Documentos y lenguaje de marcado: conceptos, problemas y tendencias

Por **Mela Bosch**

Mela Bosch

Resumen: *Se presenta un panorama de la estructuración de información documental por medio de lenguajes de marcado. Se hacen notar dos líneas de problemas: de estándares de descripción y del mercado de software. Dentro de la primera se describe la sintaxis concreta de sgml, el subconjunto html y la extensión xml. Respecto a la segunda se tipifica la oferta en el mercado. Finalmente se considera el lugar del profesional de la información en este contexto, enfatizando su importancia para el desarrollo de la web semántica en general y, especialmente, en herramientas de apoyo a la misma, como las ontologías.*

Palabras Clave: *Estructuración de información, Información documental, Lenguaje de marcado, Organización del conocimiento, Web semántica.*

Title: Documents and markup language: concepts, problems and trends

Abstract: *This article presents a panorama of the documental information that is structured using markup languages. This subject presents two different issues: description standards and software market limitations. Within the first we find sgml concrete syntax, the html subset and xml. The second presents itself as a tipification of software products in the market. Having described these matters, the article reflects upon the role of the information professional in this context emphasizing his importance for the development of the Semantic Web in general and the tools that such Web needs, such as ontologies.*

Keywords: *Documental information, Knowledge organisation, Markup language, Semantic web, Structured information.*

Bosch, Mela. *"Documentos y lenguaje de marcado: conceptos, problemas y tendencias". En: El profesional de la información, 2001, noviembre, v. 10, n. 11, pp. 4-9.*

Introducción: una gran solución y grandes problemas

El desarrollo de los lenguajes de marcado cambió el panorama de la estructuración de documentos, a pesar de que no era su objetivo inicial. En la década de los años 80 se planteó como una forma de asegurar la transferencia de datos manteniendo aspectos de presentación.

Sgml fue desarrollado en sus orígenes (1969) por **Charles Goldfarb** con el nombre de *general markup language* (*gml*), y tenía como objetivo facilitar el intercambio de documentos en *IBM*. Permite definir estructuras tipificadas de documentos a partir de las reglas que los rigen, las cuales son expresadas físicamente en los documentos por medio de marcas (señales o códigos convenidos). Fue adoptado rápidamente para la circulación de documentos por el *Departamento de Defensa* de EUA y por la *Oficina de Publicaciones Oficiales* de la *Comunidad Europea*, ambos clientes de *IBM* (**Bryan**, 1998). En una meteórica gestión (ya que ni siquiera fue previamente autorizado como estándar americano), *ISO* lo lanzó como la norma *ISO 8879* en 1986 con el nombre de "standard generalized markup language" (sgml).

Su desarrollo no se produjo de manera anecdótica, ya que la historia ha marcado dos de las características de sgml: está pensado para sistemas de oficina, es decir, para documentos corporativos, y está orientado al procesamiento de datos. Más concretamente dentro de este ámbito, al intercambio.

Si se observa la portada de la norma *ISO 8879* los descriptores indican claramente su contenido: *Data processing, Documentation, Logical structure, Programming (computers), Artificial languages, Programming languages* (*ISO*, 1986).

Sgml es, por lo tanto, un lenguaje artificial útil para representar la organización lógica de documentos a efectos de procesamiento y programación. En los años siguientes apareció como la gran solución para la transferencia de datos documentales con una arquitectura lógica determinada. Sin embargo, a la vez se generaron grandes problemas: de estándares de descripción y de mercado de software.

Mela Bosch, *Profesora e investigadora de la Universidad Nacional de La Plata y profesora de la Universidad Nacional de Mar del Plata, Argentina.*
euris@sinectis.com.ar

Fecha de recepción 13-09-01
Aceptación definitiva: 2-11-01

Antes de puntualizar sobre estos temas agregamos una reflexión: tal y como puede deducirse de lo que se ha comentado hasta ahora sobre su origen, y según nos hace notar **Winograd** (de forma pionera hace más de 18 años), el problema de la descripción y representación de las estructuras conceptuales en medios informáticos estuvo ligado y determinado por las necesidades de proceso. Era más importante lograr similitud con el lenguaje de las máquinas que con el de las personas: "*In designing a programming language there are two potentially conflicting sources of criteria organization (things can be classed as alike because they are implemented with the same underlying mechanism, or because they have similarity for the language user). In most programming languages, the balance lies towards the implementation side*" (**Winograd**, 1983, p. 417).

«El desarrollo de los lenguajes de marcado cambió el panorama de la estructuración de documentos, a pesar de que no era su objetivo inicial»

Esto nos lleva a una discriminación conceptual más general: los lenguajes, a los que definimos como conjuntos de símbolos normalizados para la representación y comunicación, se dividen en: naturales, controlados y artificiales.

Los primeros son los que se crean en un contexto social y convencional por conjuntos de individuos. Incluyen tanto las lenguas de las comunidades humanas como las formas de comunicación animal (aún hay muchos que discuten su valor y otorgan sólo al ser humano la capacidad de poseer lenguaje). Por su parte, los controlados son porciones del lenguaje natural normalizados para fines específicos: entre éstos se encuentran los lenguajes documentales y las ontologías. Finalmente los artificiales son conjuntos de símbolos arbitrarios también para fines concretos, incluyéndose entre ellos: los de programación —destinados al procesamiento— y los lenguajes de descripción, como sgml, cuyo objetivo es la interpretación por parte de lenguajes de programación (**Isasi**, 1997).

Ésta es una mera simplificación expositiva que nos permitirá avanzar operativamente en el desarrollo de este trabajo; no ignoramos que existen formas combinadas complejas como los lenguajes de transcripción, los de restricciones y los lenguajes gráficos, entre otros.

Es importante no olvidar que con sgml nos encontramos ante un lenguaje de naturaleza artificial cuya meta es la descripción de información para facilitar su proceso; de donde surge el tema de qué es lo que se describe y para qué, produciéndose así un efecto de retroalimentación sobre el lenguaje. Este punto nos conduce al primero de los problemas que estamos analizando.

Problemas de estándares de descripción

Por lo que puede deducirse de todo lo comentado hasta ahora, el objetivo de sgml de permitir la transmisión de datos determina la forma de descripción. La ventaja es que se permite establecer grupos de códigos propios según la característica de la estructura documental. Esto da origen a varias líneas de cambios dentro del lenguaje de marcado:

—Sintaxis concreta.

—Subconjuntos.

—Extensiones.

1. Sintaxis concreta. En el campo de la lingüística se denomina sintaxis a las reglas que definen el significado de los códigos y a los nombres reservados usados por un lenguaje en particular. Al tratarse de un lenguaje de descripción, sgml necesita ajustarse al objeto de su descripción y, a la vez, mantener consistencia en códigos y denominaciones. La forma de lograrlo es por medio de dos tipos de sintaxis: la abstracta y la concreta.

La primera es usada para especificar cómo deben ser escritas tanto las declaraciones de sgml como las de tipo de documento. Por otra parte, nos encontramos con el conjunto de reglas utilizadas para definir cómo deben ser codificados documentos específicos. Una forma particular de sintaxis concreta, llamada en inglés *reference concrete syntax*, fue definida de manera explícita en la *ISO 8879*. Su objetivo es suministrar una referencia para que las sintaxis concretas eventuales la tomen como base. Se conoce como *international reference version* (*IRV*) y contiene los siguientes puntos:

—Códigos que deben ser ignorados (*shunned character number identification*).

—El conjunto básico de caracteres *baseset* definidos por la norma *ISO 646*.

—Códigos ascii homologados.

—La ampliación de ese conjunto *descset*.

—Códigos de caracteres de funciones (*function character identification*).

—Reglas de denominación para cuando se crean entidades o etiquetas (*naming rules declaration*).

—Conjunto de delimitadores alternativos de marcado (*delimiters declaration*).

—Convenciones de denominación de declaraciones (*reserved name use*).

—Las cantidades permitidas para el anidamiento de elementos y entidades.

Esta sintaxis es asumida automáticamente por los sistemas que procesan sgml. Para ello es necesario indicarlo de la siguiente forma: *syntax public "iso 8879-1986//syntax reference//en*"

A esta sintaxis pueden agregarse esquemas propios de codificación. La sintaxis concreta de referencia debe ser entonces conocida y comprendida por sus usuarios potenciales, sean humanos o máquinas. Los cambios en las opciones por defecto se deben hacer de acuerdo con ciertas reglas, declarando públicamente (es decir de forma explícita) sus variantes (**Bryan**, 1998).

Existen tres formas de especificar las variedades de sintaxis concreta en un documento:

a. Por medio del uso de la opción *switches*: se indican puntualmente diferencias de la sintaxis concreta de referencia o de cualquier otra que se utilice. Permite establecer pares de caracteres que pueden ser cambiados. Por ejemplo, si se quiere indicar que en lugar de usar "[ ]" (códigos ascii 91 y 93, homologados *ISO 646*) se prefiere utilizar: "{ }" (cuya codificación es 123 y 125) se haría siguiendo el procedimiento que a continuación se expone:

syntax public "iso 8879-1986//syntax reference//en"

switches 91 123

93 125

b. Declarando públicamente que se usa una variante de sintaxis concreta:

syntax public "iso 8879-1986//syntax prueba//sp"

Con esto se informa que se emplea una sintaxis llamada *prueba* y que recoge el juego de caracteres en español. Ésta, de nueva creación, debe estar redefinida cubriendo todos los aspectos de la *IRV* mencionada anteriormente, asumiendo algunos puntos concretos y modificando un grupo particular que incluye, por ejemplo, entidades o elementos (**Bryan**, 1998). Con esto se elaboran sintaxis concretas orientadas a usos de algunas áreas temáticas y organizaciones.

c. Produciendo declaraciones de tipo de documento (*dtd*) que engloban las variantes de sintaxis respecto a la básica de sgml.

Las *dtd*s están normalizadas siguiendo la *ISO 12082*, que determina un marco de trabajo general. Además, nos encontraremos con otras *dtd*s que responden a iniciativas con fines específicos, como *Sgml initiative in health care* (*Hl7 health level-7 and sgml/xml*) o *National Center for Biotechnology Information* (*Ncbi*), de la *National Library of Medicine* y de los *National Institutes of Health* (**Cover**, 2000).

Un importante esfuerzo común es la *dtd* de *tei* (*text encoding initiative*) financiada por varias universidades y la UE, cuyo objetivo es construir una referencia universal para la codificación de textos (*Tei*, 2001).

Existen también otras *dtd*s valiosas para el profesional de la información como *ead* (*encoded archival description*) para documentos corporativos y de archivo, mantenida por la *Network Development and Marc Standards Office* de la *Library of Congress* (*LoC*) junto con la *Society of American Archivists.* Esta norma está en directa relación con la específica de marc: *marc dtd* (*machine readable cataloging document type definition*).

«El problema de la descripción y representación de las estructuras conceptuales en medios informáticos estuvo ligado y sobredeterminado por las necesidades de proceso»

2. Subconjuntos. Pueden ser más simplificados o más completos que las sintaxis concretas. Lo importante es que toman aspectos de la sintaxis abstracta. El más conocido es html, con versiones sucesivas que aparecen de forma continua y cuyo formato de intercambio se encuentra definido por la norma *ISO 9069*.

Existe también la *ISO 10744*, que desarrolla el *hypermedia/time-based structuring language* conocido como *HyTime*. Se trata de un subconjunto con una sintaxis propia que permite la representación en hipermedia vinculada a la evolución temporal. Incorpora técnicas que permiten asociar información adicional sin que haya sido almacenada como parte del documento fuente. *HyTime* puede referenciar cualquier texto, imagen, efecto sonoro o área espacial dentro de una publicación multimedia.

Un aspecto destacable es que hace posible que las direcciones de información se almacenen de manera independiente al sistema, lo que posibilita la gestión automatizada de ficheros. De esta forma se usa para verificar y cambiar direcciones cuando un documento se reestructura o es transferido a una nueva ubicación.

Esta norma define formas arquitectónicas de documentos, que en realidad no son más que reglas para crear y procesar los componentes de éstos. Se definen cuatro tipos de formas arquitectónicas en la *ISO/IEC*

10744:1997: de elementos, de atributos, de entidad de datos y de atributos de datos.

Así pues, se constituyen los *architectural form definition requirements* (*afdr*), es decir los requerimientos de arquitectura, cuyas especificaciones se pueden encontrar en el anexo 1 de *ISO/IEC 10744:1997*. Su uso se declara así:

<!afdr "iso/iec 10744:1997">

Existen, además, aplicaciones de *HyTime*, como *smdl* (*standard music description language*, *ISO/IEC DIS 10743:1995*).

3. Extensiones. La fundamental es xml, que puede entenderse como un dialecto de sgml (se suele usar el término dialecto porque contempla alteraciones estructurales respecto del lenguaje madre). Es una versión extremadamente simplificada de sgml (algo así como un sgml genérico) que admite el procesamiento en web de objetos, algo que no es posible hacer en html. Permite definir objetos como entornos de realidad virtual, movimiento, señales olfativas, etc.

En realidad es un metalenguaje, ya que permite a su vez crear subconjuntos. De hecho se está reescribiendo todo html en xml. En esencia lo que permite es una forma flexible, pero normalizada, de añadir etiquetas a los documentos. En html, que tal y como hemos visto es un subconjunto, las etiquetas son acotadas y tienen una semántica específica para la navegación hipertextual. Por su parte, sgml es excelente y completo, pero a la hora de implementar la navegación resulta complicado dado que los enlaces son "artesanales". Xml no sustituye a éste, que sí puede gestionar documentos xml, favoreciendo la funcionalidad de navegación y la manipulación de diferentes objetos no sólo textuales.

«Sgml es excelente y completo, pero a la hora de implementar navegación resulta complicado dado que los enlaces son 'artesanales'»

En su aspecto formal xml se nutre de reglas de producción lógicas para interpretar la sintaxis y permite su compilación. Define una regla básica y establece el valor de cada uno de los elementos.

En cuanto a sgml, los valores ya están establecidos en la sintaxis abstracta de sgml o en las *dtd*. Este lenguaje, al haber sido concebido para describir documentos de texto, estipula la forma de organización, o sea, no sólo la sintaxis sino la estructura. Por lo tanto no es posible poner un título de nivel 1 después de uno de nivel 3, por ejemplo, porque tiene restricciones para anidar etiquetas. Estas limitaciones, como ya indicamos con anterioridad, están en la *IRV*, de manera que hay que indicarlo cada vez que se modifican las restricciones y la sintaxis básica, o bien declarar una *dtd* con todos los cambios.

Xml no requiere una *dtd*, pero es necesario realizar otras especificaciones y declaraciones. Por ejemplo, se debe señalar quiénes son los usuarios: navegadores, etc. También en forma de declaraciones se deben indicar elementos, atributos, entidades, así como anotaciones para especificar datos externos.

Aunque se ha logrado normalizar la parte de xml correspondiente para libros, artículos y series con la norma *ISO 12083*, el problema es que en este momento nos encontramos en medio de un gran número y proliferación de variantes. Además, el subconjunto html se origina con extensiones que cambian según las ofertas de productos. Esto nos lleva a los problemas que analizaremos en el siguiente punto.

Problemas del mercado de software

Si bien es posible codificar textos sgml en cualquier editor de texto, la tarea artesanal es muy lenta y susceptible de que se produzcan errores. Se utilizan tres tipos de software para sgml: editores, analizadores (*parsers*) y visualizadores (*browsers*).

El editor es el que permite redactar *dtd*s de documentos. La gama oscila desde los más sencillos freeware —software gratuito— hasta los más complejos que permiten convertir texto procedente de otro origen; otros integran hojas de estilo y entremezclan sus posibilidades con análisis y visualización.

El analizador o *parser* compara las reglas de edición con la sintaxis concreta o la *dtd*. Trabajan a la par que el editor —o sobre documentos una vez compuestos— para verificar o modificar el marcado. Algunos pueden trabajar con hojas de estilo *dsssl* (*document style semantics and specification language*, *ISO 10179*) y *spdl* (*standard page description language*, *ISO 10180*). Las primeras permiten agregar abundantes detalles tipográficos mientras que la segunda está más orientada a la impresión.

Finalmente los visualizadores o *browsers* permiten recorrer el documento en toda su arborescencia. El líder en el mercado es *SoftQuad*, con *Xmetal*, cuya versión de prueba puede ser descargada de la web.

El panorama de editores, analizadores y visualizadores es amplio y confuso. Se crean subconjuntos, sintaxis concretas y *dtd*s de acuerdo con la conveniencia de las empresas, no existiendo además compatibilidad total entre ellos. Los más populares resultan los edito-

res html que permiten diseñar páginas web, entre los cuales encontramos *FrontPage*, de *Microsoft*, y *Composer*, de *Netscape*. Tienen la ventaja de que no es necesario conocer los lenguajes de marcado, pero no suelen ser totalmente compatibles entre ellos y con otros editores.

En cuanto a xml estamos en plena efervescencia. Su importancia estriba en que permite encapsular instrucciones de programación, ya sea *VBScript* o *Java applets*, compilar controles *ActiveX* y hasta lenguajes antiguos como *Cobol*. También tiene capacidades de edición, análisis y visualización, pero como además cuenta con posibilidades de programación, requiere un lenguaje especial para modelar los datos. Es aquí donde aparece *document object model* (*dom*) con adaptaciones para *Explorer* y *Netscape*. *Microsoft* desarrolló *com* (*component object model*) para el desarrollo de aplicaciones con xml.

En esta babel en la que nos encontramos, xml empieza a ser la *lingua franca* que permitirá crear y compartir documentos. Pero aún estamos lejos de esa situación: las ofertas de software se multiplican, los productos son costosos y la capacitación que requieren es grande (**Conallen**, 2001).

La web semántica, un futuro

La pregunta para los profesionales de la información es saber qué lugar les cabe en este panorama de estructuración de la información documental. Su posición puede estar, sin duda, en el apoyo que puedan prestar a los desarrolladores, trabajando en la definición de *dtd*s y sintaxis concretas adaptadas a las necesidades de cada corpus de información. En este sentido, el estudio y capacitación en este contexto es indispensable. Pero otro punto donde serán de inestimable valor es en el desarrollo de la web semántica.

Con el uso de los lenguajes de marcado, los sistemas informáticos en la www realizan tareas de rutina y también otras más expertas, siempre y cuando se trate de trabajar dentro de las páginas web. Sin embargo no se tiene aún un procedimiento fiable para procesar la semántica. Por esta razón se producen avances en lo que se está llamando web semántica.

En los párrafos que siguen haremos una reseña comentada del artículo de **Berners-Lee**, **Hendler** y **Lassila** titulado "The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities", publicado en *Scientific American* en mayo de 2001, desde la perspectiva de interés para el profesional de la información.

Según estos autores, la web semántica brindará contenido significativo a las páginas de la Red, creando un ambiente donde agentes de software, moviéndose de una página a otra, puedan fácilmente efectuar tareas sofisticadas para usuarios. Según ellos no será una red separada sino una extensión de la actual en la que la información tenga un significado definido, permitiendo que máquinas y personas puedan trabajar en cooperación. Estos autores nos dicen que para que la web semántica sea operativa los ordenadores deben tener acceso estructurado a colecciones de información y conjuntos de reglas de inferencia que ellos pueden usar para conducir al razonamiento automatizado.

En el documento mencionado se indica que los sistemas de representación de conocimiento (tal y como se denomina esta tecnología) han sido centralizados, requiriendo que todos compartan exactamente la misma definición de conceptos. Por otra parte limitan el tipo de preguntas que pueden formularse y, para evitar tales problemas, estos modelos de representación han tenido un propio y limitado conjunto de reglas para hacer inferencias sobre sus datos.

En cambio los autores dicen que el desafío de la web semántica está en proveer un lenguaje que exprese tanto los datos como una lógica para darles sentido de diferente origen y forma, valiéndose para ello de tres tecnologías: xml, *resource description framework* (*rdf*) y las ontologías.

—Xml, sobre el que ya hablamos, es importante en su doble carácter de metalenguaje de descripción que permite crear etiquetas, tanto propias como estándar, y a la vez como lenguaje de especificación de puntos de ejecución de porciones de software o programas completos que pueden hacer uso de ellas.

—*Rdf* (marco de descripción de recursos) es el complemento de xml. El documento que estamos reseñando nos indica acertadamente el etiquetado genérico xml, que permite agregar marcas arbitrarias a los documentos, las cuales no proveen semántica. Su significado podría ser expresado por el *rdf*, que son conjuntos de tríos —constituido cada uno de ellos por el sujeto, verbo y predicado de una frase elemental— y que pueden escribirse usando etiquetas de xml.

En síntesis, tal como plantean estos autores, *rdf* es un lenguaje de metadatos genérico que ofrece una manera natural para describir la mayoría de los objetos procesados. Cada sujeto y cada predicado son identificados por el *universal resource identifier* (*uri*) —recordemos que url es el tipo más difundido de *uri*—.

Volviendo a lo que se reseña en el artículo que nos ocupa y que estamos comentando, los verbos podrán ser identificados también por *uri*s, permitiendo definir otros nuevos simplemente definiendo un *uri* para ellos. De esta forma, los autores nos hacen notar que el lenguaje natural puede usar un mismo término para signi-

ficar algo distinto, según el contexto, y utilizar un *uri* diferente para cada concepto específico. Los tríos de *rdf* forman redes de información entre objetos conexos, y dado que éstos usan *uri*s para codificar datos en un documento, se asegura que los conceptos no son simplemente palabras sino que las vinculan a una definición única que todos pueden encontrar en el web.

Los *uri* aún no se encuentran muy difundidos y no lo estarán, creemos, hasta que una norma *ISO* los generalice, algo que, esperamos, ocurra en breve. Sin embargo, coincidimos con **Berners-Lee** en que esto no evita la superposición, pues dos recursos, por ejemplo dos bases de datos en línea, pueden usar diferentes identificadores para lo que de hecho es el mismo concepto.

Los sistemas informáticos deben disponer de un mecanismo que permita descubrir tales significados comunes para cualquier base de datos a la que se conecten. En este caso interesa encontrar una forma de representar semántica común en estructuras diferentes.

Berners-Lee, **Hendler** y **Lassila** confían en que la solución a este problema se puede encontrar en el tercer componente básico de la web semántica:

—Las ontologías. Este término tiene su origen en la filosofía, y la definición más general en el contexto de los sistemas informáticos es el de la especificación de una conceptualización.

Cada uno de los conceptos es expresado en una red terminológica que define sus atributos. Por ejemplo tipos de datos y sus comportamientos tales como relaciones con otros conceptos. Además, tienen una forma de establecer el alcance de estos atributos y comportamientos por medio de reglas, permitiendo que la ontología deduzca, o por lo menos proponga, a qué clase o categoría puede pertenecer cada nuevo concepto que se ingresa.

Sintéticamente una ontología tiene una taxonomía de conceptos —que define clases de objetos o conceptos y relaciones entre ellos— junto a un conjunto de reglas de inferencia. Según estos autores, puede facilitar el funcionamiento de la web para mejorar la exactitud de la recuperación, ya que el programa adquiere capacidad para realizar la búsqueda sólo en las páginas que se refieren al concepto preciso y, a la vez, funciona también cuando la respuesta no radica en una única página. Además, con las ontologías comienzan a surgir soluciones a los problemas terminológicos en la web. El significado de términos o de códigos xml usados en una página puede ser definido desde ésta a una ontología.

Dejamos ahora la reseña del documento sobre la web semántica y retomamos la reflexión inicial sobre los lenguajes naturales, artificiales y controlados. En este contexto las ontologías se sitúan dentro de los lenguajes controlados. Es fácil ver la vinculación entre ellas y los lenguajes documentales, pues muchos de sus fines son similares.

Es importante destacar que la inteligencia artificial, después de años de lucha con el procesamiento del lenguaje natural, parece haber optado como solución global el uso de los lenguajes controlados.

La incógnita es si los profesionales de la información seremos capaces de ampliar nuestras miras desde la tradicional representación de información concebida para recuperación en un sistema automatizado y asumiremos trabajos para enfocar la representación de conocimiento en la web.

Bibliografía

Berners-Lee, T.; Hendler, J.; Lassila, O. "The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities". En: *Scientific American*, 2001, mayo. Consultado en: 2001.
La versión en castellano de este artículo está disponible en: *Investigación y ciencia*, 2001, julio, n. 298, pp. 38-47.
http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html

Bryan, M. *Sgml: an author's guide*. New York: Addison-Wesley, 1998.

Conallen, J. *Building web applications with UML*. Boston: Addison-Wesley, 2000.

Cover, R. *Sgml/xml applications in cross-domain and multi-disciplinary enterprises*. *Oasis*. Consultado en: 2001.
http://www.oasis.org

Cover, R. Publicly available software for sgml/xml/dsssl. Consultado en: 2001.
http://www.oasis.org

Ead (*Encoded archival description*)**.** Consultado en: 2001.
http://www.loc.gov/ead/

International Standards Organization. IS0 8879 standard generalized markup language. Ginebra: 1986.

International Standards Organization. ISO/IEC 10744 hypermedia/time-based structuring language. Ginebra: 1997.

Isasi, P.; **Martínez, P.**; **Borrajo, D.** *Lenguajes, gramáticas y autómatas*. New York: Addison-Wesley, 1997.

Jacquesson, A.; **Rivier, A.** *Bibliothèques et documents numériques*. Paris: Electre, 1999.

Marc dtd, machine readable cataloging document type definition. Consultado en: 2001.
http://www.loc.gov/marc/marcdtd/marcdtdback.html

TEI guidelines for electronic text encoding and interchange: interchange, 2001. Consultado en: 2001.
http://etext.lib.virginia.edu/TEI.html

Winograd. T. *Language as a cognitive process*. New York: Addison-Wesley, 1983.