es evitar la dupli-

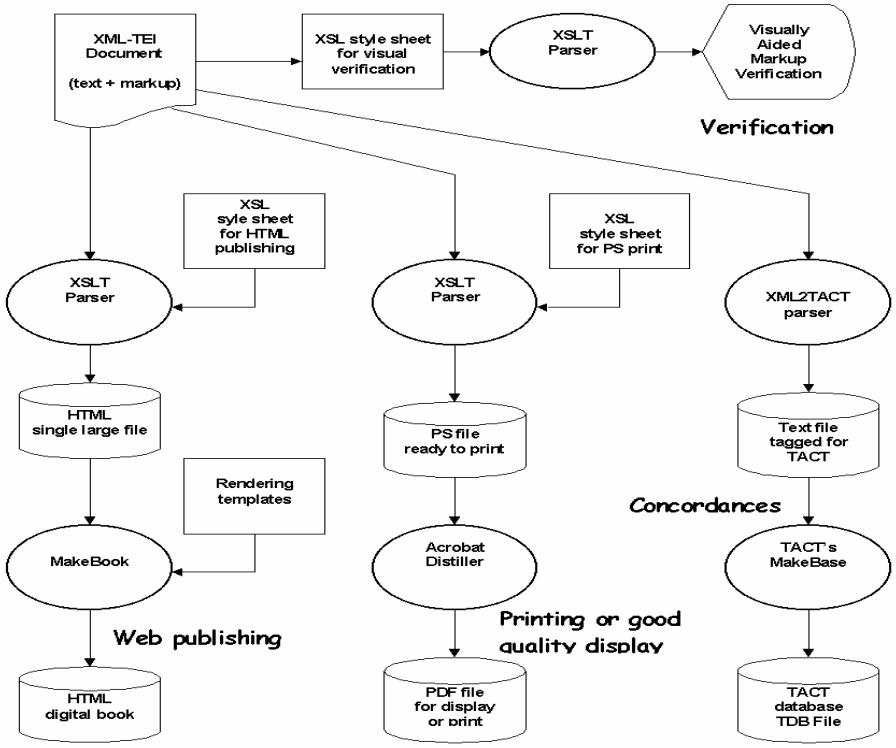

Fig. 2. Formatos diferentes obtenidos automáticamente a partir de un único fichero fuente.

cación de datos, y esto incluye los textos" [7]. Por tanto, es necesario elegir un formato para nuestro documento fuente a partir del cual todos los otros formatos puedan generarse automáticamente por medio de programas de transformación específicos (vea figura 2). No seguir este principio de fuente única tendría como resultado la creación de diferentes versiones de un mismo documento, lo cual crearía costosos problemas de actualización y mantenimiento de versiones y la posibilidad de errores y pérdida de información. Por otra parte, la alternativa respecto a automatizar la obtención de otros formatos que sean necesarios es crearlos a mano, lo cual llevaría una cantidad de tiempo y esfuerzo adicional considerables.

Arms describe, en su libro *Digital Libraries*, la relación entre la estructura y apariencia así como los métodos para obtener diferentes visiones de un mismo documento [8].

5.2 Consideraciones sobre preservación de documentos

Un aspecto importante en estas decisiones preliminares es la preservación de documentos. Debe preverse que el formato escogido para los documentos fuente

dure lo máximo posible y que pueda fácilmente convertirse a los nuevos formatos que la tecnología del futuro puedan producir. Esto nos ha hecho excluir a los formatos propietarios, que son difíciles de manejar, cambian según las caprichosas decisiones corporativas de los fabricantes y no son fáciles de convertir a otros formatos cuando la opción no es proporcionada por el fabricante. Este criterio de selección del formato debe ser parte de un plan global de preservación que debe abarcar aspectos como los procedimientos de preservación a seguir. Entre estos, podemos citar por ejemplo la renovación periódica de archivos para evitar la pérdida de datos debido al envejecimiento del soporte, los criterios de redundancia de datos y la elección adecuada de medios y lugares de almacenamiento.

5.3 Estructura vs. apariencia

El esquema de marcado que se utilice debe centrarse en la estructura de los documentos en lugar de su apariencia. La apariencia debe ser tratada de forma separada, fuera del documento, de modo que puedan aplicarse los cambios a voluntad a toda una colección de documentos sin gran esfuerzo. Como dice Arms, "no deben verse como alternativas o competidores, sino como una doble necesidad, ya que ambas merecen atención" [8].

5.4 Intercambio de documentos y fácil conversión

Es preferible que el formato escogido sea ampliamente usado, de no ser una norma universal, para facilitar la distribución y utilización del documento, así como la aplicación de herramientas de procesamiento estándar. La necesidad de un formato fácilmente adaptable a nuestras necesidades nos hizo rechazar formatos rígidos como el HTML, en favor de lenguajes de marcado más ricos y extensibles como el SGML y el XML. Otro aspecto deseable, aunque no obligatorio, es que el formato pudiera ser directamente soportado por los navegadores de internet.

Table 1. Comparación entre HTML, SGML y XML con respecto a los requerimientos de nuestra biblioteca digital (S=sí, N=no, E=se espera a corto plazo)

REQUISITO	HTML	SGML	XML
Es fácil de usar	S	N	S
Es extensible	N	S	S
Se centra en la estructura	N	S	S
Es fácilmente convertible	S	S	S
No es un lenguaje propietario	S	S	S
Es ampliamente usado	S	N	E
Se espera que dure	S	S	S
Es directamente soportado por los navegadores	S	N	E

De la comparación entre el HTML, SGML y XML (vea Tabla 1), podemos concluir que el XML tiene más ventajas que los otros. Steven DeRose compara estos tres lenguajes de marcado de forma detallada en su artículo *XML and the TEI* [9]. Podemos encontrar otra comparación entre el SGML y el XML llevada a cabo por James Clark en el sitio web de la W3C [10].

5.5 Elección del esquema de marcado

Después de elegir el XML, tuvimos que decidir entre definir nuestro propio conjunto de etiquetas o simplemente usar uno ya existente. La primera opción significaba mucha planificación y esfuerzo, y el resultado no favorecería el intercambio de documentos. La segunda implicaba encontrar un esquema de marcado adecuado a nuestras necesidades. Tras analizar esta opción encontramos dos candidatos: DocBook [11] y TEI [12] [13].

El tipo de textos que nosotros procesamos son principalmente libros clásicos literarios e históricos, tanto en prosa como en verso y drama, y en ocasiones diccionarios. Esto requería un esquema de marcado muy completo. Nos decidimos finalmente por el esquema TEI porque parecía el más adecuado para este tipo de documentos y porque había sido usado con éxito en otros importantes proyectos de digitalización [3], como Perseus [4], el Oxford Text Archive [5], el Women Writers Project [6] en la Brown University, etc. Hemos elegido trabajar con la versión XML del TEI, debido a las ventajas del XML descritas anteriormente, aun cuando el esquema TEI no había terminado de ser convertido de SGML a XML. Cuando tomamos la decisión, en 1999, había sólo un subconjunto del esquema TEI convertido a XML por la Universidad de Oxford, llamado TEI-lite [14] [7], por lo que tuvimos que empezar con esta DTD e introducirle algunas modificaciones propias.

6 Nuestra política de marcado

Con XML se puede codificar cada aspecto de un texto, desde las divisiones estructurales hasta los rotos de un manuscrito, desde los párrafos hasta cada una de las palabras. Sin embargo, en la Biblioteca Virtual Miguel de Cervantes hemos establecido algunos parámetros sobre lo que debe y lo que no debe marcarse (ver fig. 3) Nosotros pensamos que los textos literarios deben tener un marcado bastante completo, pero en todo caso no hasta el punto de marcar palabras individuales. No obstante, hemos visto que otros proyectos de digitalización usan un

[3] Ver la página de aplicaciones del TEI, http://quirk.oucs.ox.ac.uk/TEI/Applications/. Esta lista indica todos los proyectos que están usando el esquema de marcado del TEI para etiquetar sus documentos.
[4] http://perseus.tufts.edu/
[5] http://ota.ahds.ac.uk/
[6] http://www.wwp.brown.edu/
[7] La versión en XML del TEI-lite utiliza una DTD llamada teixlite.dtd, que está disponible en internet.

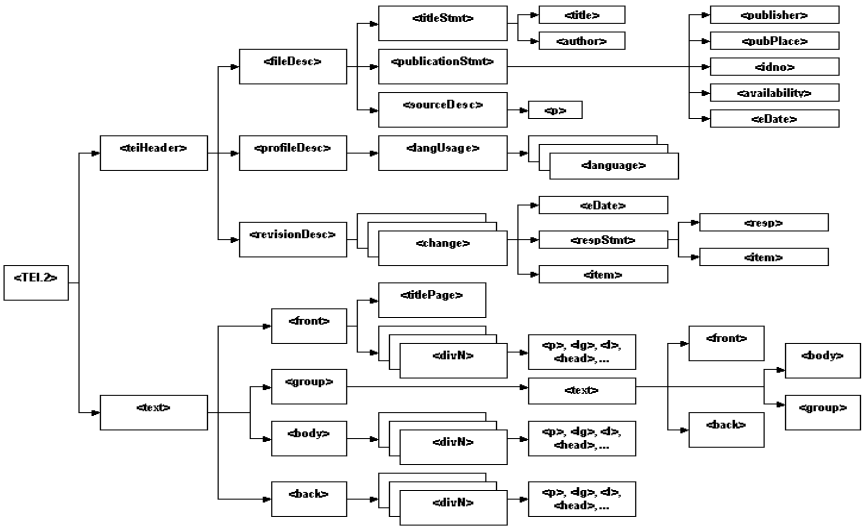

Fig. 3. Esquema de marcado usado en la Biblioteca Virtual Miguel de Cervantes.

esquema de marcado innecesariamente complejo que lleva a un incremento en el tiempo y coste de producción de los textos. Por tanto, nosotros necesitábamos encontrar una solución de compromiso entre un marcado lo suficientemente complejo y un coste lo suficientemente bajo.

Finalmente decidimos que uno de nuestros principios de diseño sería no agregar marcas que no fueran a utilizarse. Esto implicaba prever los usos y procesos que se aplicarían a los textos marcados, como la transformación del formato, las búsquedas, las concordancias automáticas, etc. De esta manera, nosotros sabríamos qué elementos estructurales debían marcarse. Si cualquier aspecto imprevisto aparecía después, podrían agregarse nuevos elementos, no sin coste, pero esto es preferible a realizar un marcado excesivo desde el principio. Según el TEI, y desde un punto de vista de alto nivel, la estructura de un texto incluye dos grandes partes: un *teiHeader* o cabecera TEI, con los metadatos correspondientes a ese texto y el proceso por el que ha pasado antes de publicarse electrónicamente, y un elemento *text* que recoge el contenido de la obra literaria. Dentro del *text*, se codifican también las tres partes principales de la obra según lo establece el TEI: *front*, *body* y *back*.

Del mismo modo, también se marcan las divisiones, títulos, párrafos, poemas, líneas de verso, palabras en otro idioma, etc. Sin embargo, nosotros no indicamos los fines línea y las tachaduras (excepto en el caso de manuscritos).

7 El modelo que proponemos

El modelo de trabajo que proponemos consta de tres niveles:

1 - un vocabulario de marcado general, completo y multi-propósito (como TEI), para facilitar el intercambio de documentos entre proyectos diferentes.

2 - una DTD de carácter general para nuestro proyecto, que no es más que un subconjunto seleccionado del vocabulario de marcado general. Una herramienta idónea para la construcción de esta DTD es el Pizza Chef [15].

3 - Varias DTDs reducidas obtenidas por simplificación automática a partir de la DTD de carácter general (una por cada subtipo de documento). Estas DTDs son útiles para validar los documentos, actuando a su vez de guía al crear un entorno restringido de marcado que limita las posibilidades a las mínimas necesarias para cada subtipo de documento, simplificándose de este modo la tarea mientras que se favorece también la reducción del número de posibles errores. Para ello usamos DTDprune [16] [17], un programa de simplificación automática de DTDs que hemos desarrollado. Contrariamente a lo que pueda parecer, una DTD reducida más restrictiva es más simple de entender, simplificando considerablemente la tarea de marcado.

7.1 Cinco niveles de modificación al DTD

Para obtener las DTDs reducidas a partir de la DTD general, hemos establecido cinco reglas o tipos de modifiación permitidas. Las primeras tres reglas son básicamente restricciones que producen documentos que cumplen la DTD original del TEI-lite. Los otros dos tipos de modificaciones producen documentos que no cumplen totalmente la DTD original, pero puede hacerse que la cumplan por medio de una transformación simple.

1. El primer tipo de modificación consiste en añadir valores predefinidos específicos a los atributos de los elementos, para obligar a los codificadores a insertar los valores esperados en lugar de una cadena libre de caracteres. Este tipo de cambio es muy aconsejable para normalizar los valores del atributo dentro de un proyecto del digitalización, prevenir posibles errores humanos y para forzar el uso de los valores predefinidos, que serán necesarios en el posterior procesamiento. Los documentos resultantes de este tipo de modificación cumplen la DTD original que incluye la declaración "CDATA" para la mayoría de los valores de los atributos.

2. El segundo tipo de modificación consiste en agregar nuevos atributos que no existen en la DTD original. Por ejemplo, nosotros incluimos atributos más específicos para el formato en lugar del *rend* sugerido en la DTD del TEI. Por ejemplo, en el caso del elemento $< hi >$, hemos sustituido el atributo *rend* por cuatro nuevos atributos: *italic*, *bold*, *underline* y *position*. En el caso del elemento $< p >$, hemos agregado *align*, *indent* y *specialindent*.

```
<!ATTLIST p
     ...
    align  (left | right | center | justify)"justify"
    indent (left | right |  both | none) "none"
    specialindent (none |   firstline | french) "firstline"
    TEIform CDATA "p" >
```

Este tipo de cambio, si bien se aparta de la norma TEI, puede deshacerse fácilmente quitando todos los nuevos atributos por medio de un programa filtro adecuado, en el caso de que se necesite por algún motivo un cumplimiento total del TEI. Los ejemplos mostrados tienen relación con el problema discutido anteriormente, sobre hasta qué punto puede separarse la estructura del formato. Nosotros creemos que debe haber un mínimo necesario de información de formato incluida en el marcado.

3. El tercer tipo de modificación consiste en hacer obligatorios (#REQUIRED en la jerga de las DTD) algunos atributos que no lo son (#IMPLIED). De esta manera, la DTD obliga a los etiquetadores a rellenar el atributo, normalmente a partir de una lista de valores predefinidos.

4. El cuarto, consiste en quitar de la DTD aquellos elementos que consideramos innecesarios para nuestros propósitos. Para esto debemos tener una idea clara de los posibles procesos que aplicaremos a nuestros textos a posteriori, y las correspondientes necesidades de marcado. Esta eliminación de elementos hace que la DTD sea más restrictiva para los etiquetadores y evita, en alguna medida, errores humanos ya que aquellos elementos que no deseamos que se usen no podrán ser incluidos. Nosotros siempre hemos estado a favor de un esquema de marcado simple ya que no tiene ningún sentido usar marcas que no se utilizarán para nada posteriormente, y esto dificulta innecesariamente la tarea de marcado. Hay dos tipos de eliminación de elementos. Uno es quitar elementos de la DTD original para que estos ya no puedan usarse en nuestros textos XML. Por ejemplo, nosotros hemos quitado el elemento $< div >$ (divisiones sin numerar), para forzar el uso exclusivo de las divisiones numeradas $< divN >$. El otro tipo de eliminación consiste en quitar la posibilidad de inserción de algunos elementos dentro de otros modificando los modelos de contenido (content models), como por ejemplo, para desactivar el uso de $< hi >$ dentro del elemento $< head >$.

5. El quinto tipo de modificación consiste en agregar elementos completamente nuevos que no existen en la DTD general. Este es el tipo cambio más delicado, porque los archivos que se produzcan con DTDs que hayan sufrido este tipo de cambio no cumplirán la norma TEI. Nosotros hemos evitado este tipo de cambios y sólo justificamos su uso cuando implican una ventaja o simplificación de gran importancia.

Finalmente, hemos tratamos que todos nuestros documentos XML cumplan el TEI en la mayor medida posible. Por tanto, podemos decir que nuestro esquema de marcado es un subconjunto, muy restrictivo quizás, del TEI, pero no un esquema diferente.

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. 1st edn. ACM press and Addison Wesley, Edinburgh Gate, Harlow, Essex CM20 2JE, England (1999) See also http://www.dcc.ufmg.br/irbook or http://sunsite.dcc.uchile.cl/irbook.
2. Sperberg-McQueen, C.M., Huitfeldt, C.: Concurrent Document Hierarchies in MECS and SGML. In: ALLC/ACH98 conference, Debre-

cen (1998) http://www.oasis-open.org/cover/sperbergACH98.html and http://lingua.arts.klte.hu/allcach98/abst/abs47.htm (the 2nd is the canonical and official URL).
3. Coombs, J., Renear, A.H., DeRose, S.J.: Markup systems and the future of scholarly text processing. Communications ACM **30/11** (1987) 933–947 Cf. CACM 31/7 (July 1988) 810-811.
4. Morrison, M.: XML Al descubierto. Prentice Hall, Madrid (2000) (translated from XML Unleashed, 2000, Sams, 0-672-31514-9).
5. Prescod, P.: Formalizing XML and SGML Instances with Forest Automata Theory. Technical report, University of Waterloo, Department of Computer Science, Waterloo, Ontario (1998) Draft Technical Paper.
6. Bradley, J.: SGML2TDB: User's Guide. http://www.chass.utoronto.ca/cch/tact.html (1997) (available via anonymous ftp from tactweb.humanities.mcmaster.ca (North America) and ilex.cc.kcl.ac.uk (Europe)).
7. Bia, A., Pedreño, A.: The Miguel de Cervantes Digital Library: the Hispanic voice on the web. LLC (Literary and Linguistic Computing) journal, Oxford University Press **16** (2001) 161–177 Presented at ALLC/ACH 2000, The Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities, 21/25 July 2000, University of Glasgow.
8. Arms, W.: Digital Libraries. MIT Press, Cambridge, Massachusetts (2000)
9. DeRose, S.: XML and the TEI. In Mylonas, E., Renear, A., eds.: Text Encoding Initiative: Anniversary conference; 10th — November 1997, Providence, RI. Volume 33(1) of Computers and the Humanities 1999; /2., Norwell, MA, USA, and Dordrecht, The Netherlands, Kluwer Academic Publishers Group (1999) 11–30
10. Clark, J.: Comparison of SGML and XML. http://www.w3.org/TR/NOTE-sgml-xml-971215 (1997)
11. Cover, R.: DocBook, homepage (within OASIS). http://www.oasis-open.org/docbook/ (visited July 10th 2000)
12. Sperberg-McQueen, C.M., Burnard, L., eds.: Guidelines for Electronic Text Encoding and Interchange (Text Encoding Initiative P3), Revised Reprint, Oxford, May 1999. TEI P3 Text Encoding Initiative, Chicago - Oxford (1994)
13. Burnard, L.: Text encoding for information interchange: An introduction to the text encoding initiative. http://www-tei.uic.edu/orgs/tei/info/teij31/index.html (1995)
14. Burnard, L., Sperberg-McQueen, C.M.: Tei lite: An introduction to text encoding for interchange (document no: Tei u 5). http://www.uic.edu/orgs/tei/intros/teiu5.html (1995)
15. Burnard, L.: The Pizza Chef: a TEI Tag Set Selector. http://www.hcu.ox.ac.uk/TEI/pizza.html (1997) (Original version 13 September 1997, updated July 1998; Version 2 released 8 Oct 1999).
16. Bia, A., Carrasco, R.C.: Automatic DTD simplification by examples. In: ACH/ALLC 2001. The Association for Computers and the Humanities, The Association for Literary and Linguistic Computing, The 2001 Joint International Conference, New York University, New York City (2001) 7–9
17. Bia, A., Carrasco, R.C., Forcada, M.L.: Identifying a reduced DTD from marked up documents. In Sánchez, J.S., Pla, F., eds.: Pattern Recognition and Image Analysis (Proceedings of the IX Spanish Symposium on Pattern Recognition and Image Analysis – SNRFAI'2001 conference). Volume II of Treballs d'informática i tecnologia, num.7., Castellón de la Plana, Spain, Publicacions de la Universitat Jaume I, 2001 (2001) 385–390