Procesadores de texto: estúpidos e ineficientes

Allin Cottrell[*]

4 de junio del 2003

Índice general

1. El alegato

El procesador de textos es una herramienta estúpida e ineficiente para preparar trabajos en equipo. Éste es el alegato que defenderé a continuación. Probablemente le parecerá grotesco a primera vista. Si estoy en contra de los procesadores de texto: ¿qué alternativa propongo? ¿escribir a mano o con una máquina de escribir? No. Aunque hay argumentos a favor de estos tipos de escritura presupongo que los lectores de este ensayo harán la mayor parte de su trabajo en un ordenador, al igual que yo. Mi proposición es que hay alternativas mucho mejores, dentro de la informática, que un procesador de textos.

[*]Traducción: José María Martín Olalla

El texto del alegato es intencionadamente provocativo, pero aclaro: cuando se dice: los *procesadores de textos* son estúpidos no se está diciendo que lo sea el *usuario*. Se está criticando la tecnología, especialmente una promovida por el mayor vendedor de software y que se ha convertido *de facto* en un estándar. A no ser que estuviese en el lugar adecuado en el momento adecuado probablemente desconozca la existencia de alternativas. Las alternativas nos se promocionan por los grandes distribuidores de software por una buena razón: son gratuitas.

Empecemos por el principio. El texto ideado para comunicar ideas e información se propaga de varias formas:

1. Como una copia impresa
2. En forma digital: correo electrónico, páginas web, documentos para visión en pantalla

Hay un cierto solapamiento en esta clasificación. Por ejemplo, un documento para imprimir puede ser distribuido electrónicamente en la esperanza de que el receptor será capaz de imprimirlo. Aun así consideraremos estos dos modos por separado.

2. Copias impresas

Queremos escribir un documento en el ordenador y que se imprima correctamente en la impresora. Por supuesto no es necesario que esto ocurra al tiempo (el texto en la impresora a la vez que se teclea en el ordenador). Queremos teclear primero, "grabar" digitalmente en algún medio adecuado. Después querremos recuperar el texto, editarlo a voluntad e imprimirlo cuando esto todo correcto. ¿Seguro que un procesador de texto —como el líder de ventas Microsoft Word— es la elección "natural" para hacer esto? Bueno, es una elección, pero no la mejor. ¿Por qué no?

2.1. Composición y tipografía

Preparar texto con un procesador fuerza a trabajar, a la vez, con dos hechos que son *conceptualmente* diferentes y que deberían estar separados en la práctica si queremos aprovechar mejor el tiempo y que el resultado final sea el mejor de los posibles. Los dos hechos son:

1. La composición del texto en sí misma. Es decir, la elección de palabras para expresar ideas y la estructura lógica del texto. Lo último depende de la naturaleza del texto. Incluye conceptos como la división del texto en párrafos, secciones o capítulos, la elección de si cierto material aparecerá como pie de página o dentro del texto principal, la presentación del partes del texto como citas en vez de palabras propias del autor, etcétera.
2. La tipografía del documento. Esto se refiere a conceptos como la elección del tipo de letra en la que se imprimirá el texto y la forma en la que se visualizarán elementos estructurales del texto. Irán las cabeceras de sección en negrita o como versalitas? ¿El texto estará alienado

a la izquierda o centrado? ¿Justificado o no? ¿Las notas aparecerán al pie de página o al final? ¿El texto irá en una columna o en dos columnas? Etcétera.

En principio el autor debería concentrase en el primer aspecto. Es ese su problema. Adam Smith señaló los grandes beneficios que se obtienen de la división del trabajo. Composición y estructura lógica del texto es la contribución específica del autor a la producción del texto impreso. Tipografía es el trabajo del tipógrafo. Esta división del trabajo era normal en la era anterior a la informática. El autor escribía e indicaba el editor la estructura lógica del texto por medio de anotaciones. El tipógrafo traducía las anotaciones en el documento impreso e implementaba el diseño del autor en un diseño tipográfico específico. Uno sólo tiene que imaginarse a, por ejemplo, Zorrilla preocupándose de las cabeceras de sección de su *Don Juan Tenorio* para observar lo ridículo que sería esto. Zorrilla fue buen un gran literato pero no un tipógrafo.

Puede pensar que nos hemos salido del argumento. Los escritos de Zorrilla se publicaban y los tipógrafos profesionales se interesaban en diseñarlo e imprimirlo. Usted y yo no somos tan afortunados; si queremos imprimir lo hacemos nosotros (y además, queremos hacerlo mucho más rápido que por el camino tradicional). Bien, sí y no. Lo hacemos nosotros (en nuestros ordenadores), pero tenemos mucha ayuda a nuestra disposición. En particular hay programas que realizan la tipografía de un documento de forma profesional. *Ese programa (o conjunto de programas) haría por nosotros, gratis y en segundos, lo que el tipógrafo hacía por Cervantes, Shakespeare, Zorrilla, Paz...* Sólo tenemos que decirle al programa las anotaciones apropiadas para que él sepa que hacer, justo como el método tradicional.

Por tanto, sugiero que deberíamos distinguir dos "momentos" de la producción de un texto en un ordenador. Primero un teclea el texto y lo estructura correctamente, indicando la estructuración vía anotaciones simples. Esto se consigue con un *editor de texto*, un tipo de software que no debe confundirse con un procesador de texto —posteriormente se explicará la diferencia en detalle.— Después, se manda a un programa de tipografía que en muy poco tiempo nos devuelve una maravillosa copia tipográfica.

2.2. Los demonios del WYSIWYG

Los dos hechos que señalé antes van acoplados en los modernos procesadores de texto WYSIWYG ("*What You See is What You Get" —lo que ves es lo que obtienes—).* Se teclea el texto y *a la vez,* el texto aparece en la pantalla con una representación tipográfica que corresponde, supuestamente, a lo que aparecerá cuando se mande el documento a imprimir (aunque por varias razones esto no siempre es así). En efecto el texto aparece continuamente formateado tal cual teclea. A primera vista esto es una enorme ventaja; un análisis más detallado revela que es una condena. Hay tres aspectos para explicar esto.

1. El autor se distrae de lo que es su preocupación de composición del texto en favor de elecciones tipográficas de las que no es experto.("jugar con tipos de letra y márgenes" cuando debería preocuparse del contenido).

2. El algoritmo de tipografía que emplea el procesador WYSIWIG sacrifica calidad por la rapidez requerida para que le texto aparezca, en tiempo real, tal y como el usuario introduce el texto. El resultado final es de interior calidad a aquel de un programa de tipografía.

3. El uso de un procesador de texto tienta al autor de perder la visión de la estructura lógica del texto y lo distrae con elementos tipográficos que son superficiales.

Los primeros dos puntos son autoexplicatorios. Expliquemos el tercero. (Su importancia depende del tipo de documento en cuestión).

2.3. La estructura del documento

Tomemos por ejemplo una cabecera de sección. En lo que se refiere a la estructura lógica del texto lo único que importa es "marcar" un trozo del texto como cabecera de sección. Por ejemplo se puede teclear

```
\section{Texto de la cabecera}.
```

Cómo la cabecera se implemente tipográficamente en la versión a imprimir es otra cuestión. Por contra, cuando se usa un procesador de texto, lo que ve es (¡todo!) lo que obtiene. Se fuerza a decidir qué tipografía aparecerá en la cabecera al mismo momento que ésta se crea.

Supongamos que se decide poner la cabecera en negrita y con un cuerpo mayor que el resto del texto. ¿Cómo conseguirlo? Hay muchas formas, para casi todo el mundo la forma más intuitiva (dado el contexto WYSIWYG en el que se mueve) es introducir la cabecera, seleccionar el texto, pulsar la tecla de negritas, y elegir el tamaño de texto apropiado. La cabecera está ahora en negritas y con un cuerpo mayor.

¡Bien! Pero ¿quién dice que esto es una cabecera? Nada identifica esta pequeña porción de texto como una cabecera. Supongamos que después decide que prefiere, mejor, tener la cabecera en versalitas, o numeradas con números romanos, o centrada, o lo que sea. Tendría que decir: "Por favor, haga tal y cual cambio en todas las cabeceras." Pero tendría que repetirlo a lo largo de todo el texto y modificar manualmente cada cabecera.

Ahora hay una forma de especificar una estructura lógica en (por ejemplo) Microsoft Word. Se *puede*, si se es cuidadoso, conseguir estos cambios con un solo comando. Pero pocos usuarios de Word explotan esto consistentemente, y no es sorprendente: los WYSIWYG no animan a preocuparse de la estructura lógica. Se puede fácilmente —muy fácilmente— "corromper" la estructura con comandos de formato de bajo nivel. Cuando, por contra, se usa un editor de texto, la necesidad de indicar una estructura lógica es inmediata.

2.4. Editores de texto

Vale, es hora de explicar qué es un editor de texto y en qué se diferencia de un procesador de texto. Un editor de texto moderno se parece a un procesador de texto. Es el concepto usual de menús o iconos desplegables

para funciones como abrir, grabar o cerrar archivos, buscar, sustituir, corrector ortográfico etcétera. Pero no tiene la funcionalidad tipográfica. El texto aparece en pantalla de forma nítida pero sin ánimo de pretender que represente el resultado final de la copia impresa del documento.

Cuando se graba un documento se hace en forma de *texto simple* lo que significa generalmente en "ASCII" (American Standard Character-Set for Information Interchange —Standard Americano del Conjunto de Caracteres para el Intercambio de Información—). El código ASCII se compone de 128 caracteres (lo que normalmente se conoce como un conjunto de caracteres de "7 bits" ya que requiere 7 dígitos binarios para su codificación: dos elevado a siete resulta 128). Incluye los dígitos del 0 al 9, el alfabeto inglés tanto mayúsculas como minúsculas y un conjunto de caracteres especiales. ASCII es el mínimo común denominador de la comunicación digital. Un mensaje en código ASCII es "entendible" por cualquier ordenador del mundo. Si se envía un mensaje en este código, sin duda el receptor verá lo que se ha escrito.

Por contra, cuando se graba un archivo de un procesador de texto contiene varios caracteres del "control", fuera del rango ASCII. Estos caracteres representan el formato que se ha aplicado (por ejemplo negritas o itálicas) además de varias clases de "material" interno relacionado con la mecánica del procesador de texto. No son universalmente inteligibles. Para entenderlos, se necesita una copia del procesador de texto con el que se creó el documento (o algún tipo de filtro conversor apropiado). Si se abre un archivo proveniente de un procesador de texto con un editor de texto se ve (además del texto o sus bits) un montón de "simbolitos extraños"; es el código binario de formato.

Puesto que un editor de texto no introduce código binario de formato, si se desea representar estas aplicaciones tales como itálicas, se necesita introducir unas *etiquetas*. Es decir se teclean "anotaciones" (usando también ASCII) que dicen al *programa tipográfico* que cierto texto se ponga en itálicas. Por ejemplo para el programa tipográfico LaTeX (más abajo se darán más detalles) se teclearía

```
\textit{lo que vaya a aparecer en itálicas}.
```

Realmente si se usa un editor de texto diseñado para facilitar el "trabajo" de LaTeX no tendría que hacer esto por sí mismo. Se teclean cierto tipo de secuencias de control cortas, se selecciona de un menú o se hace clic sobre un icono, y la anotación deseada aparecerá; la forma de teclear un documento ASCII preparado para LaTeX no es muy diferente de lo que ocurre en un procesador de texto.

2.5. Virtudes de ASCII

Elegir componer un texto en código ASCII usando un editor de texto y formatearlo con un programa específico tiene otras ventajas adicionales.

1. Portabilidad: como se dijo anteriormente, *cualquiera*, en un ordenador, puede leer el texto codificado incluso si no tiene medios de visualizarlo o imprimirlo. Por contra, un archivo del procesador de texto

Polilla 9.0 es incomprensible para un receptor que no tenga el mismo tipo y versión de tu procesador —a no ser que el receptor sea capaz de extraer el texto de la basura binaria que lo circunda.— Esto también se aplica al autor cuando se considera el paso del tiempo. Puede ser dificultoso leer un archivo de Polilla 8.0 en Polilla 9.0 o viceversa, pero nunca tendrá ese problema en ASCII.

2. Compacto: los archivos ASCII representa *ideas* y no un montón de códigos del procesador de texto. Para documentos pequeños, un archivo de procesador de texto puede ser incluso diez veces mayor que el correspondiente ASCII que contenga la misma información.

3. Seguridad: la elección "editor de texto más tipógrafo" garantiza que no tendrá problemas de corrupción del archivo (a no ser que se sufra un daño en el disco duro o algo similar). El texto fuente siempre está allí, incluso si el programa tipógrafo falla por alguna razón. Si usa un procesador de texto y *no* ha tenido problemas de corrupción de archivo ¡está de suerte!

2.6. El programa tipógrafo

Ahora pido un poco más de atención para el programa tipográfico. No entraré en detalles técnicos pero trataré de decir lo suficiente como para que tenga una idea sobre lo que estoy hablando.

El programa de tipografía básico que tengo en mente se llama TeX —así, tal y como está escrito— y fue escrito por Donald Knuth de la Universidad de Stanford. TeX es gratuito (se descarga vía Internet desde muchos servidores) y está disponible para cualquier tipo de ordenador. (Puede también comprar un CDROM con el conjunto completo de archivos TeX por un precio pequeño). Knuth comenzó a trabajar en TeX en el 1977; en el 1990 anunció que no seguiría desarrollando el programa, no porque hubiera perdido interés sino porque ya era esencialmente perfecto. Está tan libre de errores como cualquier otro programa y hace un trabajo espléndido tipografiando cualquier tipo de documento desde texto simple a grandes fórmulas matemáticas.

Me referí antes a LaTeX —así, tal y como está escrito.— si TeX es el motor básico para tipografía, LaTeX es un gran conjunto de macros, desarrollado inicialmente por Leslie Lamport en los ochenta y ahora mantenido por un grupo de expertos. Los macros hacen la vida sencilla para el usuario medio. LaTeX aún se desarrolla: paquetes y capacidades adicionales aparecen constantemente. Algunas utilidades de valor añadido también se desarrollan para TeX, tales como la que permite hacer documentos PDF (el "Portable Document Format" de Adobe) directamente desde los archivos ASCII. (Digo "bajo desarrollo" pero sólo quiere decir que está continuamente mejorando. Los programas ya son muy estable y buenos.)

La estructura y forma del documento se indica en LaTeX por medio de "anotaciones" tal y como se mencionó antes. Hay muchos libros (y guías en Internet) que explican con cierto detalle estas anotaciones, no voy a ir por ahí. Las anotaciones más comunes son simple y fáciles de recordar[1]

[1] Sobre todo si se sabe inglés (N. del T.)

además editores de texto diseñados para LaTeX (hay un montón) ofrecen gran ayuda.

Una de las propiedades atractivas de LaTeX es la posibilidad de cambiar la apariencia del documento drásticamente y *consistentemente* con pocos comandos. La estructura global del documento se controla por:

1. El "tipo de documento" —`document class`— que eliges (informe —`report`,— carta —`letter`,— artículo —`article`,— libro —`book`—
2. Los paquetes —`package`— que decides cargar.

Puede, por ejemplo, cambiar completamente la familia de tipos de letra (consistentemente a lo largo del texto, cabeceras de sección, pies de página y todo) y los tamaños usados cambiando sólo uno o dos parámetros en el "preámbulo" del documento. Igualmente puede poner todo en formato de dos columnas o rotarlo de la orientación normal a apaisado. Es posible hacer algo similar en un procesador de texto pero, generalmente, no es conveniente y probablemente se volverá loco e introducirá, sin querer, inconsistencias en el formato del texto.

Puede complicar un texto en LaTeX tanto como quiera. Lo más simple es especificar un tipo de documento y dejar que cargue las definiciones de los macros. Generalmente esto produce buenos resultados y la tipografía es mucho mejor que la que produce un procesador de texto. (Por supuesto, cosas como la numeración de capítulos, secciones, pies de páginas, referencias entrelazadas etcétera se tratan automáticamente) O puede ser más "intervencionista" y cargar paquetes adicionales (o crear sus propios paquetes) para controlar aspectos de la tipografía. Si esa es su elección puede conseguir resultados verdaderamente bonitos y exclusivos.

2.7. Uniendo

Ahora se trata de explicar como funciona todo esto. Si tiene una buena configuración de TeX es algo como lo siguiente: Introduce el texto en su editor favorito. Incluye las "anotaciones" requeridas directamente o el editor lo hace vía menús o botones. Cuando quiera ver como va la versión tipográfica elige una opción del menú o hace clic en un botón del editor e invoca al programa de tipografía. Otro elemento de menú o botón abrirá un previsualizador en el que se puede ver lo que aparecerá en la impresora. Y es generalmente un verdadero "WYSIWYG" —el previsualizador mostrará una representación muy exacta de lo que se obtendrá en la impresora.— Puede agrandar una zona, visualizar una página entera, y las posibilidad similares. Se envía esta salida a la impresora con otro botón o elemento de menú y se continúa editando.

Algo después querrá previsualizar el archivo que ha actualizado. Se vuelve a invocar el programa de tipografía. Esta vez no tendrá que invocar el previsualizador: si lo ha dejado corriendo se actualizará automáticamente. Cuando haya terminado una sesión de edición de texto puede borrar el archivo de la versión tipográfica si necesita espacio en el disco. Sólo es necesario conservar el archivo fuente en código ASCII; la versión tipográfica se crea fácilmente cuando se vaya a necesitar.

3. Propagando el texto por ordenador

La sección anterior se dedicó fundamentalmente a conseguir un resultado tipográfico adecuado. Hay otras consideraciones importantes cuando se prepara un documento si se tiene en mente la necesidad de hacer transmisiones digitales (correos electrónico, páginas web, etcétera).

Por ejemplo, tomemos el caso del correo electrónico. Si alguien quiere mandar un mensaje corto lo introduce en el cliente de correo que este usando, tanto si es un cliente "tradicional" basado en texto como Pine, o una interfaz gráfica como Netscape o Eudora. En este caso el mensaje irá, probablemente, en código ASCII (o quizás incluya HTML, i.e. Hyper-Text Mark-up Language, el lenguaje de las páginas web el cual se compone principalmente de código ASCII). Pero ¿y si se quiere mandar un texto más largo que ya ha preparado independientemente del cliente de correo?

Para esto frecuentemente se "adjunta" un documento de un procesador de texto. ¿Hay alternativa?

Tenemos que distinguir dos situaciones: ¿es el texto en cuestión relativamente corto y simple (un informe, carta, agenda, horario) o es más complejo (un artículo con muchas fórmulas, un libro)? La opción "ASCII más programa de tipografía" conduce a propuestas diferentes en según qué caso.

3.1. Documentos sencillos

Con documentos sencillos nos tenemos que preguntar: ¿necesita elementos de tipografía, tipos de letras y todo eso? No es más eficiente —el "más" en el sentido de una comunicación económica y efectiva— enviar un texto simple en código ASCII con el formato elemental que permite el código? Esto conserva el ancho de banda de la comunicación (recuérdese que el archivo del procesador puede ser *mucho* mayor que el de código ASCII) y asegura que nadie tomará en vano el esfuerzo realizado porque no tiene el Polilla 9.0. Puede adjuntar el ASCII a través de un fichero creado por un editor igual que lo hace con un documento creado por un procesador de texto, o simplemente pegarlo en el cuerpo del mensaje de correo electrónico (puesto que no es más que texto simple). Ya que el texto TeX no es más que código ASCII —y si hablamos de un documento sencillo no habrá muchas "anotaciones" y de haberlas serían autoaclaratorias[2]— y puede ser tratado de la misma forma.

3.2. Documentos complejos

Los documentos complejos y largos son muy fáciles de tratar en la forma tipográfica. Las fórmulas matemáticas son difíciles de escribir en ASCII y los diagramas e imágenes también. ¿Qué hace TeX en este asunto? He argumentado que el procesador de texto puede ser problemático porque el receptor puede no tener el Polilla 9.0. ¿No sirve esto en los dos casos? Incluso si llegas a probar TeX, ¿cuántos de sus amigos tendrán una instalación de TeX? Aquí hay algunas soluciones:

[2] Siempre que el receptor sepa inglés claro (N. del T.)

1. Convertir el archivo TeX en código HTML. Hay programas buenos para esto. (HTML y TeX pertenecen a una familia muy parecida de programas en la que se incluye diseño lógico del formato de texto, lo que posibilita la interconversión de formatos con gran fidelidad.[3]) Ahora, su receptor si puede leer el texto usando un navegador de Internet.

2. ¿Puede el receptor imprimir un archivo PostScript? Es muy probable que en una universidad o empresa así sea ya que dispondrá de una impresora PostScript. Si no, el receptor puede instalar el programa "ghostview" —gratuito— y hacer lo mismo con cualquier impresora. En tal caso se envía la versión tipográfica del documento en la forma de archivo postscript el cual se puede enviar a la impresora.

3. ¿Tiene el receptor el lector "Acroread" para archivos PDF? (También es gratuito) Entonces se le envía una versión PDF del documento tipografiado.

En la discusión anterior hemos tocado el punto de la preparación de una página web. Tenemos la opción de hacerlo en HTML directamente. Pero si no quiere hacer eso, puede escribir HTML usando una interfaz de usuario como Netscape Communicator, por ejemplo. También puede hacerlo usando MS Word (por cierto que crea un código HTML horroroso, lleno de elementos extraños que hacen difícil editarlo con otro programa). Si elige TeX se puede convertir a HTML de forma rápida, sencilla y limpia.

4. Limitaciones

He intentado hacer un fuerte alegato en favor de la opción "ASCII más programa tipográfico" en vez de procesador de texto. Admito, sin embargo, que ha *algunos* tipos de documentos para los cuales un sistema WYSIWYG es la herramienta natural. Pienso en documentos cortos muy específicos con gran cantidad de formateo de texto en vez de texto en sí: posters, invitaciones y similares. Podría hacer esto en TeX pero no sería lo más eficiente. El LaTeX estándar sería de poca ayuda. Y aunque LaTeX es muy hábil para manejar automáticamente el tamaño de los tipos de letra que se necesitan en un texto, no está diseñado para el tipo de fuentes divertidas que podría necesitar para un documento muy informal. La estructura lógica no es importante en este caso: sólo interesa la apariencia. Normalmente se desea saber, por ejemplo, "si pongo tal línea a 36pt, mandaría la última línea a la siguiente página, que es lo que no quiero que ocurra" Un sistema WYSIWYG es lo apropiado.

Si su trabajo es de este estilo, habría dejado de leer esto hacer mucho tiempo. Si su trabajo es la presentación de documentos formales, esta limitación poco importa.

[3] El código binario que utiliza un procesador de texto es una bestia totalmente diferente, interconversión entre TeX y un procesador de texto no es fácil. Además, como TeX es una máquina tipográfica superior, es imposible pasar de TeX a, por ejemplo, Word sin pérdida de información.

5. Estrambote

Quizá haya percibido que estoy un poco entusiasmada con el tema. Lo estoy. La razón no es sólo una cuestión de debate académico entre modos alternativos de preparar un texto. Es un conjunto de factores en el que el poder de los grandes vendedores de software sólo empujan de una parte. Por ser amables, afrontamos una situación en la que MS Word se convierte, para casi todo el mundo, en una suerte de estándar para la preparación de documentos usando un ordenador. Pero Word es un estándar que tiene que ofrecer más que el hecho de que *es* (o aspira a ser) un estándar.

Es un poco como el QWERTY. ¿Conoce la historia? ¿Por qué las teclas de un máquina de escribir (y también, por extensión, de los ordenadores) tiene QWERTYUIOP en la línea superior? No fue la disposición original de la teclas. Fue diseñada para un propósito muy específico: entorpecer a los mecanógrafos. El problema era que la habilidad de los primeros mecanógrafos rápidamente superaban las capacidades de las primeras máquinas de escribir: un mecanógrafo rápido podría atascar las teclas pulsándolas más rápidamente de lo que ellas tardaban en retornar después de golpear la cinta de tinta. QWERTY distribuye las teclas para entorpecer la mecanografía. Esto es absurdo en un teclado electrónico pero es muy tarde para cambiar: QWERTY es un estándar y todos los intentos de racionalizar han fracasado ante esta realidad.

De forma análoga, estoy argumentando que MS Word no tiene *derecho* a ser considerado un estándar en la preparación de documentos. Es pero que no sea demasiado tarde en este caso, que aún haya oportunidad de decir *No a Word.* Realmente, en cierto sentido Word es incluso peor que QWERTY: no es un estándar sino un "ascensorista" —un ¡trepa!.— El "estándar" de Microsoft para al representación binaria de documentos varía a voluntad de... Microsoft Corporation.[4] El cuasi-monopolio de MS Word se soporta sobre el cuasi-monopolio de Microsoft Windows (tema que no se trata aquí). Y como no están presionados por rivales comerciales Microsoft no tiene interés en establecer algún tipo de estándar para la representación binaria de formato de texto. Por el contrario, tiene gran interés en forzar la "actualización" de Word cada poco tiempo. Oh cielos, Word N.0 no leerá el documento que le mandé su compañero si lo escribe en Word (N+1).0. Bien, haría mejor en actualizar, ¿verdad? Incluso si ninguna de las nuevas características de la versión N+1 —suponiendo que las haya— sean de su interés.[5]

6. Referencias

Si ha llegado hasta aquí quizá le interesen más detalles sobre buenos editores de texto, TeX y todo lo demás.

[4] Imagínese el desastre que sería que el kilogramo patrón guardado en la Oficina Internacional de Pesos y Medidas —BIMP— variase a voluntad del BIMP que es el organismo encargado de garantizar que ese estándar *no varíe*. (N. del T.)

[5] En mi opinión —la de alguien que ha usado Word durante años antes de cambiar a TeX y que tiene interés en tipografía— no ha habido ninguna mejora digna de tal nombre en MS Word para Windows desde la versión 2.0 de 1990.

Lo mejor es empezar en un sitio de amigos de TeX, por ejemplo la página de TUG[6] (TUG is the TeX Users Group —Grupo de Usuarios de TeX—). Hay muchos enlaces útiles. Uno de los principales es la Red Completa de Archivos TeX, abreviado CTAN[7] de donde se puede descargar TeX, LaTeX y todo lo demás. Incluye el programa de tipografía y una gran colección de macros, previsualizadores, y software para imprimir archivos.

Los paquetes de TeX no incluyen, generalmente, un editor de texto así que necesitará uno (a no ser de que ya tenga uno). Hay muchas elecciones pero mi sugerencia es Emacs[8], junto con el paquete AUC TeX[9]. Esto hace que Emacs trabaje perfectamente con TeX y LaTeX. Los macros aparecen destacados y pueden verse errores a simple vista. Hay gran cantidad de comandos cortos y menús desplegables que facilitan la tarea.

Aquí puede ver una captura de pantalla[10] de una sesión de edición de TeX usando Emacs con AUC TeX (JPEG, 145345 bytes).

[6] http://www.tug.org
[7] http://www.tug.org/ctan.html
[8] http://www.emacs.org
[9] http://sunsite.auc.dk/auctex/
[10] http://ricardo.ecn.wfu.edu/ cottrell/emacs-screen.jpg