Artículos

Edición digital: formatos y alternativas

Por **Pedro Hípola** y **Ricardo Eíto Brun**

Pedro Hípola

Resumen: *La gran cantidad de formatos existentes en la actualidad para desarrollar documentos digitales lleva, en ocasiones, a analizar y evaluar profundamente las prestaciones que cada uno presenta. Su accesibilidad, portabilidad y métodos para su creación se convierten en puntos esenciales a tener en cuenta a la hora de elegir entre uno u otro. Esta situación trae consigo la búsqueda de estándares lo más ampliamente aceptados posible que permitan su máximo desarrollo en el futuro, y que los documentos que han sido ya creados no se queden incompatibles con los nuevos formatos. En este artículo se hace un repaso esencial a todos aquellos que en los últimos años han ido apareciendo, prestando especial atención a xml como uno de los formatos más interesantes desarrollados para internet y como posible sucesor de html.*

Palabras clave: *Edición digital, Xml, Sgml, Formatos de documento digital, Accesibilidad al documento digital.*

Ricardo Eíto Brun

Title: Digital publishing: formats and alternatives

Abstract: *The large variety of formats currently used for creating digital documents calls for a detailed analysis and evaluation of their functionality. Accessibility, portability and methods of creation are essential factors to consider when selecting a format. Choosing the appropriate format involves a search for a widely accepted standard to ensure that the documents won't become incompatible with new formats developed in the future. The present article reviews all of the formats which have appeared in recent years, with special attention given to XML, one of the more promising, created for internet and a likely successor to HTML.*

Keywords: *Digital publishing, Xml, Sgml, Digital document formats, Digital document access.*

Hípola, Pedro; Eíto Brun, Ricardo. *"Edición digital: formatos y alternativas". En: El profesional de la información, 2000, octubre, v. 9, n. 10, pp 4-14.*

Involucrarse de lleno en el mercado de la edición digital implica tomar decisiones sobre el formato de documento digital que se va a emplear. Durante mucho tiempo ha sido normal escuchar a los directivos de las empresas editoriales tradicionales preguntas como: ¿qué aplicaciones y formatos debemos utilizar?, ¿qué alternativas soportan con más facilidad y economía el ciclo de vida de los documentos?

Hasta mediados de los noventa la diversidad de los formatos existentes fue una auténtica fuente de obstáculos para que se pudiera llevar a cabo un intercambio fluido y económico de la información documental en soporte digital. La elección de uno u otro era crítica, ya que los productos de información digital podían llegar a quedar aislados del resto por no ajustarse a unas normas que permitiesen su gestión homogénea y aseguraran un mínimo de portabilidad. El acceso universal que iba a facilitar la difusión digital de contenidos podía verse comprometido y la inexistencia de estándares suficientemente aceptados podía conducir a la siguiente paradoja: las dificultades que tendrían los usuarios para acceder a la información serían directamente proporcionales al volumen de documentos disponibles en formato digital.

Hoy día la situación se ha clarificado bastante. Podríamos resumir la oferta de formatos de documento digital abierto (no propietario) y avanzado (más allá de las prestaciones elementales de normas sencillas tipo ascii) en los dos principales: primero parecía que iba a reinar PDF, luego triunfó html (y, por supuesto, dentro de los formatos propietarios hay casi una hegemonía del formato *Microsoft Word*). Con el predominio del estándar html es indudable que se ha abierto un nuevo estilo de trabajo en la tarea de conseguir la normalización del documento digital. De todas formas, al poco tiempo de generalizarse se hicieron manifiestas sus limitaciones. Para solucionarlas los fabricantes de navegadores desarrollaron extensiones propietarias del lenguaje incompatibles entre sí, con lo que volvió a surgir una nueva variante del problema del intercambio de información.

Original recibido el 11-1-00
Aceptación definitiva: 20-9-00

Entre las últimas soluciones propuestas, el lenguaje xml (*extensible markup language*) se presenta como una gran promesa que gradualmente va ganando adeptos, si bien es verdad que ha sido cuestionado en algunos sectores. Sea cual sea la situación actual, parece claro que queda siempre en pie un presupuesto fundamental: la necesidad de disponer de estándares en el campo de la edición digital lo más ampliamente aceptados posible. Dado que estamos en un proceso que evidentemente no ha llegado a su fin, creemos que puede ser interesante identificar las principales direcciones por las que se ha avanzado en estos últimos años, lo cual puede servir no sólo para resaltar los aciertos y errores cometidos sino también para intentar predecir, en la medida de lo posible, cuáles son las líneas en las que queda más trabajo pendiente de llevar a cabo.

«Frente a las alternativas basadas en marcas descriptivas, los documentos sgml pueden ser procesados fácilmente por la aplicación que los recibe, lee o interpreta»

En los siguientes apartados se describen algunas de las distintas alternativas disponibles para la edición digital en el camino hacia la normalización. Para ello hemos partido de un esquema que agrupa los avances ya conseguidos en nueve tendencias principales (v. *Information world en español*, n. 31, pp. 1-8):

~~La~~ opción semántica

~~Si~~stemas *DIP*

~~Do~~cumentos compuestos estáticos

~~Do~~cumentos compuestos dinámicos

~~El~~ paradigma hipertexto

~~Te~~cnologías para la interacción.

Conviene señalar que nuestro objetivo no es clasificar los diferentes tipos de formatos existentes, sino más bien destacar los hitos que se pueden observar en una evolución histórica caracterizada por *idas y venidas*, por muchos vaivenes. Y es evidente que no estamos hablando de grupos o tendencias excluyentes pues, por ejemplo, un mismo formato de documento digital normalmente explota características de más de una de estas tendencias. Además, *a lo largo de su ciclo de vida un único documento (entendiendo como tal un mismo contenido informativo) puede representarse en múltiples formatos para satisfacer distintas necesidades*.

En estas páginas también resaltaremos la utilidad de otra tecnología, la de los visores universales, que ofrece una vía más para abordar la integración de formatos heterogéneos.

La opción semántica

Con este término nos referimos a una serie de formatos que introducen etiquetas en el contenido del documento no sólo con el propósito de diferenciar sus partes significativas y hacer explícita su estructura sino también para asignar un valor semántico a cada una de sus partes, Su inclusión en el documento no es en sí el factor diferenciador de estos formatos. De hecho, todos los formatos digitales mínimamente avanzados intercalan marcas en los documentos. La principal diferencia está en el propósito con el que se añaden. Por ejemplo, en los utilizados por los procesadores de textos convencionales la mayor parte de ellas indican cómo se debe formatear el documento cuando se muestra en pantalla o se imprime. Es decir, casi nunca son representativas de su estructura ni suelen asignar un valor funcional a las diferentes partes y por supuesto son propietarias, definidas por el fabricante de cada sistema de edición.

El uso de marcas para asignar valores semánticos a las partes que componen un fichero informático no es precisamente una novedad reciente. Las etiquetas que identifican los campos de una base de datos son un ejemplo de este método de trabajo que cuenta ya con décadas de explotación.

Dentro de los sistemas que se basan en el etiquetado de los campos es necesario hacer una mención especial a la estructura de un formato clásico en el entorno bibliotecario: *Marc* (*machine readable cataloguing*), esquema de registro empleado fundamentalmente para codificación e intercambio de registros catalográficos y que está regulado por la norma *ISO 2709*. Como es sabido, su origen se remonta al año 1965, cuando la *Library of Congress* aprobó un formato interno para la codificación de registros catalográficos legibles por ordenador: *Lcmarc*. El último paso en su evolución es el llamado *Marc21*, anunciado recientemente y conocido como el *Marc* del nuevo siglo. Entre las principales aportaciones de esta versión está la unificación de las versiones canadiense y norteamericana (*Canmarc* y *Usmarc*) así como la inclusión de una etiqueta que permite el acceso a los documentos digitales: el campo 856.

Pero, como se sabe, *Marc* no es propiamente un sistema de edición digital. Si lo hemos citado aquí es porque constituye un ejemplo internacional de uso de estándares para la estructuración e identificación funcional de las partes de un documento digital.

Sgml: el rey de la edición digital

Este lenguaje es, sin lugar a dudas, la iniciativa más emblemática en el camino hacia la normalización del documento digital y también es el formato que más explota lo que hemos denominado la *opción semántica*. Establece:

la sintaxis que se debe utilizar para diseñar un conjunto de marcas aplicables a cada tipo de documento, llamado *DTD* (*document type definition*), y

la forma en la que se deben intercalar marcas en el texto de un documento para identificar las distintas partes de su estructura.

> **«La generalización de la Red y de las intranets ha obligado a los fabricantes de los formatos de réplica a estudiar las posibilidades de integrarse con el lenguaje html»**

Cada tipo de documento tiene sus propias peculiaridades y necesidades de tratamiento. Sgml ofrece respuesta a este problema con la posibilidad de diseñar *DTD*s específicas para cada caso. Una *DTD* establece la estructura de un tipo de documento, de qué elementos va a constar, en qué orden deben situarse, qué valores deben recoger, posibilidad de ser repetidos y qué elementos pueden contener a otros. Por ejemplo, todos los documentos de tipo informe se codificarían de acuerdo a una misma *DTD*, las citas bibliográficas según otra, etc.

En sgml se utiliza el término "aplicación" para denominar a las *DTD*s creadas de acuerdo con las especificaciones de la norma. Los documentos que se escriben utilizando una *DTD* específica se llaman instancias de la *DTD*. Para que un sistema informático sea capaz de procesar un documento sgml es necesario distribuir, junto a su texto, la *DTD* que está utilizando.

Entre las ventajas que ofrece sgml hay que destacar su carácter no propietario, la independencia de aplicaciones software y de plataformas hardware específicas, la posibilidad de aplicar un control estricto sobre el contenido de los textos y la independencia entre el contenido textual del documento, su estructura y su presentación.

El origen del sistema se encuentra en el lenguaje de marcas *GML*, desarrollado por *IBM* en 1969 bajo la dirección de **Charles Goldfarb**. Junto a él participaron en el desarrollo **Edward Mosher** y **Raymond Lorie** (de las iniciales de sus apellidos procede el nombre del lenguaje). Las ideas detrás de la norma se encuentran en trabajos previos de **William Tunnicliffe**, presidente del *GCA Composition Committee*, y **Stanley Rice**.

La primera aplicación desarrollada para utilizar dicho sistema fue la *IBM document composition facility*, conocida como *script*, que incluía un conjunto de etiquetas genéricas predefinidas. Según **Truly Donovan**, este grupo de etiquetas iniciales hicieron que se confundiese la filosofía abierta del proyecto con un conjunto limitado de marcas.

Sgml evolucionó hasta convertirse en el estándar internacional *ISO 8879:1986 "Information processing-Text and office systems-Standard generalized markup language"*. Como prueba de la solidez de la norma se suele destacar el hecho de que no haya tenido que ser actualizada hasta 1994, ocho años después de su creación. Para que se produjera el «éxito» de sgml han desempeñado un papel trascendental dos hechos:

La decisión del *Departamento de Defensa* norteamericano de adoptar sgml en su proyecto *Cals* (*Computer-aided acquisition and logistic support*) para la representación de información textual, con lo que se obligó a los suministradores del ejército a utilizar este formato en las descripciones técnicas de sus productos. Ha tenido como finalidad mejorar y acelerar la cadena de suministros entre el ejército y las empresas que le facilitaban productos, con especial atención a la documentación técnica. *Cals* desarrolló y adoptó distintos estándares, entre ellos el lenguaje sgml para la representación de información textual y los formatos gráficos *Ccitt T.6 fax grupo 4* o *CGM* (*computer graphics metafile*) e *Iges* (*initial graphics exchange specifications*), diseñados los dos últimos para la representación de imágenes vectoriales.

El desarrollo y la amplia aceptación de html (derivado de sgml) y el www en 1994. Tanto el éxito inicial de este formato como el posterior reconocimiento de sus limitaciones han hecho que un número creciente de editores hayan puesto su atención en sgml, que hasta entonces había sido un sistema casi desconocido por lo compleja y costosa que resultaba su utilización.

A raíz de este resultado, parece que sgml ha entrado en un ciclo positivo: existe una oferta más amplia de aplicaciones a precios asequibles, y los productores de contenidos consideran seriamente la conveniencia de migrar hacia entornos de edición y publicación basados en las directrices que establece la norma. Pero ¿qué implicaciones tiene para un productor de contenidos aceptar la norma sgml y cómo afecta a los procesos de edición?

Sgml: ciclo de producción y herramienta

La creación de documentos sgml es costosa. En primer lugar, debe dedicarse tiempo en la fase de análisis de tipos documentales y desarrollo de *DTD*s. Incluso en el caso de que este paso se suprimiese adoptando un conjunto de *DTDs* existentes, el aprendizaje y la adaptación a un entorno de producción sgml resulta bastante trabajoso.

El proceso de intercalar etiquetas en los documentos es laborioso. La conversión automática de aquellos creados con procesadores de texto y aplicaciones de autoedición a documentos sgml utilizando las marcas de estilo del original no es suficiente y requiere la inclusión posterior de marcas para los elementos que no pueden ser inferidos a partir del formateo, si bien es verdad que algunas herramientas disponibles (*Dynatag* de *DBT*, *Context-Wise* de *US Lynx*, o *EasyTag* de *Tetrasys*) facilitan este proceso.

Siguiendo la categorización para trabajar con documentos sgml propuesta por **Goldfarb** [1997, p. 358], las herramientas de conversión se pueden clasificar en dos grupos:

—*n-converters*: conversores de documentos no sgml a documentos sgml, y

—conversores entre documentos sgml o que toman a éste como fuente y lo convierten a formatos no sgml.

Los ejemplos antes citados forman parte del primer grupo.

Una vez se han creado y se les ha añadido las marcas, es necesario proceder a su validación. Este paso se ejecuta con la ayuda de un analizador o *parser*, que comprueba la existencia de las entidades a las que se hace referencia en el documento, si éste cumple las restricciones indicadas en su *DTD* y si se adecua a las reglas de producción sgml. En el caso de que el documento sea válido, se puede proceder a su composición.

Con sgml es posible incluir referencias a recursos externos utilizando declaraciones de entidades. Como recursos externos se puede indicar la propia *DTD*, otros archivos sgml o archivos en formatos diferentes. Generalmente las entidades se utilizan para incluir archivos gráficos o multimedia, en unión con otro componente de los documentos sgml: las notaciones.

Pero no es éste el único objetivo de las entidades: también se utilizan para poder trabajar con cadenas de caracteres que serán reemplazadas por otras más extensas, o para utilizar directamente los códigos ascii correspondientes a caracteres que no están disponibles en el teclado desde el cual se edita el texto.

La finalidad de la composición es generar un documento susceptible de entregarse a la imprenta o distribuirse para su lectura en línea, ocultando sus marcas y aplicándole un formateo.

Las herramientas de edición de documentos sgml incorporan utilidades para diseñar páginas de estilo. Para una misma *DTD* se pueden diseñar diferentes hojas de estilo. La asociación entre una *DTD* y sus hojas de estilo puede hacerse de distinta forma. Algunas herramientas generan un único archivo compilado en el que se aplican las instrucciones de formateo indicadas en dicha hoja a cada elemento del documento. Otros visores son capaces de generar la presentación del documento sgml con el formato que establezca la hoja de estilo asociada de forma dinámica, sin un proceso de compilación previo.

Se han desarrollado normas que guían en la creación de hojas de estilo para documentos sgml: *Os* (*output format*), desarrollada en el marco del proyecto *Cals*, o la norma internacional *ISO 10179 Dsssl* (*document style semantics and specification language*). El propósito de *Dsssl* es más ambicioso que el de *Os*, ya que no sólo pretende resolver el tema del formateo de los textos sgml, sino que también especifica los procesos de conversión de documentos de una *DTD* a otra.

En la fase de distribución del documento es donde se pueden obtener las máximas ventajas de trabajar con sgml. Frente a las alternativas basadas en marcas descriptivas, los documentos sgml pueden llegar a ser procesados por la aplicación que los recibe, lee o interpreta. Por ejemplo, los distintos elementos se podrían utilizar para alimentar una base de datos relacional, bibliográfica, etc.

De cualquier forma, si bien sgml ofrece grandes ventajas, la dificultad que implica el proceso de creación de estos documentos y la alta curva de aprendizaje a la que están asociados constituyen los principales escollos que deben afrontar las organizaciones para adoptar este formato.

La opción semántica no sólo está presente en sgml, sino que también constituye buena parte de la base de otros sistemas, como *ODA/Odif* (*open document architecture/open document interchange format*). Y es la filosofía subyacente en las transacciones *EDI* (*electronic data interchange*) que utilizan mensajes (tampoco en este caso se trata de un ejemplo típico de edición digital) ajustados normalmente a las normas *Edifact* (*ISO 7372* y *9735*) ó *Ansi X.12*.

Sistemas *DIP*

Paralelamente a los modelos que hacen uso de la opción semántica, se fue desarrollando una serie de programas que se englobaron bajo el nombre *DIP* (*do-*

cument imaging processing). Se trata de softwares que han jugado un papel importante en una fase de la ofimática, pues sirven para la gestión de archivos digitales con las imágenes escaneadas de los documentos manejados por las diversas organizaciones.

A lo largo de la historia, las empresas involucradas en este ámbito (*FileNet*, *Olivetti*, *Xerox*, *Wang*, etc.) han hecho uso de diferentes formatos gráficos, algunos ajustados a normas; otros desafortunadamente propietarios, lo cual les hizo incompatibles, con los consiguientes problemas para ulteriores migraciones de software. Según el formato utilizado, se hace mayor o menor uso de sistemas de compresión. *Tiff* (*tagged image file format*) es el que ha alcanzado una mayor popularidad en la representación de los documentos mediante imágenes de tipo *raster*, aunque podríamos incluir otros como *jpeg* (*join photograph experts group*), *gif* (*graphics interchange format*), etc.

Como es natural, los documentos, además de texto, pueden contener tablas, gráficos vectoriales e imágenes, pero con la peculiaridad de que todos estos componentes constituyen un único archivo y no pueden tratarse separadamente. Esto presenta serias dificultades: cada página escaneada conforma un archivo independiente, por lo que si el documento consta de varias resulta necesario agruparlas de alguna forma (guardándolas en un mismo directorio o ensamblándolas utilizando alguna aplicación que ofrezca esta funcionalidad). La imagen escaneada debía ser procesada por un *OCR* (*optical character recognition*) para poder tratar sobre su contenido y había que asignar una serie de propiedades identificativas y descriptivas que se almacenaban en una base de datos externa al documento.

Documentos compuestos estáticos

Actualmente la representación del documento como imagen puede considerarse un precedente histórico de los formatos de réplica o portables: *Acrobat-PDF*, *Envoy*, *DigitalPaper*, etc. Sus proveedores han dado un gran paso adelante con la comercialización de productos que permiten obtener una representación en su formato propietario que va más allá del mero resultado de escanear un documento impreso.

La necesidad de este tipo de formatos hizo surgir el estándar *ODA/Odif* recogido en la norma *ISO 8613*. Su historia es la historia de un fracaso: comenzó a elaborarse en 1982, y entonces las "O" iniciales correspondían a *Office* (más tarde *Open*). Su objetivo era conseguir que "los documentos, tanto textuales como gráficos, pudieran ser transferidos, con todos sus atributos intactos, de un sistema a otro para poder ser luego editados, procesados, almacenados, impresos y transmitidos." Se pretendía, en definitiva, solucionar el problema de comunicación entre formatos de procesadores de texto propietarios, cuya proliferación se veía en aquellos momentos (más que ahora) como una dificultad para el intercambio de documentos complejos. Usando *ODA/Odif* como formato de comunicación intermedio, el sistema que recibiera un texto convertido desde otro software podría presentarlo en pantalla y gestionarlo como si él lo hubiera creado.

La norma recibió un fuerte impulso en 1991, fecha en la que se creó el *ODA Consortium*, constituido por *Bull*, *DEC*, *IBM*, *ICL*, *Siemens Nixdorf*, agregándose después otros como *WordPerfect*. El consorcio desarrolló un conjunto de herramientas software, que se distribuyeron gratuitamente, para transformar documentos al formato de intercambio *Odif*, para que a continuación se pudieran transferir de nuevo esos documentos a cualquier otro sistema que diera soporte a *ODA*.

A diferencia de sgml donde el énfasis se centra en la representación del contenido y la estructura del documento— *ODA* codifica, además de la estructura lógica del documento, el formateo que se debe aplicar para su impresión y visualización en pantalla. La norma define un modelo de documento constituido por tres estructuras: la lógica que podría ser equiparable a la *DTD* de sgml, la física grupos de páginas, páginas, marcos y bloques, y el contenido propiamente dicho, el cual constaría de texto (codificado de acuerdo con los sistemas *ISO 8859* e *ISO 6937*), imágenes tipo *bitmap* (*raster*) y gráficos vectoriales. Para la representación de gráficos se adoptó la utilización de los formatos *CGM* (normalizados en *ISO* y *Ccitt fax* grupos *3* y *4*).

Esta complejidad obligó a desarrollar distintos perfiles de la norma, denominados *document application profiles* (*DAP*), entre ellos el *Fod11*, *Fod26*, *Fod36*, *Fod112*, etc. Los podríamos definir como niveles de conformidad con la norma estandarizados y compatibles entre sí. A pesar de que su última revisión se hizo hace sólo cuatro años, el formato ha caído en el olvido más absoluto. La actividad del *ODA Consortium* cesó, y el soporte y desarrollo de productos acordes con la norma también se vieron paralizados. Quizá su complejidad fue una de las causas por las que *ODA/Odif* no consiguió alcanzar un nivel de aceptación suficiente entre los fabricantes de software. También es verdad que el predominio logrado por *Microsoft* y sus aplicaciones de sobremesa en los últimos años fueron los principales enemigos para su éxito. Nos encontramos ante un ejemplo que demuestra que, en el ámbito de la normalización, no siempre alcanzan el éxito las mejores normas y que en su adopción juegan un papel importante otros factores como la posición en el mercado de sus impulsores.

Ante tal fracaso, las empresas especializadas en software para la gestión de documentos digitales se enzarzaron en una carrera para ver quién conseguía imponer como estándar su propio formato de intercambio. Así surgió en 1993 *Acrobat*, un sistema con el que *Adobe* ha facilitado la difusión de documentos digitales entre sistemas software y hardware que son incompatibles entre sí. Haciendo uso de la tecnología *Acrobat* es posible ver en la pantalla del ordenador o imprimir (con gran fidelidad a la tipografía, estilo, gráficos y colores del original) documentos que pueden haber sido creados en un sistema completamente incompatible.

«*Adobe* era la única cuyo software de edición electrónica ha ocupado siempre un lugar destacado en su línea de productos, situándose el resto en un puesto secundario dentro de sus ofertas comerciales»

Además del formato propio de *Acrobat* (*PDF*) hay que citar *DigitalPaper* de *Common Ground* (antes *No-Hands Software*), *Envoy* de *Tumbleweed*, *Replica* de *Farallon*, etc., que ya han sido abandonados. Estos formatos, denominados también *de réplica*, son legibles en distintas plataformas hardware y sistemas operativos, siempre que se disponga de un lector especial distribuido (generalmente de forma gratuita) por su fabricante. Normalmente contienen dentro de un solo fichero textos, tablas, gráficos, etc., haciendo uso de sistemas de compresión. Permiten la presentación formal definitiva (en pantalla o impresión) y el intercambio (portabilidad) de los materiales para su visualización.

Los documentos de réplica se crean mediante un proceso de conversión, utilizando un software especial que actúa como un controlador (driver) de impresión. Para generar un documento basta con *imprimirlo* desde la aplicación con la que se creó (*Microsoft Word*, *WordPerfect*, *QuarkXpress*, *FrameMaker*, *Excel*, etc.) tras haber seleccionado el controlador correspondiente desde el cuadro de diálogo que contiene las opciones de impresión.

El documento resultante mantiene las características tipográficas y el formateo del original, y todos los objetos que lo componen se fusionan en un único archivo comprimido, con lo que se minimiza el espacio de almacenamiento requerido y se facilita su transmisión a través de redes. El formato también almacena información sobre las fuentes utilizadas en el documento original. En el caso de que el ordenador desde el que se lee no disponga del tipo de letra utilizado al crearlo, el software lector es capaz de emularla de forma razonablemente satisfactoria. Tras la conversión es posible añadir al documento tablas de contenidos, crear índices en texto completo, hiperenlaces, etc., así como indicar restricciones de seguridad. Esta edición se lleva a cabo utilizando aplicaciones proporcionadas por el fabricante del formato.

Para leer y editar los documentos es necesario disponer de un software propietario. Ésta es una de las características más importantes de los formatos de réplica: sólo pueden leerse o imprimirse si se dispone de un programa especial. Algunos sistemas, como *Envoy Run-Time*, *Replica Viewer*, o *MiniViewer* de *Common Ground*, fusionaban los documentos con el visor en un único archivo ejecutable, con lo que no es necesario poseer el visor para poder leerlo.

Las posibilidades de edición disponibles con este tipo de formatos son muy limitadas (eliminación e inserción de páginas extraídas de otro archivo en el mismo formato, creación de notas e hiperenlaces, etc.) y no suelen estar disponibles en los lectores de libre distribución.

En la lista de formatos compuestos estáticos se podría incluir también *RealPage*, desarrollado y utilizado por la editorial inglesa *Catchword* que, desde 1994, se dedica en exclusiva a la edición digital de publicaciones académicas. La orientación de negocio de la empresa difiere de la del resto de productores de formatos de réplica: no pretende vender software, sino servicios de *hosting* (alojamiento) para editoriales que faciliten la distribución electrónica de documentos, gestionando el proceso de conversión a partir de los documentos suministrados por los editores y su distribución a través de servidores dedicados, así como el control y la facturación de los accesos a las publicaciones. *Real Page* requiere un visor especial, disponible únicamente para *Windows* y a través de un applet *Java* para cualquier navegador internet.

El impacto de internet en los formatos de réplica

La generalización de la Red y de las intranets ha obligado a los fabricantes de los formatos de réplica a estudiar las posibilidades de integrarse con el lenguaje html. La adaptación al nuevo entorno se ha centrado en las siguientes líneas de trabajo:

Optimización de los formatos para lograr representaciones de los documentos que ocupen menor espacio, para así facilitar su distribución a través de la Red.

Posibilidad de crear enlaces que tengan como destino documentos html.

Desarrollo de *plug-ins* y controles *ActiveX* que permitan leer documentos en formato propietario desde cualquier navegador internet.

Distribución de documentos a través de servidores web página a página: enviándolas una tras otra a medida que las solicita el software cliente, sin que sea necesario descargarlas todas para poder comenzar a leerlo. Esta posibilidad se lleva a cabo mediante scripts *CGI* ejecutados por el servidor web a través del cual se accede a los documentos.

Este sistema de distribución fue adoptado inicialmente por *Adobe* en su especificación *Amber*, que constituyó la base para la versión 3.0 liberada el año 1996. Posteriormente lo adoptaron otros fabricantes.

Adobe y sus competidores

En el apartado anterior hemos citado formatos portables desarrollados por distintas compañías. Hasta hace poco tiempo podía considerarse a algunos de ellos alternativas al que había logrado una mayor popularidad: PDF, que siempre ha ocupado una posición predominante. De las empresas citadas, *Adobe* era la única cuyo software de edición digital ha ocupado siempre un lugar destacado en su línea de productos, situándose el resto en un puesto secundario dentro de sus ofertas comerciales. Por ejemplo, la línea de trabajo de *Farallon* y *Hummingbird*, recientemente adquirida por *PC Docs* se centró en el desarrollo de software para la conectividad entre sistemas hardware heterogéneos.

«Con sgml se pueden crear hiperenlaces que permiten recorrer el documento de forma no lineal. Pero hay una limitación: sólo es posible crear hiperenlaces que unan dos partes del mismo archivo»

En 1998 *Adobe Acrobat* y PDF dejaron de tener competencia: *Farallon* abandonó la tarea de desarrollar y comercializar *Replica*; *Hummingbird* había hecho pública su decisión de hacer lo mismo con *Common Ground* en febrero de ese mismo año y *Tumbleweed* (encargada de desarrollar *Envoy*) orientó su actividad hacia la distribución segura de documentos, recomendando el uso de PDF, al que definió como estándar de facto en una nota de prensa publicada en octubre de 1997.

Esto no debe interpretarse como una superioridad técnica del formato PDF frente al resto de sus competidores. Simplemente se trata, nuevamente, de una mayor aceptación debida sobre todo a la posición de superioridad que *Adobe* tenía en el mercado, gracias en buena parte a la importante implantación del estándar de facto *PostScript*, que había sido lanzado por la compañía en 1985.

Formatos para el intercambio

La necesidad de intercambiar documentos entre aplicaciones diferentes, y en la mayoría de los casos incompatibles, ha sido uno de los principales problemas que ha tenido que afrontar la aplicación de las tecnologías informáticas dentro de la gestión documental.

Para resolverlo se han diseñado distintos formatos. Algunos de ellos han sido propuestos por los fabricantes de procesadores de texto o herramientas de autoedición, como por ejemplo *RTF* (*rich text format*) de *Microsoft*, *MIF* (*maker interchange format*) de *Adobe* o *CDA* (*compound document architecture*) de *Digital*. A excepción de este último, que proponía un modelo más complejo y sofisticado, en el resto de casos se trata de formatos que descartan la utilización de las características de los originales que los hacen incompatibles, a favor de un modelo de representación más simple para el que se pueden diseñar filtros de conversión con mayor facilidad.

Estos formatos están pensados para ser reconocidos por aplicaciones distintas a la aplicación con la que fueron creados (siempre que incorporen un filtro de conversión entre ellos), ser convertidos al formato propietario para poder ser editados y de nuevo poder ser almacenados en el formato de intercambio.

Con los formatos orientados al intercambio:

es posible siempre la edición de un documento,

la conversión es bidireccional (contrariamente a lo que sucede con los formatos portables, en los que sólo hay conversión en una dirección), y

no es necesario disponer de una aplicación especial para leer los documentos, ya que su objetivo es que el resto de aplicaciones sean capaces de interpretar el formato.

El paradigma hipertexto y el www

Los sistemas hipertexto son una de las asignaturas pendientes de la edición digital. El término paradigma y conceptos como *transclusión*, *lectura no secuencial*, etc., han contribuido a crear unas expectativas mayores a lo que podríamos considerar tan sólo como una inestimable ayuda para la lectura de grandes textos: la posibilidad de unir sus partes mediante enlaces. De hecho, las implementaciones reales de sistemas hipertexto distan mucho de las concepciones visionarias auguradas por sus precursores en la década de los cincuenta y sesenta: **Vannevar Bush**, **Ted Nelson**, etc.

Bajo la denominación hipertexto encontramos distintas aplicaciones bastante dispares y basadas en formatos o lenguajes de marcas propietarios capaces de ser interpretados por lectores especiales: *HyperCard*, *Folio*, etc. Con sgml se pueden crear hiperenlaces que permiten recorrer el documento de forma no lineal. Pero hay una limitación: sólo es posible establecer hiperenlaces que unan dos partes del mismo archivo. Para crear un vínculo entre dos zonas de un documento sgml es necesario definir su origen y su destino. El origen se marcará utilizando un elemento especial, al que se llamará por convención *link* o *xref*, que contará con un atributo *ref* de tipo *idref*. Este atributo tomará como valor el nombre que se haya dado al punto destino del enlace, el cual se declara como un atributo con nombre *target* de tipo *id*.

Un parser identificará los orígenes del hiperenlace y comprobará si existen las declaraciones de los destinos a los que apuntan. Si se necesita crear uno que una dos archivos, es preciso recurrir a recursos ofrecidos por la norma *HyTime*, aprobada como estándar en 1986.

Pero desde el año 1992 hablar de hipertexto equivale casi a hablar del web, del lenguaje html y sus derivados. Html es una aplicación del lenguaje sgml que indica cómo se deben codificar documentos para distribuirlos en el www. Independiente de plataformas hardware o software, html parecía ser la solución idónea a los problemas de intercambio de documentación digital. Pero pronto se hicieron manifiestas sus lagunas:

~~in~~capacidad para mantener las características tipográficas y el formateo de los documentos,

~~el~~evados costes en ancho de banda y dificultades en la transmisión de documentos compuestos,

~~ca~~rencia de fórmulas de compresión asociadas al formato,

~~in~~existencia de mecanismos de acceso a la información (tablas de contenidos, índice de palabras clave, etc.).

Por otra parte, la evolución de html en los últimos años ha estado condicionada por la presión ejercida por *Netscape* y *Microsoft*, fabricantes de los navegadores internet más utilizados. El *W3C*, encargado de normalizar y controlar la evolución de los estándares para el www, se vio desbordado por los desarrollos de estas dos empresas, que añadían nuevas etiquetas al formato aprobado como estándar y diseñaban módulos y lenguajes de script que sólo podían ser interpretados por sus respectivos navegadores.

Para resolver estas limitaciones se han diseñado distintas alternativas: el lenguaje html dinámico (*dhtml*), el desarrollo *html-help* (propietario de *Microsoft* y propuesto al *W3C* para su inclusión en el estándar) y el lenguaje xml. Esta última opción se perfila como la alternativa más fuerte en el panorama de la edición digital y acapara la mayor atención por parte de fabricantes de software y usuarios.

«Frente al conjunto de etiquetas predefinidas que conforman html, xml ofrece la posibilidad de definir nuevas marcas para codificar la estructura y contenido de los distintos tipos de documentos»

1. *Dhtml*. El término *dhtml* se utiliza para hacer referencia a una serie de características soportadas por la versión 4 del *Internet Explorer* y *Netscape Communicator*. Algunas de estas particularidades se tomaron de los borradores de la que iba a ser la nueva versión de html, la 4.0, cuyo desarrollo se llamó *Proyecto Cougar*.

El objetivo final de *dhtml* se centra en lograr una mayor interacción en las páginas html, minimizando los intercambios de datos entre el cliente y el servidor web, y permite:

~~un~~ mayor control sobre los elementos que conforman una página html, mediante un modelo jerárquico de descripción de documentos (*document object model*) y lenguajes de scripts,

~~po~~sibilidad de aplicar formato al contenido de una página y, lo que es más importante, modificar el formato desde el navegador en respuesta a acciones ejecutadas por el usuario,

~~co~~ntrol de la posición de los objetos en una página y posibilidad de desplazarlos por ella, y

~~ca~~pacidad para cambiar su contenido una vez ha sido descargada del servidor.

Para lograr estas mejoras se han reunido distintos desarrollos: las hojas de estilo (*cascading style sheets*) que especifican cómo se deben formatear y posicionar los elementos en una página y lenguajes de script capaces de interpretar la jerarquía de objetos *DOM*.

Sin embargo, la normalización no se ha alcanzado. *Netscape Navigator* e *Internet Explorer* no utilizan modelos idénticos para las hojas de estilo y siguen existiendo distintos lenguajes de script: *Vbscript* y *Jscript* (de *Microsoft*) y *Javascript* (de *Netscape*), aunque en este punto se ha alcanzado cierta homogeneidad a partir de la decisión de la *Ecma* (*European Computer Manufacturers Association*) de adoptar este último como estándar, y la apuesta de *Microsoft* a favor de su versión de *Javascript* en detrimento de *VBscript*.

A pesar de que se trata de un gran avance, *dhtml* es un formato excesivamente orientado hacia la presentación de páginas y a los efectos visuales, con serias dificultades para representar el contenido semántico y la lógica de la información.

Html compilado: *html-help*

El excesivo tamaño de los documentos html y de sus componentes (gráficos, animaciones, etc.) así como la necesidad de aplicar mecanismos de acceso, ha llevado a *Microsoft* a proponer un formato propietario: *html-help*, que consiste en páginas html compiladas en un único archivo. Su lanzamiento se anunció en la *WinHelp Conference* de 1996, y la primera versión de prueba (llamada beta 1) se distribuyó a través del web site de *Microsoft* el 27 de noviembre de 1996.

El formato permite realizar búsquedas en texto completo y ofrece las opciones de navegación que caracterizaron el sistema de edición hipertexto *WinHelp*, desarrollado por *Microsoft* para distribuir ayuda en línea para los programas que se ejecutasen en el sistema operativo *Windows* (tablas de contenidos, palabras clave, etc). *Html-help* está concebido como el nuevo formato para la distribución de documentación en línea para aplicaciones software, sustituyendo a *WinHelp* e incluso al que había sido el sistema utilizado por *Microsoft* para la edición digital de grandes volúmenes de documentación: *Media View*.

Pero *html-help* presenta una limitación: sólo puede ser interpretado por equipos que dispongan del navegador *Internet Explorer.* A pesar de que puede considerarse como una alternativa válida para la edición digital y fue propuesto al *W3C* para su inclusión en el estándar html, su utilización parece quedar restringida a la publicación de documentación técnica de software.

Xml: el futuro de la edición digital

En menos de tres años ha pasado de ser una alternativa a html a convertirse en el formato con mayores perspectivas de acaparar el mercado de la edición digital y el desarrollo de aplicaciones internet/intranet. Su evolución comenzó en septiembre de 1996, auspiciado por el *W3C* con un claro propósito: diseñar un lenguaje de marcas optimizado para ser utilizado en internet, que combinase la simplicidad de html con la capacidad expresiva de sgml.

En su definición participaron *Microsoft*, *IBM*, *Sun Microsystems*, *Novell* y *Hewlett Packard*. La versión 1.0 fue ratificada por el *W3C* en la conferencia sobre sgml/xml celebrada en Washington en diciembre de 1997.

Las principales características de este formato son: extensibilidad, formateo, gestión avanzada de los hiperenlaces y modularidad.

1. Extensibilidad. Xml es un perfil de sgml y no una aplicación, como es el caso de html. Frente al conjunto de etiquetas predefinidas que conforman html, xml ofrece la posibilidad de definir nuevas marcas para codificar la estructura y contenido de los distintos tipos de documentos. Al igual que sucede con sgml, es posible crear *DTD*s que indiquen las reglas que debe cumplir cada tipo de documento y que faciliten su tratamiento automatizado por cualquier aplicación capaz de interpretar la *DTD*.

La viabilidad de marcar los elementos informativos permite procesar el documento en el puesto cliente. Xml, unido al lenguaje *Java*, hace realidad la utilización de los navegadores como clientes universales válidos para cualquier tipo de aplicación.

2. Formateo. Ofrece un mecanismo que permite indicar cómo se deben formatear, imprimir y mostrar en pantalla los distintos elementos de un documento. Estas hojas de estilo se diseñarían de acuerdo con las especificaciones *Xsl* (equivalente a las *cascading style sheets* del lenguaje html).

3. Gestión de hiperenlaces . *Xlink* es el modelo propuesto en xml para la gestión de hiperenlaces. El tratamiento es similar al utilizado por sgml, aunque también se incorpora la utilización de URLs característica de html. Las principales mejoras frente al modelo usado en html son: la posibilidad de trabajar con enlaces bidireccionales, con vínculos que, partiendo de un mismo origen, alcancen dos o más puntos destinos y la transclusión del destino del enlace en el documento origen.

4. Modularidad. Un documento xml puede ser modular gracias a la noción de entidades heredada de sgml y podrá estar formado por distintos archivos xml que se presentarán en pantalla como si se tratase de un único documento.

Xml y sgml

Muchas de estas ventajas, quizás con la excepción del tratamiento de los hiperenlaces, ya se encontraban en sgml. ¿Por qué xml y no sgml?

El primero es una versión simplificada del segundo, del que se han eliminado aquellas características poco utilizadas que añadían una complejidad en ocasiones insalvable para los fabricantes de software que afrontan el diseño de software para la edición con sgml.

La distribución de documentos se simplifica al ofrecer la opción de distribuir o no la *DTD* que utiliza un documento junto a éste. El motivo es justificado: según el tratamiento que se quiera hacer en el ordenador receptor del documento, la transmisión de la *DTD* y los procesos de validación pueden suponer mayores tiempos de transferencia y costes de ancho de banda innecesarios. Conviene señalar que xml es compatible tanto con sgml como con html y en un futuro las aplicaciones internet harán un uso conjunto de ambos.

«En un entorno docucéntrico los formatos y la estructura de almacenamiento interno de los componentes de un programa son transparentes para el usuario»

Actualmente los fabricantes de aplicaciones para la edición digital ya soportan xml: *DBT*, *Dataware*, *NextPage*, *ArborText*, *Interleaf*, etc. *Microsoft* incorpora xml a su navegador y ofrece un parser escrito en *Java* (*Msxml*) para comprobar la validez de los documentos. Ya contamos con gran cantidad de aplicaciones que hacen un uso intensivo del nuevo formato: por ejemplo, *RDF* (*resource description format*) para la descripción de documentos disponibles en el web; *CDF* (*channel description format*), desarrollado por *Microsoft* para transmitir información a través de los canales push del *Internet Explorer*; *MathML* (*mathematical markup language*); *OFX* (*open financial exchange*) para el intercambio de datos financieros; *OSF* (*open software description*), creado por *Microsoft* y *Marimba* para describir paquetes de instalación de aplicaciones software...

Tecnologías para la interacción

En los párrafos anteriores hemos hecho un resumen de los principales tipos de formatos disponibles para la edición digital. A continuación dedicamos dos apartados a las tecnologías utilizadas para la interacción y compatibilidad de documentos digitales: *OLE* (*object linking and embbeding)* y *Opendoc*, para la creación y edición de documentos dinámicos; y los visores universales, capaces de mostrar documentos en distintos formatos sin necesidad de hacer ninguna conversión previa.

1. Documentos compuestos dinámicos. Están formados no sólo por texto, sino también por gráficos, tablas, componentes multimedia, etc., creados con distintas aplicaciones y ensamblados posteriormente. Los componentes que lo conforman se almacenan en archivos independientes que pueden ser editados con distintas utilidades y reutilizados en un número ilimitado de documentos. Los cambios que se hagan en cada componente se reflejarán en el documento del que forman parte. De ahí la denominación dinámico.

Conviene aclarar que no es un formato especial para guardar y distribuir documentos, sino unas tecnologías que permiten ensamblar componentes en distintos formatos (textual, gráfico, etc.) y el usuario podrá editarlos sin necesidad de abrir la aplicación capaz de interpretar su formato.

Estas tecnologías están vinculadas a las interfaces gráficas de usuario y fueron el desencadenante de lo que se llamó sistemas operativos docucéntricos. Este término se utiliza para indicar que el documento, y no las aplicaciones, es el protagonista de la informática personal y de la ofimática, como confirma la metáfora del escritorio utilizada por *Macintosh*, por *Windows* y por las interfaces de usuario características de los sistemas de gestión documental (*Documentum*, *Docs Open*, *FileNet*, etc.). En un entorno docucéntrico los formatos y la estructura de almacenamiento interno de los componentes de un programa son transparentes para el usuario.

Las dos tecnologías más importantes para la integración de componentes en documentos compuestos dinámicos son *OLE*, de *Microsoft*, y *Opendoc*, desarrollada inicialmente por un consorcio del que formaron parte *IBM*, *Apple*, *Lotus*, *Adobe* y *Novell-WordPerfect*.

Este grupo de empresas creó los *CILabs* (*Component Integration Laboratories*), encargados de desarrollar las especificaciones del modelo. Entre éstas se encuentra el subsistema de almacenamiento (*Bento* de *Apple*), el de intercambios de mensaje entre objetos (*System Object Model* de *IBM*), lenguajes de scripts e incluso una interface que permite incrustar objetos *OLE* en documentos *Opendoc* (*componentGlue*), etc.

De nuevo la preponderancia de *Microsoft* ha marcado sus destinos: los creadores del estándar *Opendoc*

se vieron obligados a dar soporte a la tecnología propietaria de la competencia, para pasar a continuación a plegarse ante el éxito de *MS OLE* y abandonar el desarrollo de *Opendoc*.

OLE es la base del modelo de documento compuesto dinámico utilizado por *Microsoft*. Su origen se remonta a 1991 y surgió por la necesidad de incluir gráficos creados con *Microsoft Graph* en presentaciones *PowerPoint*. *OLE* ofrece dos mecanismos de integración de componentes en un documento compuesto: embbeding y linking. En el primer caso el componente sólo puede ser accedido desde el documento contenedor, mientras que en el segundo tiene existencia independiente. La tecnología *OLE* ha alcanzado un gran impacto y ha evolucionado hasta convertirse en la infraestructura tecnológica por excelencia para la informática basada en componentes y la orientación a objetos.

2. Visores universales. Los fabricantes de documentos digitales han desarrollado visores que permiten leer e imprimir los documentos creados con sus aplicaciones, por ejemplo *FrameViewer* de *Adobe* para *FrameMaker* o *WorldView* de *Interleaf*. Estos visores, aunque sólo permiten trabajar en modo lectura con un único formato, pueden considerarse una alternativa a los lenguajes de réplica si se trata de distribuir documentos digitales que hayan sido creados con una aplicación propietaria.

Incorporan utilidades para la compresión y encriptación. Frente a estos visores asociados a un único formato, los universales no requieren la conversión previa a uno propietario (interpretan directamente los archivos fuente), mantienen las características tipográficas del original y se han convertido en un componente clave en los sistemas de gestión documental y en un complemento de gran utilidad para las aplicaciones de mensajería digital y los navegadores internet. En esta línea de productos se encuentra *KeyView Pro*, desarrollado por *FTP Software* y actualmente propiedad de *Verity*, o *Quick View Plus* de *DBT*, o *Autovue*.

Conclusiones

En los apartados anteriores hemos ofrecido una breve descripción de las iniciativas más importantes que han contribuido a la homogeneización de los formatos para la edición digital y a lograr mayores posibilidades de interacción entre aplicaciones. Uno de los principales objetivos del texto es presentar las tendencias observadas como alternativas complementarias.

A lo largo del ciclo de vida de un documento, y dependiendo del uso que se le quiera dar en cada fase, se pueden utilizar distintos formatos para aprovechar las ventajas que ofrece cada uno de ellos. Así, los sistemas para la gestión de documentos digitales aplican el formato original del documento para su edición, formatos de réplica en las fases de validación, o html para su distribución a través del web.

Algunos analistas auguran un futuro en el que sgml será sustituido por su versión simplificada, xml, al que califican como un formato integrador en el ciclo de vida de los documentos: desde su producción hacia su distribución. Otros se resisten a aceptar xml como una alternativa real a sgml.

Cuestionar la importancia de xml y su creciente protagonismo en los próximos años resultaría ingenuo. Sin embargo, la producción de documentos estructurados implica elevados costes, y xml ha de afrontar, en su camino hacia la supremacía, las mismas dificultades que han frenado el desarrollo de sgml a lo largo de los últimos años, si bien en un mercado ya mucho más dinámico. La sustitución de html por xml también parece algo lejana.

El protagonismo de xml hay que buscarlo en aplicaciones específicas que utilicen internet como red de transmisión y que requieran: tratamientos complejos de los documentos e integración de datos entre aplicaciones diferentes. El comercio electrónico es el ejemplo más significativo del tipo de aplicaciones que van a sacar partido de este formato.

En el suministro de documentos, xml va a tener dificultades para demostrar sus ventajas frente a la facilidad de creación y conversión asociada a los formatos portables, en gran parte debido a la eliminación de barreras entre editores, distribuidores y usuarios que limita las necesidades de procesamiento en local. No obstante, algunos proveedores de contenidos ya han empezado a utilizarlo en la difusión push de información para integrar fuentes internas con información externa (por ejemplo, *Dow Jones* con su *Intranet Toolkit*).

Xml, al igual que sgml, ofrece la ventaja de poder compartir información en formato estructurado para facilitar su procesamiento en distintos centros de tratamiento de datos o con aplicaciones heterogéneas. Si esta necesidad no existiera, trabajar con xml puede acabar convirtiéndose en algo tan costoso como sgml.

Pedro Hípola
phipola@ugr.es

Ricardo Eíto Brun
ricardo.eito@adecco.es