Comunidad de Madrid

Curso de Formación
"Lenguajes de Marcas y Sistemas de Gestión de la Información"
Del 12 de Abril al 7 de Junio de 2010

Unidad II:
XML: Estructura. DTD

Ponente:

José Luis Delgado Leal
Profesor del Departamento de Informática (Ciclos)
I.E.S. "Juan de la Cierva" (Madrid)

Contenidos de la Presentación

- Características de XML
- Documentos XML
- DTD
- Espacios de Nombres

Características de XML *(I)*

Comunidad de Madrid

- XML *(*eXtensible Markup Language*)*, surge de la revisión de SGML por W3C
- Utiliza prácticamente la misma notación, pero evita las irregularidades de SGML
- Los elementos tienen marca de terminación <x>...</x> y están bien anidados
- Los elementos sin contenido se pueden representar de manera abreviada <z/>
- Se introducen las llamadas instrucciones de procesamiento: <?....?>
- Se mantiene la referencia al tipo de documento mediante <!DOCTYPE ...>
- Se crea un lenguaje de definición de lenguajes particulares denominado XSD *(*XML Schema Definition*)*, más completo que DTD
- Se introducen los espacios de nombres *(*namespaces*)*, que facilitan combinar en un mismo documento marcas de varios lenguaje de marcado particulares

Características de XML *(II)*

Comunidad de Madrid

- XML ha servido para definir un gran número de lenguajes de marcado:
  - XHTML: revisión de HTML para adaptarlo a XML
  - SVG: descripción de gráficos vectoriales
  - DocBook: esquema general de documentos
  - MathML: descripción de fórmulas matemáticas, y otros ...

- XML se ha concebido desde el principio en un contexto de separación entre contenido y forma de presentación

- Dispone de un lenguaje para convertir el contenido abstracto de un documento XML en una forma concreta de presentación con el estilo adecuado

- El lenguaje de estilo asociado a XML es XSL *(*eXtensible Stylesheet Language*)* basado en la notación genérica XML. Es una combinación de varios estándares:
  - XPath: para referenciar nodos o conjunto de nodos en un documento
  - XSLT: XSL Transformations - convierte a otro XML, a HTML o a texto puro
  - FO: Formatting Objects - lenguaje de descripción de páginas

Características de XML (III)

Comunidad de Madrid

- XML admite las hojas de estilo en cascada CSS como alternativa a XSL
- XML es, globalmente, un conjunto de estándares
- Ya se han mencionado algunos de ellos: XPath, XSLT, etc. Otros estándares interesantes son
  - XLink, XPointer - referencia a fragmentos de documentos
  - XQuery - lenguaje de consulta
  - RDF: Resource Description Framework - descripción de recursos en la web, etc.
- Existe un gran número de herramientas para proceso de documentos XML
- También existen librerías e interfaces para el desarrollo de nuevas herramientas

Características de XML (IV)

Comunidad de Madrid

- Interfaces estándar:
  - DOM: Document Object Model - API orientada a estructura
  - SAX: Simple API for XML - Parsing orientado a eventos

- Librerías basadas en las interfaces anteriores o en otras similares, invocables desde diversos lenguajes (C, C++, Java, Perl, Ada, etc.):
  - libxml, libxsl
  - PerlXML, AdaXML
  - Expat: parser XML

- Diversos procesadores XML:
  - Apache: Xerces (parser), FOP (procesador XSL-FO), ...
  - MSXML (Microsoft) - parser XML, procesador XSLT. Incluido en Internet Explorer 5.0 y posteriores
  - Saxon, xsltproc: procesadores XSLT

Características de XML *(V)*

Comunidad de Madrid

- Herramientas: Editores XML
  - XMLnotepad (Microsoft) - editor del árbol de contenido
  - CookTop - editor de texto XML y acceso a MSXML
  - Morphon - editor WYSIWYG de XML, y editor CSS.
  - XXE: Xmlmind Xml Editor - editor WYSIWYG del contenido XML
  - XmlSpy - entorno completo de desarrollo XML

- Ejemplo de aplicaciones que utilizan XML de manera intensiva, se pueden citar:
  - AbiWord: Procesador de texto
  - OpenOffice: Paquete de ofimática
  - Netscape/Mozilla: Navegador web

Características de XML *(VI)*

Comunidad de Madrid

- La información descrita mediante marcado XML se organiza en objetos denominados documentos XML, almacenados como ficheros de texto

- Un documento XML puede contener los siguientes caracteres *(*en hexadecimal*):*
  - Caracteres de control*:* 09 - HT *(*Horizontal Tab*)*, 0A - LF *(*Line Feed*)*, 0D – CR *(*Carriage Return*)*. La versión 1.1 de XML todos los caracteres del 01 al 1F
  - Caracteres ASCII*:* 20-7F
  - Caracteres no ASCII*:* 80-D7FF, E000-FFFD, 10000-10FFFF

- Se recomienda emplear codificación, UTF-8, UTF-16, ISO-8859-1 o ASCII

- XML puede representar cualquier documento usando sólo caracteres ASCII

- Algunos caracteres están reservados y no pueden ser usados directamente en el contenido básico de información *(*texto*)* del documento. Estos son*:* < > & ' "

- Cuando son parte de los datos se escriben como referencias a entidades. Por ejemplo usando las siguientes formas simbólicas*:* < > & ' "

Documentos XML (I): Características Generales

Comunidad de Madrid

- La estructura general de un documento XML está formada por tres partes:
  - Prólogo (opcional): contiene una secuencia de instrucciones de procesamiento y/o declaración de tipo de documento
  - Cuerpo: árbol único de elementos marcados, con anidamiento estricto.
  - Epílogo (opcional): contiene una secuencia de instrucciones de procesamiento

- Puede haber comentarios en cualquier parte.
- Intuitivamente, el contenido de información del documento es el cuerpo. El prólogo y el epílogo sirven para facilitar la interpretación del documento.

- El documento completo es también una estructura en árbol. Para distinguir entre el cuerpo y el documento completo se usan los términos:
  - Document entity (o Document root) - se refiere a todo el documento
  - Document element - se refiere al cuerpo

Documentos XML (II): Nombre, Elementos y Comentarios

Comunidad de Madrid

- En XML se utilizan nombres que deben estar formados de la siguiente manera:
  - Inicial : letra _ : (letra, subrayado, dos puntos)
  - Resto: letra _ : - . (lo mismo más: guión, punto)
  - Se distinguen mayúsculas y minúsculas
  - Un nombre simple sólo contiene letras, subrayado y guiones. Los caracteres punto y dos puntos se usan en nombres cualificados
- Los elementos son fragmentos de información delimitados por marcas, de la siguiente manera:
  - Marca inicial: <x ....>
  - Contenido: texto u otros elementos.
  - Marca final: </x>
  - El contenido del elemento puede incluir, a su vez: referencia a caracteres, referencia a entidades y secciones CDATA
- Un documento XML puede contener anotaciones en forma de comentario:
  - Los comentarios no son parte del contenido de información del documento, y pueden ser ignorados por los procesadores XML
  - Los comentarios se escriben como <!-- ...texto del comentario... -->
  - El texto de un comentario no puede contener la secuencia --

Documentos XML (III): Instrucciones de Procesamiento, Marcas

Comunidad de Madrid

- Las instrucciones de procesamiento son directivas que pueden ser interpretadas por los procesadores XML
  - Dependiendo del procesador, se interpretarán determinadas instrucciones de procesamiento, pero otras no
  - El formato de una instrucción de procesamiento es: <?nombre ... texto de la instrucción ... ?>
  - El texto no tiene un formato definido. Es analizado por el procesador cuyo nombre se indica
- Las marcas sirven para delimitar los elementos que componen el documento XML
  - Un elemento queda delimitado por una marca inicial y otra final
  - Si el elemento no tiene contenido, se puede escribir en forma abreviada como una sola marca
  - El formato de las marcas es:
    - Marca inicial: <nombre atributos_opcionales>
    - Marca final: </nombre>
    - Elemento vacío: <nombre atributos_opcionales /> (equivale a <nombre atributos_opcionales></nombre>)

Documentos XML *(IV)*: Literales y Atributos

Comunidad de Madrid

- Los literales sirven para delimitar fragmentos de texto, de acuerdo con las siguientes reglas:
  - Delimitados por comillas simples o dobles: 'ejemplo' "ejemplo"
  - Se puede usar la otra dentro del literal: "Roger O'Connors dijo 'Sí' al votar"
  - Si hay que usar el delimitador dentro del literal se usa la referencia a entidad ' *(')* o " *(")*

- Los atributos son fragmentos de información que forman parte de la marca inicial de un elemento.
  - La sintaxis es: <nombre_marca nombre_atributo = 'valor' nombre_atributo = "valor" ...>
  - No puede haber dos atributos con el mismo nombre en la misma marca
  - Los valores de los atributos se dan como literales, entre comillas o apóstrofos.

Documentos XML *(V)*: Contenido

Comunidad de Madrid

- Toda la información básica contenida en el documento se representa como texto
- No hay datos numéricos, binarios, lógicos, etc.
- Estos datos de texto se escriben según las siguientes reglas:
  - Pueden contener todos los caracteres Unicode válidos en XML
  - Los caracteres < y & no se pueden usar directamente: se introducen como < *(<)* y & *(&)*
  - Se recomienda usar también > en forma simbólica >
  - Se considera espacio en blanco el formado por: espacio, HT, salto de línea
  - El espacio en blanco es parte del valor del dato
  - El salto de línea puede ser: LF, CR, CR-LF. Se convierte internamente a LF

Documentos XML *(V)*: Referencias a caracteres y entidades

Comunidad de Madrid

- Los caracteres se pueden escribir directamente si forman parte del conjunto de caracteres correspondiente al sistema de codificación del texto del documento

- Además se pueden escribir como referencias, con el siguiente formato:
  - &#NNNNN; - decimal *(*hasta 5 dígitos*)*
  - &#xXXXX; - hexadecimal *(*hasta 4 dígitos*)*
  - El código numérico *(*decimal o hexadecimal*)* corresponde al código Unicode

- Una entidad es un fragmento de información, definido como un valor constante, al que se puede hacer referencia mediante un nombre

- La forma de hacer referencia a una entidad se hace así: &nombre;

- Sólo hay 5 entidades predefinidas: < *(<)* > *(>)* & *(&)* ' *(')* " *(")*

- Otras entidades deben ser definidas para poder usarlas

Documentos XML *(VI)*: CDATA

Comunidad de Madrid

- Una sección CDATA es un texto literal que puede contener directamente incluso caracteres de marcado reservados, sin necesidad de escribirlos como referencias

- La sintaxis es*:*

  *<![*CDATA*[* ... texto con caracteres especiales < > ' & " ... *]]>*

- La combinación *]]>* no puede aparecer dentro de una CDATA

- Tampoco puede aparecer directamente como texto fuera de ella. En este caso debería ser representada como *]]*>

Documentos XML *(VII)*: Prólogo: Declaración XML

Comunidad de Madrid

- La declaración XML es una instrucción de procesamiento especial
- Es opcional
- Cuando existe debe ser la primera instrucción del prólogo
- Su formato es:

  <?xml version="1.0" encoding='utf-8' standalone="yes"?>

  - version: atributo obligatorio, sólo puede valer '1.0' y '1.1' *(por ahora)*
  - encoding: atributo opcional, recomendado, debe ser un valor IANA válido, por defecto 'utf_8' o 'utf-16'
  - standalone: atributo opcional, puede valer 'yes' o 'no'. Indica si el documento puede ser procesado sin necesidad de acceder a definiciones externas
- Los nombre 'xml', 'version', ... deben escribirse en minúsculas
- Los valores pueden escribirse en minúsculas o mayúsculas *('UTF-8' = 'utf-8')*

Documentos XML (VIII): Prólogo: Declaración de tipo de documento

Comunidad de Madrid

- La declaración de tipo de documento es opcional

- Se escribe en el prólogo, y tiene un formato especial, distinto de las marcas y de las instrucciones de procesamiento

- Esta declaración puede contener una indicación explícita del lenguaje particular de marcado correspondiente al documento (definido externamente)

- También puede contener la declaración directa de ciertos elementos del lenguaje de marcado (definidos internamente)

- El formato es uno de los siguientes:

  <!DOCTYPE nombre-elemento PUBLIC public-ID system-ID ... >

  <!DOCTYPE nombre-elemento SYSTEM system-ID ... >

  - nombre-elemento es el nombre del elemento principal (elemento raíz del cuerpo)
  - public-ID es un identificador asociado al lenguaje de marcado particular
  - system-ID es una referencia a un DTD o XSD externo

Documentos XML *(IX)*: Cuerpo y Epílogo

Comunidad de Madrid

- El cuerpo es obligatorio
- Está constituido por un único árbol de elementos, es decir, con una raíz única
- Además de los elementos, puede contener comentarios e incluso instrucciones de procesamiento
- El epílogo es opcional
- En general se omite, ya que no está claro para qué sirve
- Está pensado para contener instrucciones de procesamiento
- El problema es que resulta poco intuitivo poner estas instrucciones al final

Documentos XML *(X)*: Espacio en blanco

Comunidad de Madrid

- Un documento XML puede contener espacio en blanco *(*caracteres de espacio, tabulación y saltos de línea*)*, que puede ser significativo o no

- Los no significativos pueden ser ignorado, modificado o eliminado

- El espacio en blanco que sea parte del contenido de texto de un elemento es significativo *(*a menos que se diga expresamente lo contrario*)*

- El espacio en blanco entre marcas en lugares donde no se permite contenido de texto no es significativo.

- El espacio en blanco entre partes de una marca de comienzo no es significativo

- El espacio en blanco en el valor de un atributo puede ser reajustado dependiendo del tipo de atributo

- Ejemplo: para atributos CDATA se elimina el espacio en blanco al comienzo y al final, y se reemplazan varios caracteres en blanco seguidos por uno solo.

Documentos XML *(X)*: Documentos XML bien formados vs. válidos

Comunidad de Madrid

- Se dice que un documento XML está bien formado cuando cumple las reglas sintácticas indicadas

- Los procesadores XML pueden rechazar cualquier documento que no esté bien formado

- Un documento XML válido es un documento que está bien formado, y además cumple con la definición de un lenguaje de marcado particular

- Es decir, el cuerpo del documento tiene una estructura de elementos compatible con el lenguaje concreto al que corresponde

- Así, todo documento XML válido es un documento XML bien formado, pero no ocurre al contrario

DTD (I): Características y Estructura (I)

Comunidad de Madrid

- XML mantiene el metalenguaje DTD de definición de lenguajes particulares de marcado definido en SGML por motivos de compatibilidad

- Las siglas DTD significan Document Type Definition, y se refieren, por tanto, a la definición de un tipo o esquema de documento

- La definición del tipo de documento puede hacerse:
  - En un fichero separado, y poner la referencia en el DOCTYPE
  - En el propio documento, dentro del DOCTYPE
  - Con una combinación de ambos métodos

- El lenguaje DTD permite definir elementos, atributos, entidades y notaciones

- El formato general de una definición elemental en una DTD es:
  <!clase parámetros ...>

donde clase será ELEMENT, ATTLIST, ENTITY o NOTATION, y los parámetros dependerán de la clase de definición.

DTD (II): Características y Estructura (II)

Comunidad de Madrid

- Elementos:
  - Los elementos configuran la estructura general de un documento XML
  - Se anidan unos dentro de otros formando un árbol
  - Para cada elemento se define su nombre y la estructura de su contenido (texto u otros elementos)
- Atributos:
  - Los atributos son fragmentos de información asociados a un elemento
  - Tienen nombre y su contenido es siempre texto. No pueden anidarse.
- Entidades:
  - Las entidades son similares a las macros de ciertos lenguajes de programación
  - Son fragmentos de texto constantes a los que se puede hacer referencia mediante un nombre
  - Sirven para simplificar la escritura de documentos y DTD's en los que aparecen repetidamente ciertos fragmentos de texto
- Notaciones:
  - Las notaciones permiten delimitar contenido no XML dentro de un documento XML.

DTD (III): Definición de Elementos (I)

Comunidad de Madrid

- Los parámetros de una definición de elemento son su nombre y su esquema

- El formato de la definición puede ser uno de los siguientes:

  <!ELEMENT nombre ANY >

  Define un elemento cuyo contenido puede ser cualquiera

  <!ELEMENT nombre EMPTY >

  Define un elemento sin contenido

  <!ELEMENT nombre (expresión regular)>

  <!ELEMENT nombre (expresión regular)repetición >

  - Definen elementos compuestos que contienen otros elementos y cuya estructura debe ajustarse a la expresión regular que se indica
  - La expresión regular debe estar formada por nombres de elementos y los metacaracteres: de agrupación "(" ")", de secuencia ",", de alternativa"|" y de repetición "+" [1..n], "?" [0..1], "*" [0..n]
  - Siempre es necesario un nivel externo de paréntesis
  - Estas formas de expresión no pueden contener el símbolo #PCDATA

DTD (IV): Definición de Elementos (II)

Comunidad de Madrid

- El formato de la definición puede ser uno de los siguientes (cont.):

  <!ELEMENT nombre (#PCDATA) >
  <!ELEMENT nombre (#PCDATA | nombre | nombre ...)* >

  - Definen elementos con lo que se denomina mixed content
  - Están formados por texto solo o entremezclado con otros elementos
  - El nombre especial #PCDATA indica contenido de texto
  - Debe aparecer siempre al principio de la expresión
  - Puede contener sólo ese término o será una repetición de una alternativa simple.

DTD *(V)*:
Definición de Atributos *(I)*

Comunidad de Madrid

- Los parámetros de una definición de atributos son el nombre del elemento al que corresponden y los nombres y descripciones de contenido de los atributos

- El conjunto de atributos de un elemento puede declararse en una sola definición, o por partes, en varias definiciones separadas

- El formato de una definición de atributos es el siguiente:

```
<!ATTLIST elemento
        nombre tipo tratamiento_por_defecto
        nombre tipo tratamiento_por_defecto
        ...
>
```

  - elemento: es el nombre del elemento al que corresponden los atributos
  - nombre: es el nombre del atributo
  - tipo: CDATA, (valor | valor | ...), ID, IDREF, IDREFS, NMTOKEN, NMTOKENS
  - tratamiento_por_defecto: #REQUIRED, #IMPLIED, #FIXED valor_por_defecto, valor_por_defecto

DTD (VI): Definición de Atributos (II)

Comunidad de Madrid

- Un valor del tipo CDATA corresponde a un valor de texto, en general
- Un valor del tipo enumerado (valor | valor | ...) especifica uno entre varios posibles nmtoken
- Un valor del tipo ID es un nombre que debe ser único en todo el documento, y que sirve para identificar el elemento
- Un valor del tipo IDREF es un nombre que debe aparecer como valor de un atributo ID en algún elemento del documento
- Un valor del tipo IDREFS es una lista de IDREF separados por espacio en blanco

DTD (VII): Definición de Atributos (III)

Comunidad de Madrid

- Un valor del tipo NMTOKEN es similar a un nombre, sin la restricción del carácter inicial

- Un valor del tipo NMTOKENS es una lista de NMTOKEN separados por espacio en blanco

- Un atributo #REQUIRED debe aparecer siempre

- Un atributo #IMPLIED es opcional, sin un valor por defecto

- Un atributo #FIXED debe tener el valor indicado si no se omite, y si se omite se asumirá el valor indicado

- Si no se indica #REQUIRED ni #IMPLIED ni #FIXED sino sólo un valor por defecto, el atributo es opcional y si se omite se asume el valor por defecto

DTD (VIII): Definición de Entidades

Comunidad de Madrid

- Las entidades son valores constantes a los que se puede hacer referencia mediante un nombre

- Hay dos clases de entidades: General entities (para ser usadas en el contenido del documento) y Parameter entities (para ser usadas en la DTD)

- Una general entity se define como:
  - <!ENTITY nombre valor_de_sustitución >
  - El valor_de_sustitución puede ser un texto literal, que a su vez puede contener referencias a otras entidades
  - Se hace referencia a ella con &nombre; en el contenido del documento

- Una parameter entity se define como:
  - <!ENTITY % nombre valor_de_sustitución >
  - El valor_de_sustitución puede ser un texto literal, que a su vez puede contener referencias a otras entidades
  - A la entidad se hace referencia con %nombre; en la DTD

DTD (IX): DTD's Modulares

Comunidad de Madrid

- Es posible construir una DTD por partes, en documentos separados
- Una DTD maestra puede contener parte de las declaraciones e importar otras DTD
- Para ello se utiliza el mecanismo de parameter entities
- Se hace referencia a entidades externas mediante una notación similar a un <!DOCTYPE ...> para el valor de sustitución

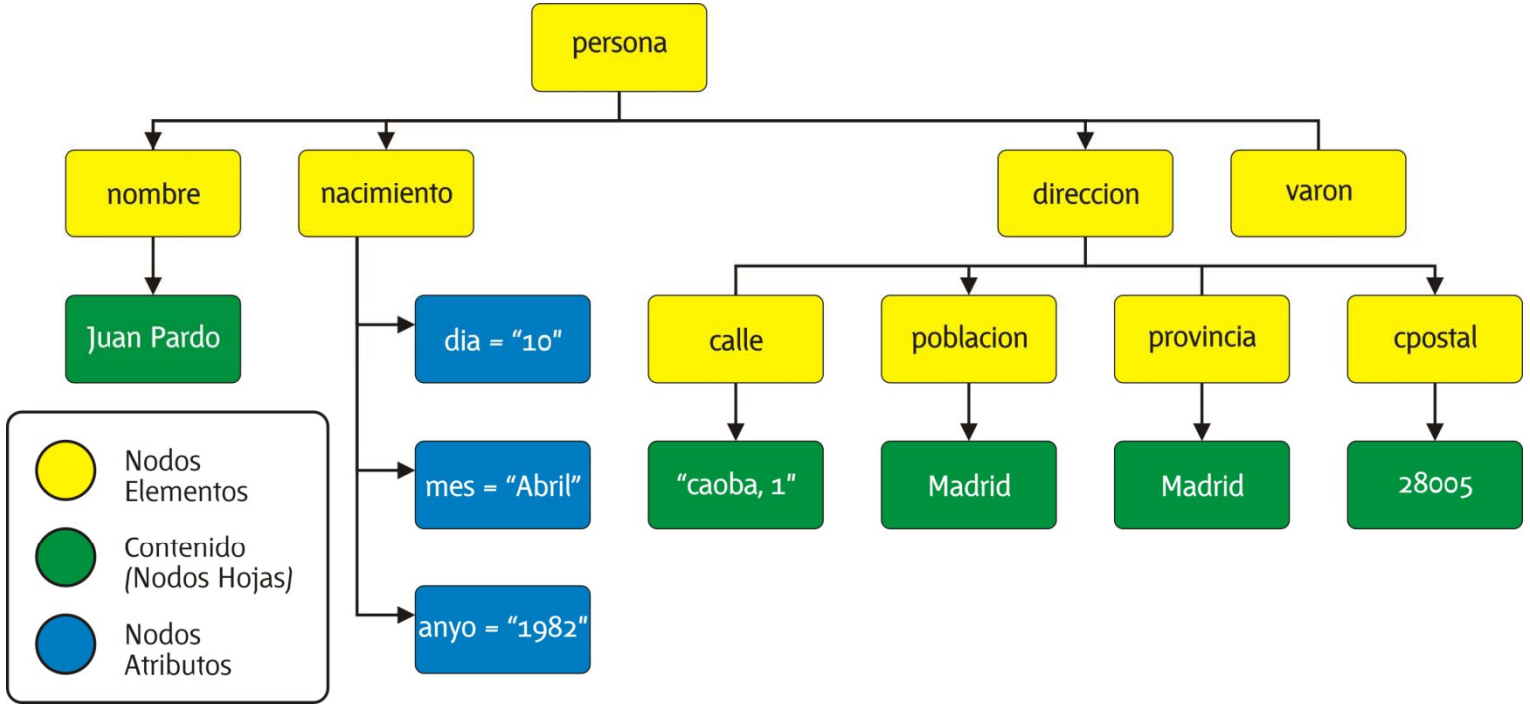
```
<!ENTITY % nombre PUBLIC public-id system-id >
<!ENTITY % nombre SYSTEM system-id >
...
%nombre;
```

- Las declaraciones externas se insertan en la DTD maestra en el punto en el que se hace referencia a la entidad externa mediante %nombre;

DTD (IX):
XML y DTD: un problema a resolver (I)

Comunidad de Madrid

- A continuación se muestra una jerarquía de datos:



1) Proporcionar una DTD que refleje esta jerarquía suponiendo que "nacimiento" es un elemento opcional y hay al menos una dirección DTD
2) Genera un fichero XML que referencie al DTD (de manera interna y externa) y contenga dos elementos persona

DTD (IX):
XML y DTD: un problema a resolver (II)

Comunidad de Madrid

- Sea el siguiente código que representa una matrícula de un alumno:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<matricula>
        <personal>
                <dni>99223366M</dni>
                <nombre>Juan Pardo Martín</nombre>
                <titulacion>Ingeniería Informática</titulacion>
                <curso_academico>1997/1998</curso_academico>
                <domicilios>
                        <domicilio tipo="familiar">
                                <nombre>C/ Principal nº1</nombre>
                        </domicilio>
                        <domicilio tipo="habitual">
                                <nombre>C/ Secundaria nº2</nombre>
                        </domicilio>
                </domicilios>
        </personal>
        <pago>
                <tipo_matricula>Matrícula Ordinaria</tipo_matricula>
        </pago>
</matricula>
```

Proporcionar una DTD que permita validar el código de arriba

Resultado

DTD (IX):
XML y DTD: un problema a resolver (III)

Comunidad de Madrid

- Sea el siguiente supuesto práctico a resolver:

Una cadena de videoclubs quiere emplear una base de datos para almacenar información referente a las facturas que se hacen a los clientes. Esta información es la siguiente:

- Cada factura está formada por dos tipos de información: datos de cliente y datos del ticket de factura propiamente dichos.
- De los datos del cliente, se desea guardar: su nombre, su primer y segundo apellidos, DNI y teléfono (uno). Además, como características del cliente, se desea conocer el identificador de cliente.
- De los datos de la factura en si, se quiere guardar un resguardo de factura y los alquileres o compras que se incluyen en la factura. En cada factura habrá alquileres, compras o los dos. El resguardo siempre se incluye.
- El resguardo incluye la forma de pago y el importe total.
- Los alquileres se realizan de películas. El alquiler de películas lleva asociada una fecha de devolución que es común a todas las películas.
- De cada película se quiere conocer su título, género, duración y los nombres y apellidos de tres actores que participan en ella. Existen dos atributos que definen a las películas: idPelicula y valoracion
- Con respecto a las compras, hay que diferenciar entre compras de DVDs y de cintas de video
- De los DVDs interesa el título del DVD, la fecha de salida al mercado y si viene o no con extras
- De las cintas de video se guardará el formato (VHS, por ejemplo) y si está rebobinada o no

Proporcionar una DTD que permita validar el código de arriba Resultado

Espacios de Nombres *(I)*

Comunidad de Madrid

- Es relativamente frecuente combinar varios vocabularios de marcado en un mismo documento

- Esto puede dar lugar a conflictos de nombres, ya que cada vocabulario o lenguaje particular de marcado puede haber sido preparado de manera independiente y usar nombres de marcas que también forman parte de otro vocabulario

- Para permitir la mezcla de vocabularios se ha ideado el mecanismo de espacios de nombres *(*namespaces*)*

- Cada vocabulario se asocia a un espacio de nombres separado

- Los espacios de nombres se identifican mediante un designador, que debe tener el mismo formato que un URI, aunque no tiene que existir necesariamente en la red

Espacios de Nombres (II)

Comunidad de Madrid

- El hecho de usar un URI como identificador del espacio de nombres es un artificio para garantizar que no se reutilizará accidentalmente el mismo identificador para dos vocabularios diferentes

- Se supone que quien asigna un identificador concreto a un espacio de nombres tiene también la autoridad para crear realmente un recurso en Internet con el URI elegido

- El uso de un espacio de nombres no implica que la definición del vocabulario exista realmente

- Tampoco sirve para localizar la definición en el caso de que exista

- Para ello se debe emplear la declaración DOCTYPE como se ha indicado anteriormente

Espacios de Nombres (III)

Comunidad de Madrid

- El siguiente ejemplo (tomado del estándar MathML) mezcla los vocabularios XHTML y MathML en una misma página web:

```
<html xmlns="http://www.w3.org/1999/xhtml">
     <head>
          <title>MathML's Hello Square</title>
     </head>
     <body>
          <p> This is a perfect square:</p>
          <math xmlns="http://www.w3.org/1998/Math/MathML">
               <mrow>
                    <msup>
                    <mfenced>
                         <mrow>
                              <mi>a</mi>
                              <mo>+</mo>
                              <mi>b</mi>
                         </mrow>
                    </mfenced>
                    <mn>2</mn>
                    </msup>
               </mrow>
          </math>
     </body>
</html>
```

Espacios de Nombres (IV)

Comunidad de Madrid

- Los espacios de nombres se declaran en el elemento donde se usan, usando el atributo genérico "xmlns"

- No es necesario declararlo en todos y cada uno de los elementos (propagación automática a todos los elementos hijos)

- Los elementos que aparezcan fuera de todas las declaraciones de espacios de nombres no se consideran asociados a ningún espacio de nombres particular

- Cuando los elementos de distintos vocabularios se entremezclan de forma intensiva puede resultar engorroso tener que declarar el espacio de nombres

- Para simplificar la escritura se pueden usar prefijos cortos que identifiquen el espacio de nombres de cada elemento

- En este caso el prefijo hay que ponerlo en cada elemento individual, ya que no hay propagación a los hijos

Espacios de Nombres (V)

Comunidad de Madrid

```
<html xmlns="http://www.w3.org/1999/xhtml"
     xmlns:m="http://www.w3.org/1998/Math/MathML">
     <head>
          <title>MathML's Hello Square</title>
     </head>
     <body>
          <p> This is a perfect square:</p>
          <m:math>
               <m:mrow>
                    <m:msup>
                         <m:mfenced>
                              <m:mrow>
                                   <m:mi>a</m:mi>
                                   <m:mo>+</m:mo>
                                   <m:mi>b</m:mi>
                              </m:mrow>
                         </m:mfenced>
                         <m:mn>2</m:mn>
                    </m:msup>
               </m:mrow>
          </m:math>
     </body>
</html>
```

Espacios de Nombres (VI)

Comunidad de Madrid

- Es este ejemplo se ha mezclado el uso de un espacio de nombres por defecto (XHTML) que no necesita prefijo, y un espacio de nombres (MathML) con prefijo explícito

- El prefijo puede elegirse arbitrariamente en cada documento, aunque existe la costumbre de usar siempre el mismo prefijo nemotécnico para cada vocabulario estándar de uso amplio.

- En un mismo documento se pueden declarar cuantos espacios de nombres sean necesarios, usando un prefijo diferente para cada uno, además del espacio de nombres por defecto que no necesita prefijo

- Finalmente conviene saber que las declaraciones de espacios de nombres afectan directamente a los nombres de elementos, pero no a los nombres de atributos

- Los nombres de los atributos se interpretan según el elemento en el que aparecen, sin necesidad de usar prefijos