Guía de usuario de Linuxdoc-SGML

Por Matt Welsh. Puesta al día por Greg Hankins, `greg.hankins@cc.gatech.edu`
Traducido por Francisco José Montilla, `pacopepe@insflug.org` v1.5, 8 Marzo 1996

Este documento es una guía de usuario del sistema Linuxdoc-SGML, un formateador basado en SGML que permite obtener varios tipos de formatos de salida. Se pueden producir archivos de texto plano (ASCII e ISO-8859-1), DVI, PostScript, HTML, GNU info, LyX, y RTF a partir de un sólo fuente SGML. Esta guía documenta el paquete Linuxdoc-SGML en su versión 1.5.

Índice General

1 Introducción

Esta es una guía de usuario para el sistema de procesado de documentación `linuxdoc-SGML`, concebido para producir información relativa a LiNUX. Consiste en una Definición de Tipo de Documento SGML (DTD), junto con un juego de "archivos de reemplazo" que convierten el SGML a fuentes inteligibles por `groff`, LaTeX, HTML, GNU info, LyX, y RTF.

`linuxdoc-sgml` Está fuertemente basado en el paquete QWERTZ DTD de Thomas Gordon `thomas.gordon@gmd.de`. Lo único que he hecho han sido revisiones a sus definiciones de tipo de documento, y los archivos sustitutorios para su empleo específico con d ocumentación relativa a LiNUX.

`linuxdoc-sgml` no pretende ser un sistema genérico de procesado de documentación. A pesar de que puede emplearse para documentos de varias clases, lo he limitado para el uso de los escritores de LiNUX para producir HOWTOs, FAQs, y (posteriormente) para los manuales del Linux Documentation Project (LDP). No pretende ser un sistema de pr ocesado de documentación genérico; He manipulado sus posibilidades con este propósito.

Si se percibe una falta de generalidad en el sistema, se debe a esta razón. No hay nada que limite `linuxdoc-SGML` a la producción de documentación sobre LiNUX, pero todo lo producido con este sistem a tendrá un aspecto característico. Si se desea que la apariencia sea distinta, sugiero que se emplee un sistema más generalizado como el QWERTZ DTD, en combinación con las herramientas Linuxdoc-SGML.

Uno de los objetivos de este sistema es hacer fácil la producción de documentos en distintos formatos. Hasta ahora, la mayoría de la documentación LiNUX había sido producida a través de edición manual. Un sistema como `groff` puede ocuparse del formate o para una versión ASCII, pero sigue sin poder proporcionar HTML (para su uso en World Wide Web), LaTeX (para documentos impresos con calidad), o GNU `info`. Por este motivo, si faltan prestaciones en este sistema de las que se quisiese disfrutar, *házmelo saber, por favor!* La idea es que no hagan falta demasiadas triquiñuelas para producir documentos agradables en múltiples formatos.

El autor debe de tener que hacer lo mínimo posible.

1.1 Acerca de este documento.

Este documento fue escrito usando el DTD `linuxdoc-sgml`. Contiene más o menos todo lo que se necesita saber para escribir documentos SGML con este DTD (definición de tipo de documento). Véase `Linuxdoc-Ejemplo.sgml` para ver un ejemplo de documento que se puede tener como modelo a la hora de escribir los propios.

1.2 ¿Por qué SGML?

Elegí SGML para este sistema porque está concebido específicamente para su posterior transformación a otros formatos. SGML, abreviatura de *Standard Generalized Markup Language*, permite especificar la *estructura* del documento—esto es, qué tipo de cosas dan forma al documento. La estructura del documento se especifica con un DTD. `linuxdoc-sgml` es un DTD que especifica la estructura para los HOWTOs y otros documentos de LiNUX. QWERTZ es otro DTD; el SGML estándar proporciona DTDs para libros , artículos, y otras clases de documentación genérica.

El DTD especifica los nombres de los "elementos" que forman parte de un documento. Un elemento es simplemente una parte de la estructura—como una sección, subsección, un párrafo, o simplemente algo menor como *texto remarcado*. A diferencia del LaT eX, no obstante, estos elementos no son en modo alguno intrínsecos al SGML en sí mismo. La definición de documento linuxdoc-sgml viene a definir elementos muy similares a su equivalentes en LaTeX —tiene secciones, subsecciones, "entornos" de texto y d emás—. No obstante, empleando SGML se pueden definir cualquier tipo de estructuras de documento que se quieran. En cierto modo, es similar al TeX de bajo nivel, mientras que el DTD `linuxdoc-sgml` se parece más al LaTeX.

No dejarse confundir por esta analogía. SGML *no* es un sistema de formateo de documentación. No existe un "formateador SGML" en sí mismo. El fuente SGML es *sólo* convertido a otros formatos para su procesado. Además, el SGML por sí mismo lo únic o que hace es especificar la estructura del documento. No existen dispositivos o "macros" intrínsecos al SGML en sí mismo. Todos estos aspectos son definidos en el DTD. No se puede hacer uso del SGML sin el DTD —el DTD define lo que hace el SGML—

1.3 Cómo funciona.

He aquí cómo procesar un documento con SGML y cómo funciona el DTD `linuxdoc-sgml`. Primero, se necesita una definición de tipo de documento (DTD). Yo estoy empleando el DTD QWERTZ, que fue realizado en sus principios por un grupo de personas que neces itaban un DTD estilo LaTeX. He modificado el DTD QWERTZ obteniendo el DTD `linuxdoc-sgml` para nuestros propósitos. El DTD únicamente configura la estructura del documento. Una pequeña porción del mismo tiene este aspecto:

```
<!element article - -
        (titlepag, header?,
         toc?, lof?, lot?, p*, sect*,
         (appendix, sect+)?, biblio?) +(footnote)>
```

Esta parte configura la estructura global para un "article", que viene a ser un "estilo de documento" en LaTeX. El artículo consiste en una página de portada (`titlepag`), un encabezamiento opcional, (`header`); una tabla de contenidos opcional también, (`toc`), listado de figuras y de tablas opcionales (`lof` y `lot`), cualquier número de párrafos (`p`), y secciones principales (`sect`) apéndices opcionales (`appendix`), bibliografía (`biblio`) y notas al pie (`footnote`).

Como puede observarse, el DTD no menciona nada acerca de cómo debería de formatearse el documento, o sobre el aspecto que debería tener. Simplemente define qué partes conforman el documento. En alguna parte del DTD, son definidas las estructuras de `titlepag`, `header`, `sect`, así como otros elementos.

No se precisa saber nada acerca de la sintaxis del DTD para escribir documentación. Simplemente lo presento para que así se sepa su aspecto, y lo que hace. Se *necesita* estar familiarizado con la *estructura* que defina el DTD. De no ser así, podrí a violarse la estructura a la hora de tratar de escribir un documento, y quedarse muy confundido con los mensajes de error resultantes. Describiremos en detalle la estructura de los documentos `linuxdoc-sgml` más tarde.

El siguiente paso es escribir el documento empleando la estructura definida por el DTD. Nuevamente, el DTD `linuxdoc-sgml` hace que la apariencia de los documentos sea muy similar al LaTeX —muy simple de seguir—. En la jerga SGML, un documento esc rito empleando un DTD específico, se conoce como una instancia de ese documento.

A fin de traducir el fuente SGML a otro formato (como LaTeX o groff) para su procesado, el fuente SGML (el documento que escribiste) es *analizado* junto con el DTD por (lo adivinaste) el *analizador* SGML.

Estoy empleando el analizador **sgmls** de James Clark, jjc@jclark.com, quien precisamente resulta ser el creador de groff. Estamos en buenas manos. El analizador (sgmls) únicamente toma tu documento y verifica que sigue la estructura definida anteriormente en el DTD. También devuelve un formato más esplícito de tu documento, con todos las “macros” y demás elementos expandidos, lo cual es inteligible para sgmlasp, la siguiente fase del proceso.

sgmlsasp es el responsable de la conversión de la salida de sgmls a otro formato (como LaTeX). Lleva a cabo esta tarea usando *archivos sustitutivos*, que describen cómo convertir elementos del SGML original al fuente correspondiente en el formato “destino” (como LaTeX o groff).

Por ejemplo, parte del archivo de reemplazamiento para LaTeX tendría este aspecto:

```
<itemize>       +       "\\begin\{itemize\}      +
</itemize>      +       "\\end\{itemize\}        +
```

El cual dice que siempre que se comience un elemento itemize en el fuente SGML, deberá ser reemplazado por

```
\begin\{itemize\}
```

en el fuente LaTeX. (Como dije, los elementos del linuxdoc-sgml DTD son muy similares a sus equivalentes en LaTeX).

Asi que para convertir el SGML a otro formato, todo lo que se ha de hacer es escribir un archivo de reemplazamiento nuevo para ese formato que proporcione las analogías apropiadas para los elementos SGML en el nuevo formato. En la práctica, no es tan simp le —por ejemplo, si se está intentando convertir a un formato que no está estructurado enteramente como tu DTD, van a haber problemas—. En cualquier caso, es más fácil de hacer que escribir analizadores individuales y traductores para varios tipos de formatos de salida; SGML proporciona un sistema genérico de conversión de fuentes a varios formatos.

Una vez sgmlsasp ha concluido su trabajo, se obtiene el fuente LaTeX que corresponde al documento SGML original, el cual puede ser formateado empleando LaTeX como se haría normalmente. Posteriormente daré en este documento ejemplos y mostraré los com andos empleados para llevar a cabo la traducción y formateo. Todo puede ser llevado a cabo desde la línea de comandos.

Pero primero, debo describir cómo instalar y configurar el software.

2 Instalación.

Hacerse con linuxdoc-sgml-1.5.tar.gz de alguno de los siguientes servidores ftp:

- ftp://sunsite.unc.edu/pub/Linux/utils/text/linuxdoc-sgml-1.5.tar.gz
- ftp://tsx-11.mit.edu/pub/linux/docs/linuxdoc-sgml-1.5.tar.gz
- ftp://ftp.cc.gatech.edu/pub/people/gregh/linuxdoc-sgml/linuxdoc-sgml-1.5.tar.gz

Parches actualizados para la versión 1.5 pueden encontrarse en:

ftp://ftp.cc.gatech.edu/pub/people/gregh/linuxdoc-sgml.

Puede obtener información actualizada en:

Linuxdoc-SGML WWW Page.

El archivo linuxdoc-sgml-1.5.tar.gz contiene todo lo necesario para escribir documentos SGML y convertirlos a groff, LaTeX, HTML, GNU info, LyX, y RTF. Además de este paquete, se necesitarán las siguientes herramientas – no son requeridas por el sistema SGML en sí, pero sugiero hacerse con ellas para poder así formatear tus documentos y verificar que todo aparece como debe antes de distribuirlas.

1. `groff`. *Necesitas* la versión 1.08 ó 1.09. Aparentemente, parte del tratamiento de los márgenes en `groff` se encuentra en estado de perfeccionamiento de versión en versión; ambos funcionarán, pero se obtendrán resultados ligeramente dist intos. (Particularmente, en la 1.09 el margen izquierdo no es sangrado dos caracteres como lo era en la 1.08.
   Es soportable en la 1.09, pero en la 1.08, tiene un aspecto terrible) Las versiones precedentes a la 1.08 *no funcionarán*. Puedes obtener esto de `ftp://prep.ai.mit.edu/pub/gnu`.

   Hay una versión binaria para LiNUX en `ftp://sunsite.unc.edu/pub/Linux/utils/text` También. Necesitarás `groff` para producir texto ASCII simple a partir tus documentos SGML. (TeX/LaTeX se emplearán para obtener bonitas copias impresas en PostScript y dvi).

2. TeX y LaTeX. Estos están disponibles más o menos, en todos lados; no debería de haber problema en conseguirlo e instalarlo. (Hay una distribución de binarios en `sunsite.unc.edu`). Por supuesto, sólo lo necesitaremos si pretendemos formatear nues tros documentos SGML con LaTeX/TeX, por lo que esto es opcional.

3. Si quieres visualizar el HTML generado, recomiendo conseguir el NCSA Mosaic 2.6 o posterior, disponible en `ftp://sunsite.unc.edu/pub/Linux/system/Network/info-systems`.

4. `gawk` y las herramientas GNU info, para formatear y visualizar los ficheros info. Están disponibles también en

   `ftp://prep.ai.mit.edu/pub/gnu`, o en `ftp://sunsite.unc.edu/pub/Linux/utils/text` (para `gawk`) y

   `ftp://sunsite.unc.edu/pub/Linux/system/Manual-pagers` (para las herramientas GNU info.)

5. LyX (un interface quasi-WYSIWYG para el LaTeX, con esquemas SGML), está disponible en `ftp://ftp.via.ecp.fr`.

2.1 Instalación del software

Los pasos necesarios para la instalación y configuración de `linuxdoc-sgml` son los siguientes:

1. Primero, desempaquetar el archivo tar `linuxdoc-sgml-1.5.tar.gz` donde sea. Esto creará el directorio `linuxdoc-sgml-1.5`, donde residirán todos los archivos SGML. No importa donde descomprimas este archivo; simplemente, no muevas nada del dir ectorio `linuxdoc-sgml-1.5`.

2. Leer el fichero `INSTALL`. Contiene instrucciones detalladas de instalación.

   Si todo va bien, estarás listo para empezar a utilizar el sistema.

2.2 Formateando documentos

Digamos que tienes el documento SGML `pepito.sgml` que queremos formatear. He aquí una revisión genérica de cómo formatear el documento a lo s distintos formatos. Para ver una lista completa de opciones, consultar las páginas `man`

2.2.1 Obteniendo texto plano como salida.

Si se quiere obtener texto plano como resultado, emplear el comando:

```
% sgml2txt foo.sgml
```

Nótese que he limitado la conversión de groff para la salida de texto simple. Es decir, he quitado los encabezamientos de página, cambiado los márgenes, y demás. Con un poco de trabajo, puedes producir PostScript y DVI de la salida de groff, pero sugiero que se emplee LaTeX para ello.

Puedes producir fuentes groff para páginas man, que pueden ser formateadas con `groff -man`. Para hacer esto:

```
 % sgml2txt -man
foo.sgml
```

2.2.2 Produciendo LaTeX, DVI o PostScript

Para producir documentos LaTeX a partir del fuente SGML, ejecutar:

```
% sgml2latex foo.sgml
```

Si quieres producir salida PostScript (vía `dvips`) emplea la opción `-p`:

```
% sgml2latex -p foo.sgml
```

O puedes producir un fichero DVI usando el parámetro `-d`, como aquí:

```
% sgml2latex -d foo.sgml
```

2.2.3 Produciendo salida HTML

Si quieres obtener HTML como salida, haz lo siguiente:

```
% sgml2html -img foo.sgml
```

Esto producirá tanto `foo.html`, como `foo-1.html`, `foo-2.html`, y demás —un archivo por cada sección del documento. Ejecuta tu navegador WWW en `foo.html`, que es el archivo cabecera. Asegurarse de que todos los ficheros HTML correspondientes al documento están en un directorio, ya que se referencian unos a otros con URLs locales. Los iconos referenciados en la salida HTML están localizados en `$LINUXDOCLIB/icons`. Estos deberán ser copiados también a la localización final de los documentos HTML. `$LINUXDOCLIB` está definida al comienzo de los scripts de con versión SGML.

Si empleas `sgml2html` sin el parámetro `-img`, los documentos HTML tendrán las etiquetas “previous”, “next”, y “table of contents” para la navegación. Puedes saltarte esa configuración por defecto creando un archivo en `$LINUXDOCLIB /rep/html/`<`nombre_archivo`>, sustituyendo con tus palabras, las apropiadas para los distintos lenguajes. El archivo posee el siguiente formato:

```
PrevPage:    nuevovalor
NextPage:    nuevovalor
TOC:         nuevovalor
```

Ver el ejemplo `deutsch`.

2.2.4 Creando salida GNU Info

Si quieres formatear tu archivo para el visualizador GNU info, lo único que has de ejecutar es:

```
% sgml2info foo.sgml
```

2.2.5 Creando salida LyX

Para obtener salida LyX, emplear el comando:

```
% sgml2lyx foo.sgml
```

2.2.6 Creando salida RTF

Si quieres obtener salida RTF, usa el comando:

```
% sgml2rtf foo.sgml
```

Esto producirá como salida `foo.rtf`, así como `foo-1.rtf`, `foo-2.rtf`, y así; un fichero por cada sección del documento.

2.2.7 Comprobando la sintaxis del SGML

Si lo único que quieres es comprobar si habría errores en la conversión del SGML emplea el script `sgmlcheck`. Por ejemplo:

```
% sgmlcheck foo.sgml
```

2.3 Tabla de Códigos ISO 8859-1

La tabla de caracteres ISO 8859-1 puede ser usada con caracteres internacionales en transformaciones a texto ASCII plano, LaTeX y HTML[1] Para hacer u so de esta propiedad, pasar al script el parámetro `-l`, por ejemplo:

```
% sgml2txt -l foo.sgml
```

Puedes emplear los caracteres ISO 8859-1 en los fuentes, serán transformados en los códigos de escape apropiados de cada tipo de formato.

3 Redacción de Documentos con Linuxdoc-SGML

Para la mayoría, redactar documentación empleando la Definición de Tipo de Documento es muy simple, y similar en ciertas cosas al LaTeX. No obstante, hay algunas advertencias a tener en cuenta. En esta sección, expondré en una introducción la redacción de documentación con SGML.

Ver el archivo `Linuxdoc-Ejemplo.sgml` para tener un ejemplo (y tutorial) de documento SGML que pueda usarse como modelo a seguir a la hora de redactar tus propios documentos. Sólo discutiré algunas de las prestaciones del SGML aquí, pero el fuente no es muy le gible como ejemplo. Mejor imprimir el fuente (así como la salida ya formateada) de `Linuxdoc-Ejemplo.sgml` para así tener una referencia real a la que acudir.

[1] nota para los textos en Castellano:
Esta es la tabla que se ha de emplear si se emplean tildes, eñes, etc; basta con escribirlos como tales en los fuentes, y emplear el parámetro '-l' en los scripts de conversión que se empleen (sgml2latex, sgml2text o sgml2html)

3.1 Conceptos Básicos

Examinando los fuentes del documento ejemplo, te percatarás de que hay una serie de "etiquetas" enmarcadas con paréntesis en ángulo[2] (< y >). Una etiqueta especifica simplemente el comienzo y final de un elemento, siendo un elemento algo como una sección, párrafo, una frase con texto en tipografía cursiva (*itálica*), un "item" de un listado , y demás. El uso de una etiqueta es similar a los comandos de LaTeX como `\item` o `\section{...}`.

Por poner un ejemplo sencillo, para producir **este texto en negrita**, escribí:

```
Por poner un ejemplo sencillo, para producir <bf>este texto en negrita</bf>, ...
```

en el fuente. `<bf>` da comienzo a la región de texto en negrita, y `</bf>` la finaliza. Como alternativa, se puede emplear la forma abreviada

```
Por poner un ejemplo sencillo, para producir <bf/este texto en negrita/, ...
```

que encierra el texto en negrita entre barras. (Si el texto contenido entre las barras contiene barras también, deberás usar el método "largo", como es el caso de los nombres de archivo en Unix).

Hay otros aspectos que tener en cuenta respecto a caracteres especiales (esa es la razón por la que verás todas esas absurdas expresiones con simbolos de "&" si observas los fuentes; las describiré pronto).

En algunos casos, la etiqueta de fin para un elemento en particular es opcional. Por ejemplo, para comenzar una sección, empleas la etiqueta de `<sect>`; de todos modos, la etiqueta de finalización de la sección (que podría aparecer al final del cuerpo de la sección propiamente, ¡no justo tras el título de la sección!) es opcional e implícito cuando se da comienzo a otra sección de la misma "profundidad". En general, no debemos preocuparnos de esos detalles; basta con seguir el modelo empleado en el tutorial (`Linuxdoc-Ejemplo.sgml`).

3.2 Caracteres Especiales

Para obtener una lista detallada de caracteres especiales, echar un vistazo a alguno de los ficheros de reemplazamiento. Normalmente el LaTeX se quejará de la mayoría de caracteres especiales, por lo que paginando `$LINUXDOCLIB/rep/latex/general` sería un buen sitio donde comenzar. `$LINUXDOCLIB` está definido al comienzo de los scripts de conversión.

Obviamente, los paréntesis en ángulo son por sí mismos caracteres especiales en los fuentes SGML. Hay otros a los que prestar atención. Por ejemplo, digamos que quisieras teclear una expresión que contenga paréntesis en ángulo, como esta: `<foo>`. Para obtener el paréntesis en ángulo izquierdo (o de apertura)tendrás que emplear el elemento `<` que es una macro que se expandirá al carácter de paréntesis en ángulo actual. Por tanto, en el fuente tecleé:

```
... que contenga parentesis en angulo, como esta: <tt>&lt;foo></tt>.
```

Generalmente, lo que comience con un símbolo de "&"[3] es una macro especial. Por ejemplo, existe `%` para obtener un %, `|` para |, y así. Para todos los caracteres especiales existen estas entidades precedidas por "ampersands" para representarlos.

Normalmente, no necesitamos emplear la macro con ampersand para obtener un carácter especial, no obstante, en algunos casos es necesario. Los más utilizados son:

[2]N. del T.
Válgame la traducción ;-) creo que es más intuible y corto que poner "símbolos de mayor/menor q ue"

[3]N. del T.
A partir de aquí me referiré a el por su denominación anglosajona, "ampersand"

- Usar & con el “ampersand” (&),
- Usar < para el símbolo *menor que* (<),
- Usar > para el símbolo *mayor que* (>),
- Usar &etago; para el el símbolo *menor que* con una barra inversa (</)
- Usar $ con el símbolo de dólar ($),
- Usar # con la almohadilla (#),
- Usar % con el símbolo de tanto por ciento (%),
- Usar ˜ con una tilde (~),
- Usar `` ` ` `` y `' '` con las comillas, o emplear &dquot con ".

3.3 Entornos Code y Verbatim

Mientras tratábamos el asunto de los caracteres especiales, debí mencionar también el “entorno” *verbatim* empleado para incluir texto literal en la salida (preservando los espacios, sangrías, y demás). El elemento `verb` se emplea para esto; tiene el siguiente aspecto:

```
<verb>
Un poco de texto literal a incluir como ejemplo en la salida.
</verb>
```

El entorno `verb` no te permite emplear *todo* en su interior literalmente. Específicamente, deberás hacer lo siguiente en el interior de los entornos `verb`.

- Usar &ero; para obtener un ampersand,
- Usar &etago; para obtener </,
- No usar \end{verbatim} dentro de un entorno `verb` ya que es así como lo emplea LaTeX para finalizar el entorno `verbatim` (En el futuro, podría ser posible ocultar el formateador de texto subyacente enteramente, pero el analizador no soporta esta característica todavía.)

El entorno `code` es mayormente como el entorno `verb`, a excepción de las líneas horizontales que son añadidas al texto que enmarca, como aquí:

```
He aqui un entorno code de muestra.
```

Deberás usar el entorno `tscreen` antes de cualquier entorno `verb`, como aquí:

```
<tscreen><verb>
He aqui un poco de texto de ejemplo
</verb></tscreen>
```

`tscreen` es un entorno que únicamente sangra el texto y selecciona como fuente tipográfica por defecto `tt`. Esto hace que los ejemplos tengan mejor aspecto, tanto en las versiones LaTeX como ASCII. Se puede usar `tscreen` sin `verb` de todos modos, pero si se incluye algún carácter especial en el ejemplo, será necesario emplear ambos. `tscreen` no procesa los caracteres especiales. Ver `ejemplo.sgml` para los ejemplos.

El entorno `quote` es similar a `tscreen`, con la excepción de que no se selecciona como fuente tipográfica por defecto `tt`. Por tanto, puedes emplear `quote` para menciones que no requieren interacción de la computadora, como en:

```
<quote>
He aqui una muestra de texto a sangrar, como en una cita.
</quote>
```

que generará:

> He aqui una muestra de texto a sangrar, como en una cita.

3.4 Estructura Global del Documento.

Antes de meternos en profundidad con los detalles, describiré la estructura global de un documento definido por el DTD `linuxdoc`. Echar un vistazo a `Linuxdoc-Ejemplo.sgml` para ver un buen ejemplo de cómo definir un documento.

3.4.1 El Preámbulo

En el "preámbulo" del documento, se definen aspectos como la información del título, y el estilo del documento. Para un documento de tipo Howto[4] tendría este aspecto:

```
<!doctype linuxdoc system>

<article>

<title>Linux Foo-HOWTO

<author>Norbert Ebersol, <tt/norb@baz.com/

<date>v1.0, 9 March 1994

<abstract>
Este documento describe como usar las herramientas foo para frobnicar
librerias menganitas, empleando el relincador xyzzy...
</abstract>

<toc>
```

Los elementos deberán ir más o menos en ese orden. La primera línea comunica al analizador SGML que emplee el DTD Linuxdoc-SGML. La etiqueta `<article>` fuerza al documento a seguir el estilo "article". El DTD QWERTZ original define "report" [5] y "book"[6] también; no he retocado éstos para su uso con Linuxdoc-SGML.

Las etiquetas `title`, `author`, y `date`[7] deberían resultar obvias. La etiqueta `date` incluye el número de versión y la última fecha de modificación del documento.

La etiqueta `abstract` define el texto que será impreso en el encabezamiento del documento, *antes* de la tabla de contenidos. Si no se va a incluir una tabla de contenidos (la etiqueta `toc`), probablemente no sea

[4]N. del T.
COMO
[5]N. del T.
Informe
[6]N. del T.
Libro.
[7]N. del T.
Título, Autor y Fecha, respectivamente.

necesario un elemento `abstract`. Sugiero que todos los Linux HOWTOs sigan este mismo formato para el preámb ulo, para que el título, resumen, y tabla de contenidos estén todos ahí y tengan una apariencia similar.

3.4.2 Párrafos y Secciones

Tras el preámbulo, estamos listos para sumergirnos en el documento. Disponemos de los siguientes comandos de seccionamiento:

- `sect`: Para secciones de nivel principal (como 1, 2, y así en adelante)
- `sect1`: Para secciones secundarias (como 1.1, 2.1, y demás)
- `sect2`: Para secciones de tercer nivel
- `sect3`: Para secciones de cuarto nivel
- `sect4`: Para secciones de quinto nivel

Éstas son a *grosso modo* equivalentes a sus respectivas en LaTeX `section`, `subsection`, y demás.

Tras la etiqueta `sect` (o `sect1`, `sect2`, etc.) vendrá el nombre de la sección. Por ejemplo, al comienzo de este documento, tras el preámbulo, viene la etiqueta:

```
<sect>Introduccion
```

Y al comienzo de esta sección (Párrafos y Secciones), está la etiqueta:

```
<sect2>Parrafos y Secciones.
```

Tras la etiqueta de sección, se comienza el cuerpo de la sección. En cualquier caso, se debe empezar el cuerpo con una etiqueta `<p>`, como sigue:

```
<sect>Introduccion
<p>

Esta es una guia de usuario del sistema de procesado de documentacion
linuxdoc-sgml ...
```

Esto es así para comunicar al analizador que se ha terminado con el título de la sección, y que estás listo para comenzar con el cuerpo. A partir de ahí, los párrafos siguientes se comienzan con una línea en blanco (tal y como se haría en TeX). Por ejemplo,

```
He aqui el final del primer parrafo.

Y aqui comenzamos este nuevo.
```

No hay razón para usar etiquetas `<p>` al comienzo de cada párrafo; únicamente al comienzo del primer párrafo tras un comando de seccionamiento.

3.4.3 Finalizando el documento

Al final del documento, se debe emplear la etiqueta:

```
</article>
```

Para comunicar al analizador que hemos finalizado con el elemento `article` (que engloba al documento completo en conjunto)

3.5 Referencias Cruzadas

Ahora iremos con otras prestaciones del sistema. Las referencias cruzadas son fáciles. Por ejemplo, si queremos definir una referencia cruzada a cierta sección, necesitarás etiquetar esa sección con un nombre, como sigue:

```
<sect1>Introduccion<label id="sec-intro">
```

Ahora ya se podrá referir a esa sección en cualquier lugar del texto empleando la expresión:

```
Ver seccion <ref id="sec-intro" name="Introduccion"> para una introduccion.
```

Esto sustituirá la etiqueta `ref` con el número de sección etiquetado como `sec-intro`. El argumento `name` para `ref` es necesario para transformaciones a groff y HTML. La definición de macro de groff que emplea Linuxdoc-SGML no soporta referencias cruzadas, y algunas veces es preferible referirse a ella por su nombre en lugar de por su número.

Por ejemplo, esta sección es la 3.5.

Existe también un elemento `url` para los “Universal Resource Locators”, o URLs, empleados en los World Wide Web. Este elemento deberá ser empleado para referirse a otros documentos, ficheros disponibles para FTP, y demás. Por ejemplo,

```
Puedes obtener los documentos HOWTO en
<tt><htmlurl url="http://sunsite.unc.edu/mdw/HOWTO/"
name="The Linux HOWTO INDEX"></tt>.
```

El argumento `url` especifica el URL actual en sí mismo. Un enlace al URL en cuestión será añadido actualmente al documento HTML.

El argumento opcional `name` especifica el texto que será anclado al URL (para la conversión HTML) o que dará nombre como descripción del URL (para LaTeX y groff). Si no se incluye ningún parámetro `name`, se empleará el propio del URL.

Por ejemplo, puede obtener el paquete Linuxdoc-SGML de

`ftp://sunsite.unc.edu/pub/Linux/utils/text/linuxdoc-sgml-1.5.tar.gz`.

Una variante muy útil de este `htmlurl`, que suprime la creación de la parte correspondiente al URL excepto en HTML. Lo cual es muy útil para cosas como las direcciones e-mail personales; se puede escribir:

```
<tt><htmlurl url="mailto:esr@snark.thyrsus.com"
name="esr@snark.thyrsus.com"></tt>
```

y obtener “`esr@snark.thyrsus.com`” en la salida a texto mejor que el repetitivo “`esr@snark.thyrsus.com` <`mailto:esr@snark.thyrsus.com`>” manteniendo la validez en los documentos URL y HTML.

3.6 Fuentes (Tipográficas)

En esencia, las mismas fuentes soportadas por LaTeX lo son por Linuxdoc-SGML. Nótese de todos modos, que en la conversión a texto ASCII (vía groff) se prescinde de la información relativa a fuentes. Por tanto, podrán emplearse las fuentes todo lo que se quiera, con el beneficio correspondiente en la conversión a LaTeX. Pero no basarse en las fuentes para hacer comprender algo concreto en la versión ASCII.

Particularmente, la etiqueta `tt` descrita anteriormente puede ser empleada para obtener una fuente de tipo “máquina de escribir” de anchura constante, que deberá emplearse en todas las direcciones e-mail, nombres de máquina, de fichero, etc.

Por ejemplo:

```
He aqui una muestra de <tt>texto tipo maquina de escribir </tt> para
ser incluida en el documento.
```

El equivalente:

```
He aqui una muestra de <tt/texto tipo maquina de escribir/ para ser
incluido en el documento.
```

Recuérdese que sólo se puede hacer uso de este método abreviado si el texto englobado no contiene barras.

Se pueden conseguir otras fuentes; `bf` para obtener **negrita** y `em` para obtener *cursiva*. Bastantes más tipos de fuentes son soportados también, pero no las recomiendo, porque al convertir esos documentos a otros formatos como el HTML podr ían no ser soportados. La negrita, tipo máquina de escribir, y cursiva[8] deberían ser todas las necesarias.

3.7 Listados

Existen varios tipos de listados soportados. Son:

- `itemize` para listados marcados con puntos como este
- `enum` para listados numerados
- `descrip` para obtener listados "descriptivos"

Cada elemento o ítem de un listado `itemize` o `enum` deberá ser marcado con una etiqueta `item`. Los ítems en `descrip` son marcados con `tag`.

Por ejemplo:

```
<itemize>
<item>He aqui un item.
<item>He aqui un segundo item.
</itemize>
```

Que tendrá este aspecto:

- He aqui un item
- He aqui un segundo item

O, para un `enum`,

```
<enum>
<item>He aqui el primer item.
<item>He aqui el segundo item.
</enum>
```

Ya se coge la idea. Los listados pueden contener niveles también; ver el documento de ejemplo para más detalles.

Un listado `descrip` es ligeramente diferente, y también ligeramente más feo, pero podrías querer emplearlo en ciertas ocasiones:

[8] N. del T.
Boldface, `typewriter` e *italics* en Inglés.

```
<descrip>
<tag/Gnats./ bugs raros que vuelan por el ventilador de tu disipador.
<tag/Gnus./ bugs raros que corren por tu CPU.
</descrip>
```

que acaban teniendo este aspecto:

Gnats.
: bugs raros que vuelan por el ventilador de tu disipador.

Gnus.
: bugs raros que corren por tu CPU.

3.8 Más Información.

- The QWERTZ User's Guide está disponible en: `ftp://ftp.cs.cornell.edu/pub/mdw/SGML`.

  QWERTZ (y por tanto, Linuxdoc-SGML) tiene bastantes prestaciones como fórmulas matemáticas, tablas, figuras, y demás. No recomiendo emplear la mayoría de estas prestaciones en los HOWTOs sobre LiNUX debido a que no saldrían bien en las versiones de texto ASCII.

  Si deseases escribir documentación genérica en SGML, sugiero emplear el DTD QWERTZ original, en lugar del "retocado" Linuxdoc-SGML, que he modificado para el uso en particular con los HOWTOs de LiNUX y documentación similar.

- Las herramientas originales QWERTZ de Tom Gordon pueden obtenerse en: `ftp://ftp.gmd.de/GMD/sgml`.

- Más información acerca de SGML puede encontrarse en las siguientes páginas WWW:

  1. `SGML en la web`
  2. `Página Web del SGML`

- El analizador de James Clark's `sgmls`, su sucesor, `nsgmls` y otras herramientas pueden encontrarse en: `ftp://ftp.jclark.com` y en *James Clark's WWW Page* `<http://www.jclark.com>`.

- Puedes unirte a la lista de correo de Linuxdoc-SGML enviando correo a:

  `majordomo@via.ecp.fr` con `subscribe linuxdoc-sgml` en el cuerpo del mensaje. La dirección de correo es: `linuxdoc-sgml@via.ecp.fr`.

- Se puede obtener más información sobre `LyX` en:

  `LyX WWW Page`. `LyX` es un procesador de texto de alto nivel, interface al LaTeX, quasi-WYSIWYG[9], que genera muchos estilos y diseños casi automáticamente. Acelera el aprendizaje del LaTeX y hace que diseños complicados sean fáciles e intuitivos.

4 Anexo a la Traducción al Castellano.

4.1 Aspectos Importantes del SGML-LaTeX en Castellano

4.1.1 Tildes, Eñes, Etc.

Para que aparezcan las tildes, eñes y demás caracteres tipográficos típicos del castellano que *no* sean "caracteres especiales" basta como escribirlos como tales en el fuente (para ello, deberás cargar el mapa de

[9]N.del T.
WYSIWYG : "What You See Is What You Get", algo así como "lo que ves es lo que vas a obtener"

teclado español, `es.map`, cosa q ue se suele configurar con el programa de instalación de LiNUX), que el editor los soporte (para `vi` no hacen falta más configuraciones) y pasarle el parámetro '-l' al script de conversión utilizado; (actualmente, sólo soportan esta opción `sgml2la tex`, `sgml2html`, y `sgml2txt`).

4.1.2 Tamaño del Papel.

Este no es un aspecto crucial, pero no está de más saberlo. Para que los documentos a LaTeX se formateen en un tamaño de papel A4, típico en nuestro país, basta con que al script de conversión se le pase el parámetro '-a'. También se puede especificar est o a `dvips` con la opción '-a4'.

4.1.3 Los títulos de las Tablas de Contenidos, Bibliografía, etc.

Este era un aspecto bastante molesto del LaTeX, el que se generasen automáticamente los títulos de las macros `toc`, `lof`, `lot`, `biblio`, etc... con los títulos en inglés; esto está solucionado empleando el paquete *babel* que suele veni r incluido con LaTeX, y que en lo que a nosotros atañe, da soporte a castellano, catalán y gallego.

Para hacer uso del mismo basta con ejecutar `texconfig`, y configurar las "*hyphenation tables*", generalmente editando desde el shell de configuración de `texconfig` el archivo `languaje.dat`, en el que se ha de descomentar el patrón perten eciente al español; al salir de `texconfig` tendréis el LaTeX configurado para funcionar en castellano. Personalmente, dejo solamente el patrón perteneciente al español, y no tengo problemas, no sé si dará problemas dejando varios.

Después es necesario editar el fichero `&LINUXDOCLIB/rep/latex/mapping` y añadir en las definiciones de cada DTD (article, etc), "spanish", de modo que quedaría así:

```
     <qwertz>        +
     </qwertz>       +

     <article>       + "\\documentstyle\[linuxdoc-sgml,isolatin,spanish,qwertz,[OPTS]\]{article}\n"
                             "\\input{epsf.tex}\n"   +

     </article>      +       "\\end{document}"       +

     [...]
```

5 Anexo: El INSFLUG

El *INSFLUG* forma parte del grupo internacional *Linux Documentation Project*, encargándose de las traducciones al castellano de los Howtos (Comos), así como la producción de documentos originales en aquellos casos en los que no existe análogo en inglés.

En el **INSFLUG** se orienta preferentemente a la traducción de documentos breves, como los *COMOs* y *PUFs* (**P**reguntas de **U**so **F**recuente, las *FAQs*. `:)`), etc.

Diríjase a la sede del INSFLUG para más información al respecto.

En la sede del INSFLUG encontrará siempre las **últimas** versiones de las traducciones: `www.insflug.org`. Asegúrese de comprobar cuál es la última versión disponible en el Insflug antes de bajar un documento de un servidor réplica.

Se proporciona también una lista de los servidores réplica (*mirror*) del Insflug más cercanos a Vd., e información relativa a otros recursos en castellano.

Francisco José Montilla, `pacopepe@insflug.org`.