

XML: eXtensible Markup Language

Tema 3
(no entra ni Schemas XML
Path)

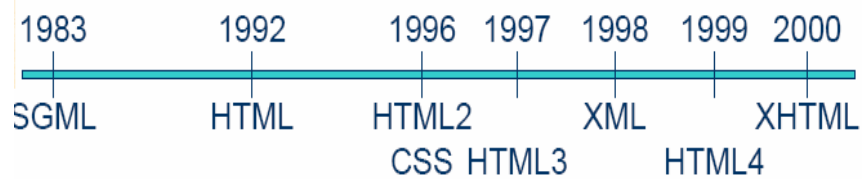
1

Introducción Histórica (I)

- **XML se constituyó como estándar de la W3C en el año 1998. En 2000 se aprueba su versión 1.0**
- Se trata de un lenguaje de **marcas**, igual que HTML o su precursor SGML
- Se diferencia de SGML por su **sencillez**
- Se diferencia de HTML por su **flexibilidad**: el número de etiquetas que puede incluir un documento XML es ilimitado
- Al igual que HTML, es **portable** a cualquier plataforma

2

Introducción Histórica (II)



3

Introducción Histórica (III)

- **Objetivos principales:**
 - **Directamente utilizable en Internet**
 - **Soporte para una amplia variedad de aplicaciones para transferencia de datos**
 - **Compatible con SGML**
 - Posibilidad de crear sencillos procesadores de XML
 - Documentos XML legibles y medianamente claros (depende de la definición)
 - Diseño rápido del lenguaje
 - Simple, pero perfectamente formalizado
 - Documentos XML fáciles de crear

4

XML vs. HTML

- HTML carece de un chequeo sintáctico. **Páginas con errores son mostradas en los navegadores**
- HTML **carece de estructura**
- HTML no es orientado a objeto
- HTML mezcla contenido y representación
- Por todo esto:
 - HTML **no puede ser fácilmente leído por una máquina**
 - HTML **nunca será un estándar de intercambio de datos**
- XML cubre todo esto con un lenguaje de sencillez extrema

5

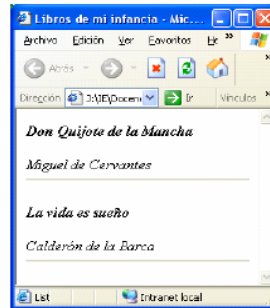
Características de XML (I)

- Es un subconjunto del lenguaje SGML
- Al igual que él, se utiliza para representar datos de forma **estructurada** (Jerárquica)
- Se basa en una **gramática de obligado cumplimiento**. Esto facilita el desarrollo de *parsers* y, por lo tanto, su utilización masiva
- La estructura interna de un documento XML puede reflejarse en otro documento denominado **DTD (Document Type Definition)**
- A diferencia de HTML, separa radicalmente la **semántica** del documento, de su representación gráfica

6

Documento HTML (I)

```
<HTML>
<HEAD><TITLE>Libros de mi
  infancia</TITLE></HEAD>
<BODY>
<P><I><B>Don Quijote de la
  Mancha</B></I>
<P><I>Miguel de Cervantes</I>
<HR>
<P><B>La vida es sueño</B>
<P><I>Calderón de la Barca</I>
<HR>
</BODY>
</HTML>
```



7

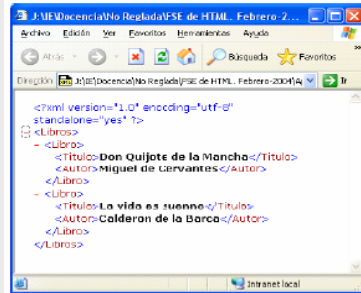
Documento HTML (II)

- En apariencia, el documento HTML anterior es correcto, sin embargo:
 - Existen etiquetas que nunca se cierran: <P>
 - Algunas etiquetas no están bien anidadas: el primer <I> nunca se cierra
 - Para un lector no humano, no se sabe qué es un libro y qué es un autor
- XML erradica todos estos problemas!!

8

Documento XML

```
<?xml version="1.0" encoding="utf-8"
standalone="yes" ?>
<Libros>
  <Libro>
    <Titulo>Don Quijote de la
Mancha</Titulo>
    <Autor>Miguel de
Cervantes</Autor>
  </Libro>
  <Libro>
    <Titulo>La vida es sueño</Titulo>
    <Autor>Calderon de la
Barca</Autor>
  </Libro>
</Libros>
```



9

Reglas Generales de XML

- Un **único** elemento raíz
- Todo elemento debe tener etiquetas de **apertura y cierre**
- Distinción entre mayúsculas/minúsculas
- **Anidamiento** perfecto entre elementos
- Los **valores** de **atributos** siempre van entre **comillas**
- Los espacios en blanco se conservan
- Los caracteres CR/LF se transforman en LF

10

Documentos Bien Formados y Válidos

- Se dice que un documento es bien formado cuando:
 - Cumple con todas las reglas anteriormente expuestas
 - Contiene uno o más elementos
 - Hay un único elemento raíz (elemento documento)
 - Si el documento consta de más de una parte, todas están bien formadas
 - No se encuentran caracteres prohibidos en el texto
- Un documento es válido cuando, además de ser 'bien formado', cumple con las especificaciones semánticas expuestas en su plantilla (DTD o XML Schema)

11

Elementos (I)

- Comentarios:
 - <!-- Esto es un comentario, y no puedo incluir un doble guión-->
- Instrucciones de procesamiento:
 - <? Instrucción ?>
 - La instrucción no puede incluir los caracteres ?>
- Secciones CDATA:
 - <![CDATA[Este texto no será tratado, puede incluir "cualquier" &carácter < >]]>
 - No son tratadas por el *parser*
 - Pueden incluir cualquier carácter prohibido (" , ' & , > , <).
 - No puede incluir la cadena]]>

12

Elementos (II)

- Prólogo:
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
 - Es una instrucción de procesamiento **obligatoria**
 - **Version**: indica la versión de XML que se está utilizando (1.0 en la actualidad). *Es **obligatoria***
 - **Encoding**: indica cómo se codificó el documento, y ***no es obligatoria*** (por defecto UTF-8). Válido para otros juegos de caracteres
 - **Standalone**: “yes” indica que el documento no va acompañado de DTDs externos; “no” indica que posee DTD interno. ***No es un atributo obligatorio***

13

Elementos (III)

- DOCTYPE: **<!DOCTYPE MiDTD SYSTEM “C:\MiDTD.dtd”>**
 - Indica la referencia (URI) al DTD, así como el nombre (MiDTD) del elemento raíz de la misma
 - La DTD podría ir incorporada en el propio documento XML, sin requerir otro fichero aparte
 - El documento XML deberá cumplir con el contenido del DTD

14

Elementos (IV)

- **Etiquetas:**
 - Deben ir correctamente anidadas: apertura y cierre
 - Etiqueta de apertura: comienza por <, más el nombre de la etiqueta y terminan por >. Ejemplo <Libro>
 - Etiqueta de cierre: </Libro>
 - Etiqueta vacía: <Libro />
 - No puede iniciar el nombre con “.”, “:”, “-”, números
 - Luego de la primera letra pueden colocarse “.”, números, “-”
 - El nombre debe comenzar por una letra o un “_”
 - No puede comenzar por “xml”

15

Elementos (V)

- **Elemento:**
 - Es el conjunto de la etiqueta (marcador) de apertura, su contenido y la de cierre
 - Por ejemplo: <Libro>Don Quijote de la Mancha</Libro>
 - Hay algunos caracteres reservados (prohibidos):
 - Signo de mayor: >
 - Signo de menor: <
 - Ampersand: &
 - Apóstrofe: ‘
 - Comilla: “
 - Estos caracteres prohibidos se reemplazan por entidades o se incluyen en secciones CDATA

16

Elementos (VI)

- **Atributos:**

- Cada elemento puede contener 0 ó más atributos
- Su valor debe ir siempre entrecomillado
- Sólo pueden aparecer en etiquetas de apertura o vacías
- El mismo atributo no puede aparecer repetido en la misma etiqueta
- Si el documento incluye DTD, cada atributo debe estar definido como atributo del presente elemento
- No puede contener ninguna referencia a entidad externa
- Son siempre tratados como cadenas de texto

17

Elementos (VII)

```
<Libro>  
<Titulo>Don Quijote de la Mancha</Titulo>  
<Autor>Miguel de Cervantes</Autor> (Sin atributos)  
<Precio> 1.123 euros </Precio>  
<Editorial> Santillana </Editorial>  
</Libro>
```

```
<Libro Precio = "1.123 euros" Editorial = "Santillana">  
<Titulo>Don Quijote de la Mancha</Titulo>  
<Autor>Miguel de Cervantes</Autor>  
</Libro> (Dos elementos son atributos)
```

18

DTDs (I) (Declaración de tipos)

```
<!DOCTYPE Libros SYSTEM
"Libros1.dtd">
<Libros>
  <Libro>
    <Titulo>Don Quijote de la
    Mancha</Titulo>
    <Autor>Miguel de
    Cervantes</Autor>
  </Libro>
  <Libro>
    <Titulo>La vida es
    sueño</Titulo>
    <Autor>Calderon de la
    Barca</Autor>
  </Libro>
</Libros>

<!DOCTYPE Libros [
  <ELEMENT Libros (Libro)+>
  <ELEMENT Libro (Titulo, Autor)>
  <ELEMENT Titulo (#PCDATA)>
  <ELEMENT Autor (#PCDATA)>
]>
<Libros>
  <Libro>
    <Titulo>Don Quijote de la
    Mancha</Titulo>
    <Autor>Miguel de Cervantes</Autor>
  </Libro>
  <Libro>
    <Titulo>La vida es sueño</Titulo>
    <Autor>Calderon de la Barca</Autor>
  </Libro>
</Libros>
```

19

DTDs (II)

- Toda DTD debe tener **uno y sólo un elemento raíz** (también conocido como **elemento documento**)
- Este **documento raíz** debe **coincidir** con el nombre que aparece a continuación del DOCTYPE
- Un documento DTD puede contener:
 - Declaraciones de elementos
 - Declaraciones de atributos para un elemento
 - Declaraciones de entidades
 - Declaraciones de notaciones
 - Instrucciones de procesamiento
 - Comentarios
 - Referencias a entidades de parámetro

20

DTDs (III) (Elemento Raíz)

- A partir del elemento raíz, pueden opcionalmente colgar (de forma jerárquica) otros elementos

<!ELEMENT Libros (Libro)+>

<!ELEMENT Libro (Titulo, Autor)>

<!ELEMENT Titulo (#PCDATA)>

<!ELEMENT Autor (#PCDATA)>

21

DTDs (IV) (Contenido de los Elementos)

- Contenido de un elemento:
 - **EMPTY**: el elemento está vacío (puede contener atributos).
<!ELEMENT IMAGEN EMPTY>
 - **ANY**: el elemento puede contener a cualquier otro elemento o incluso contenido textual.
<!ELEMENT IMAGEN ANY>
 - **Otros elementos**: un elemento puede contener uno o más elementos hijos en una cierta secuencia (Ej. Libro)
 - **#PCDATA**: texto *parseado*.
<!ELEMENT LIBRO (#PCDATA)>
 - **Mixto**: el elemento puede incluir secuencias de caracteres opcionalmente mezcladas con elementos hijos.
<!ELEMENT LIBRO (#PCDATA | AUTOR)*>

22

DTDs (V)

- Secuencias de hijos de un elemento:

- Secuencia:

- Secuencia en orden: hijos separados por comas

- Opciones: hijos separados por | (barra)

- Conjuntos de elementos pueden agruparse entre paréntesis

- Cardinalidad: un elemento, o un conjunto de ellos puede repetirse 0, 1 ó más veces:

<i>elemento</i>	Elemento repetido 1 única vez
?	Elemento repetido 0 ó 1 vez
*	Elemento repetido 0 ó más veces
+	Elemento repetido 1 ó más veces

23

DTDs (VI)

```
<!ELEMENT chiste (basilio+, antonio, aplauso?)>
<!ELEMENT basilio (#PCDATA | quote)*>
<!ELEMENT antonio (#PCDATA | quote)*>
<!ELEMENT quote (#PCDATA)*>
<!ELEMENT aplauso EMPTY>
<!ATTLIST chiste
  name ID #REQUIRED
  label CDATA #IMPLIED
  status (funny|notfunny) 'funny'>
```

1 o más veces
secuencia
0 o 1 veces
0 o más veces
alternativa
Valor por defecto

24

DTDs (VII) (Ejemplo)

```
<!ELEMENT LIBRO (Autor, Editorial)>
<!ELEMENT Autor (#PCDATA)>
<!ELEMENT PELICULA (Actor|Actriz|Director)+>
<!ELEMENT PELICULA ((Actor | Actriz)*, Director,
  Maquillaje?)>
<!ELEMENT PELICULA (#PCDATA | Actor)*>
<!ELEMENT PELICULA (Titulo, Genero, (Actor | Actriz |
  Narrador)*)>
<!ELEMENT FICHA (Nombre+, Apellido+, Direccion*, foto?,
  TelFijo*|TelMovil*)
```

25

DTDs (VIII)

Ejercicio: Hacer una DTD.

```
<?xml version="1.0" encoding="utf-8" ?>
<Agenda>
<Persona>
  <Nombre> Anabel </Nombre>
  <Apellido> Fraga </Apellido>
  <Email> afraga@ie.inf.uc3m.es </Email>
  <Oficina> 2.1 B18 </Oficina>
  <Telefono> 5555555 </Telefono>
  <Movil> 5557777 </Movil>
</Persona>
</Agenda>
```

26

DTDs (IX) (Atributos)

- Un elemento puede opcionalmente declarar uno o más atributos
<!ATTLIST Elemento Atributo Tipo Modificador>
- Los atributos de un elemento pueden incluirse en una o más declaraciones <!ATTLIST ...>. Si se hace en la misma declaración, basta con separar con un espacio (espacio, tabulador, retorno de carro)

27

DTDs (X) (Tipos de Atributos)

- Tipo de un atributo:
 - Tipo cadena: CDATA
<!ATTLIST Autor Nacionalidad CDATA>
 - Tipo enumerado:
<!ATTLIST Pelicula Genero (Ficcion | Terror | Humor)>
 - Tipo simbólico:
 - **ID**: valdrá como identificador en el resto del documento, sólo un atributo ID por cada elemento
 - **IDREF, IDREFS**: su valor debe coincidir con algún otro atributo de tipo ID en el resto del documento XML. IDREFS separa las referencias por espacio. "ID1 ID2 ID3"
 - **ENTITY, ENTITIES**: su valor debe coincidir con una o más entidades no analizadas
 - **NMTOKEN, NMTOKENS**: su valor ha de ser una cadena de tipo *token*

28

DTDs (XI) (Modificadores de Atributos)

- Modificadores:
 - **#REQUIRED**: este atributo debe ser obligatoriamente introducido.
`<!ATTLIST Pelicula Titulo CDATA #REQUIRED>`
 - **#IMPLIED**: indica que el atributo es *opcional*
 - **ValorPredeterminado**: si se omitiese el atributo, los procesadores recogerían este valor por defecto
`<!ATTLIST Pelicula Genero (Ficcion | Terror | Humor) "Humor">`
`<!ATTLIST Autor Nacionalidad CDATA "Española">`
 - **#FIXED**: se incluya o no se incluya el atributo, los procesadores siempre obtendrán este mismo valor
`<!ATTLIST Autor Nacionalidad CDATA #FIXED "Española">`

29

DTDs (XII) (Recomendaciones para modelado de Atributos)

- Frecuentemente un mismo objeto se puede diseñar como un atributo o un elemento sin pérdida de semántica pero existen criterios para decantar...
- Atributos
 - Normalmente se trata de objetos cuya existencia no tiene sentido fuera del objeto al que describen (p.e. adjetivos), metadatos, identificadores únicos, el idioma, ...
 - En general, todo aquello por lo que existe mayor interés en filtrarlo que en mostrarlo
- Ventajas
 - Más fácil de procesar por el software (mayor eficiencia)
 - Más legible, los atributos están próximos al elemento al que pertenecen

30

DTDs (XIII) (Recomendaciones para modelado de Atributos)

- Elementos, se debería optar por ellos...
 - Siempre que se quiera definir sub-elementos (ya sean partes del elemento principal o sus específicos)
 - Permiten crear vínculos
 - Siempre que queramos repetir el mismo elemento con distinto valor (los atributos tienen un único valor como máximo)
 - Siempre que el contenido sea mayor que una palabra (oraciones, párrafos, ...) y sobre todo si se quiere mostrar el texto en cuestión
 - Tienen entidad propia independientemente del resto de elementos
- Los documentos que priman a los atributos pueden ser más breves al no tener una etiqueta propia que abrir y cerrar

31

DTDs (XIV) (Problemas)

- Una DTD no sigue el formato de un documento XML **estándar**. Esto representa un **problema** para los **parsers**
- No se soportan **distintos tipos de datos** al estilo de los lenguajes de programación (CDATA, #PCDATA)
- No se pueden crear tipos de datos personalizados
- No se soportan los espacios de nombres (namespaces)
- El número de ocurrencias no se puede controlar al 100% (Ej. 2 ocurrencias)
- **Por estas y otras razones, surgen los XML Schemas (Esquemas)**

32

Referencias a Caracteres

- Permiten incluir cualquier carácter dentro de un documento XML
 - Basado en el conjunto de caracteres ISO/IEC 10646 (<http://xml.coverpages.org/xml-ISOents.txt>)
 - Dos formatos:
 - `&#xvalor`, valor representado en decimal
 - `&#xvalor`, valor representado en hexadecimal
- ```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<Libros>
<Libro Precio = "658 euros" Editorial = "Anaya">
<Titulo>La vida es sueño</Titulo> (DECIMAL)
<Autor>Calderón de la Barca</Autor> (DECIMAL)
</Libro>
</Libros>
```

33

## Entidades (I)

---

- Las entidades permiten:
  - Dar modularidad al texto evitando tener que escribir algo de forma repetitiva (Reuso)
  - Incluir caracteres prohibidos `&`, `>`, `<`, `"`, `'`
  - Incluir caracteres de otros idiomas: eñe, ...
- Comienzan por `&` y terminan en `;"`  
Como por ejemplo: `&amp;`;

34

## Entidades (II)

---

- Entidades predefinidas:
  - Signo menor lt < &lt;
  - Signo mayor gt > &gt;
  - Ampersand amp & &amp;
  - Apóstrofe apos ' &apos;
  - Comilla doble quot " &quot;

35

## Entidades (III)

---

- Tipos de entidades:
  - General y de Parámetro:
    - General:** contiene texto XML u otros caracteres
    - De Parámetro:** contiene texto XML que puede insertarse dentro de una DTD
  - Interna y Externa:
    - Interna:** contiene el texto dentro de una cadena entrecomillada
    - Externa:** hace referencia a un archivo externo
  - Analizada y no Analizada:
    - Analizada:** texto XML que será *parseado* en su punto de inserción
    - No Analizada:** no será *parseada*

36

## Ejercicio de Atributos:

---

- Hacer una DTD utilizando atributos:  
<?xml version="1.0" encoding="utf-8" ?>  
<Agenda>  
 <Persona>  
 <Nombre> Anabel </Nombre>  
 <Apellido> Fraga </Apellido>  
 <Sexo> Femenino </Sexo>  
 <DNI> 44444444-O </DNI>  
 <Nacionalidad> Española </Nacionalidad>  
 <Email> [anabel\\_fraga@mydomain.es](mailto:anabel_fraga@mydomain.es) </Email>  
 <Email> [anabel\\_fraga@fraga.es](mailto:anabel_fraga@fraga.es) </Email>  
 <Oficina>  
 <Direccion CP="28911"> Av. Universidad 30 </Direccion>  
 <Despacho> 2.1 B18 </Despacho>  
 <Email> [afraga@ie.inf.uc3m.es](mailto:afraga@ie.inf.uc3m.es) </Email>  
 <Telefono> 5555555 </Telefono>  
 <Telefono> 5555556 </Telefono>  
 </Oficina>  
 <Telefono> 5555558 </Telefono>  
 <Telefono> 5555559 </Telefono>  
 <Movil> 5557777 </Movil>  
 </Persona>  
</Agenda>

37

## XML Schemas (I)

---

- Actualmente existe una nueva recomendación de W3C de Mayo 2001 para definiciones de XML:

### **XML Schemas**

- Uso de notación XML para definiciones
- Limitación de uso: Actualmente existe una gran cantidad de documentos definidos con DTDs.

38

## XML Schemas (II) (Ejemplo)

```
<element name="Point">
 <complexType>
 <sequence>
 <element name="x"
 type="integer"/>
 <element name="y"
 type="integer"/>
 </sequence>
 </complexType>
</element>
```

point.xsd

```
<Point>
 <x>3</x>
 <y>4</y>
</Point>
```

point.xml

39

## XML Schemas vs. DTDs (I)

### Desventajas de las **DTDs**

- No escritas en sintaxis XML
- Poco uso de namespaces
- Pocos tipos de datos (y lo que es peor, no se pueden definir nuevos tipos de datos)
- Aunque se puede agrupar elementos mediante entities (%;) están poco desarrolladas

### Ventajas de las **DTDs**

- Muchas herramientas que lo soportan
- Existen muchos documentos: DTDs y XMLs basados en ellas
- Fácil de aprender

40

## XML Schemas vs. DTDs (II)

---

- **Ventajas:**
  - Permite multitud de tipos de datos (pe xs:date, xs:int, xs:language, ...)
  - Amplio uso de los namespaces
  - Permite agrupar elementos para su reutilización, permite herencia (Ejemplo: Datos Personales en distintos Dominios de uso)

41

## La Familia XML (I)

---

- **XPointer/XLink:** permiten referenciar a diferentes recursos, dentro o fuera del documento XML
- **XPath:** lenguaje de consulta para recorrer ficheros XML
- **XQL (XML Query Language):** útil para localizar y extraer elementos de un documento XML
- **XIRQL:** Una extensión de XQL para Recuperación de Información
- **XSLT:** Lenguaje para transformación de documentos XML
- **XSL-FO:** Expresa semántica de formateado de documentos, provee los medios para producir impresiones de alta calidad.

42

## XPath (II) (Ejemplo)

---

```
<catalogo>
 <libro>
 <titulo>Professional
 XML</titulo>
 <autor>Didier Martin et
 al.</autor>
 <editorial>Wrox</editorial>
 <anyo>2000</anyo>
 </libro>
 <libro>
 <titulo>XML Developer's
 Guide</titulo>
 <autor>Fabio
 Arciniegas</autor>
 <editorial>McGraw-
 Hill</editorial>
 <anyo>2001</anyo>
 </libro>
</catalogo>
```

- Todos los autores:  
**"/catalogo/libro/autor"**  
**"/catalogo/\*/autor"**  
**"//autor"**
- Todos los autores, con condición:  
**"/catalogo/libro[anyo>2001]/autor"**
- El texto de los elementos **autor**:  
**"/catalogo/libro/autor/text()"**
- El primer libro:  
**"/catalogo/libro[0]"**

43

## XPath (III)

---

- Expresiones numéricas  
**+ - \* div mod**
- Expresiones booleanas  
**and or**
- Expresiones de comparación  
**= != < <= > >=**

44

## XPath (IV)

---

- Funciones numéricas
  - **round ceiling floor**
  - **count number sum**
- Funciones booleanas
  - **boolean false true not**
- Funciones de cadenas de caracteres
  - **string string-length substring**
  - **substring-after substring-before**
  - **contains starts-with concat**
  - **normalize translate**

45

## XPath (V) (Unión)

---

- “|” sirve para calcular la unión de conjunto de nodos especificados por medio de *location paths*
- Ejemplos:
  - "//libro[anyo=2000]|//libro[anyo=2001]"**
  - "//libro[anyo=2000 or anyo=2001]"**

46

## Presentación en XML

---

- La presentación en HTML esta básicamente en los navegadores.
- Sería interesante *programar* la presentación (re-uso de código)
- Surgen las hojas de estilo:
  - **CSS**: Cascading Style Sheets (HTML)
  - **XSL**: eXtensible Style Language (XML) (XML + DTD o XML Schema + Fichero de Estilo XSL)

47

## METADATOS Y XML

---

48

## Namespaces (I)

---

- XML permite crear etiquetas 'casi' sin ninguna limitación en sus nombres
- Esto implica que, mezclar dos documentos, con diferentes etiquetas, podría resultar en una **duplicidad** de etiquetas
- Mediante la definición de espacios de nombres, se pueden **evitar** estas **colisiones**
- Tecnologías como XSL y otras muchas hacen uso de *Namespaces*

49

## Namespaces (II) (Definición)

---

- Un *namespace* se identifica por su prefijo.

Por ejemplo:

```
<xsl:stylesheet
 xmlns:xsl="http://www.w3.org/XSL/Transform/1.0">
```

donde:

- **xsl** es el prefijo del *namespace*
- **Stylesheet** es el nombre completo del *namespace*
- **http://www...** es la URI donde se puede encontrar más información sobre el estándar
- Puede incluir otros atributos como **version...**
- Como todo elemento XML, ha de **cerrarse**  

```
</xsl:stylesheet>
```

50

## Namespaces y qualified names (Qnames)

---

- Para escribir en XML metadatos hay que definir previamente la ubicación del vocabulario de metadatos (Namespaces) y un prefijo para hacer referencia al vocabulario empleado (Qname)

51

## Ejemplo NameSpace

---

- ```
<?xml version="1.0"?>
<!-- initially, the default namespace is "books" -->
<book xmlns='urn:loc.gov:books'
xmlns:dc="http://purl.org/DC"
xmlns:isbn='urn:ISBN:0-395-36341-6'>
  <title>XML el futuro</title>
  <dc:creator>yo mismo</dc:creatro>
  <isbn:number>1568491379</isbn:number>
  <notes>
    <!-- esto es un comentario en el ejemplo y html -->
    <p>This is a <i>funny</i> book! </p>
  </notes>
</book>
```

52

Metadatos

- **Registros:** repositorios para gestionar, recuperar, referenciar y reutilizar vocabularios de metadatos existentes. Estos registros suelen facilitar información sobre la definición, origen y localización del recurso. Actualmente, estándar ISO 11179
 - Proyecto Schemas, para RDF(S) y namespaces relacionados con proyectos de la UE (<http://www.schemas-forum.org/>);
 - Open Metadata Registry y ULIS son otros proyectos que recopilan metadatos relacionados con la Dublin Core Metadata Initiative (<http://dublincore.org/dcregistry/navigateServlet> y <http://avalon.ulis.ac.jp/registry/>);
- **Perfiles de Aplicación**

53

Metadatos famosos

- DC (Dublín Core) para la descripción de documentos. Con cualificadores
 - FOAF (Friend of a Friend) vocabulario sobre información personal y relaciones interpersonales. Sin vocabulario estable para realizar extensiones. Es generado automáticamente en websites que trabajan con blogs.
 - RSS (RDF Site Summary una de las siglas hay pa gustos) tiene un conjunto de metadatos multipropósito, suele ser utilizado principalmente para describir sitios web. Syndicate pages
 - Text Encoding Initiative (TEI) (<http://www.tei-c.org/>) marcado de e-text.
 - Encoded Archival Description (EAD) para descripción de archivos y colecciones especiales.
- Según Swoogle en Junio de 2004 que los NS, asociados a vocabularios de metadatos, más utilizados eran: FOAF (1.126.002 documentos), DC (1.126.002), MCVB (8.838), RSS(7.560 Junio, 80.000 septiembre de 2004), vCard (6.229) y Bio (6.183).

54

Etiqueta META en HTML

La etiqueta META se utiliza **dentro del encabezamiento HEAD** de una página HTML, para identificar, indizar y catalogar documentos

Los atributos de esta etiqueta se encuentran indicados en el RFC 1866 bajo la siguiente DTD (Document Type Definition):

```
<!ELEMENT META>
<!ATTLIST META
  http-equiv NAME      #IMPLIED
  name      NAME      #IMPLIED
  content   CDATA     #REQUIRED >
```

55

Editores metadatos

- <http://www.ukoln.ac.uk/cgi-bin/dcdot.pl>
- <http://vancouver-webpages.com/META/mk-metas.html>
- <http://www.lub.lu.se/cgi-bin/nmdc.pl>
- Reggie metadata.net/dstc
- <http://www.ukoln.ac.uk/metadata/new-dcdot>
- http://rainbow.arch.scriptmania.com/tools/adv_metatag_generator.html

56

Dublin Core: elementos (1)

Los 15 elementos:

Subject	ObjectType
Title	Form
Author	Identifier
Publisher	Relation
OtherAgent	Source
Date	Language
Type	Coverage
	Scheme

57

Ejemplo de un documento DC HTML

```
<META NAME="Title" CONTENT="FrontOffice selects Verity
for Microsoft Exchange basad document management system">
<META NAME="DC.Author" CONTENT="Padovani,
Marguerite">
<META NAME="DC.Author" CONTENT="Siegel, Gail">
<META NAME="DC.Publisher" CONTENT="Verity Inc.">
<META NAME="DC.Date" CONTENT="1996">
<META NAME="DC.Object" CONTENT="Press Release">
<META NAME="DC.Form" CONTENT="1 ASCII file">
<META NAME="DC.Language" CONTENT="English">
```

58

Calificadores DC (1)

- Propuestos en DC4 como Canberra Qualifiers.
 - Esquema.nombre_de_elemento.nombre_de_sub-elemento = "valor"
 - DC.Creator.personalName = "Scott Adams"

- Aprobación, el 17-04-00, de Dublin Core Qualifiers (qDC)
 - Lista no cerrada que formaliza el método de utilización

59

Cualificadores DC

Elemento DCMES	Elemento refinado	Sistema de codificación
Title	Alternative	
Creator		
Subject		LCSH MeSH DDC LCC UDC
Description	Table Of Contents Abstract	
Publisher		
Contributor		
Date	Created Valid Available Issued Modified	DCMI Period W3C-DTF
Type		DCMI Type Vocabulary
Format	Extent	
	Medium	IMT
Identifier		URI

60

Cualificadores DC

Source		URI
Language		ISO 639-2 RFC 1766
Relation	Is Version Of Has Version Is Replaced By Replaces Is Required By Requires Is Part Of Has Part Is Referenced By References Is Format Of Has Format	URI
Coverage	Spatial	DCMI Point ISO 3166 DCMI Box TGN
	Temporal	DCMI Period W3C-DTF
Rights		

61

Interoperabilidad

- El término interoperabilidad ha sido definido (ALA, 2000) como la capacidad que tienen algunos sistemas para intercambiar y utilizar información procedente de otro sistema diferente.
- La forma más usual de que exista es alineando vocabularios de metadatos y esquemas XML.
- La alineación puede ser estructural o lingüística

62

RDF (Resource Description Framework)

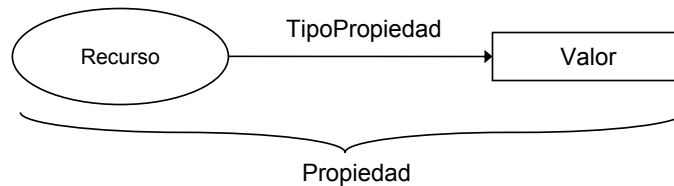
Validadores: <http://zoe.mathematik.uni-osnabrueck.de/RDF/parser.html>
<http://www.w3.org/RDF/Validator/>

Objetivos de RDF:

- interoperabilidad de metadatos a través de diferentes descripciones de recursos Web e intercambio de Metadatos.
- RDF trata de hacer compatibles diferentes estándares.
- Marco genérico de descripción de recursos
 - Colección de propiedades=RDF.
 - Cada propiedad tiene un tipo de propiedad y un valor
- Formato de metadatos
- Interoperabilidad entre aplicaciones
- Intercambio de descripciones de recursos

63

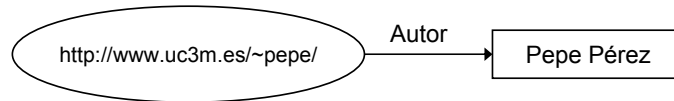
El modelo RDF



- Basado en un modelo matemático=triple
- Recursos Web representados por nodos URI
- Los conjuntos de propiedades se conocen como "descripciones"

64

RDF - ejemplo básico



“Pepe es el autor del recurso identificado por <http://www.uc3m.es/~pepe/>”

Sujeto/Resource: la URI ; Verbo/Propiedad: autor

Predicado/Values: Pepe Pérez

```
<rdf:description rdf:about="www.uc3m.es"  
dc:creator="Pepe">
```

65

Ejemplo de RDF

```
<?xml version="1.0" ?>
```

```
<RDF xmlns = "http://w3.org/TR/1999/PR-rdf-syntax-  
19990105#" xmlns:DC = "http://purl.org/DC#" >
```

```
<Description about = "http://www.amazon.com" >
```

```
<DC:Title> Ontologia </DC:Title>
```

```
<DC:Creator> Ruben Prieto-Diaz </DC:Creator>
```

```
<DC:Date> 1999-12-31 </DC:Date>
```

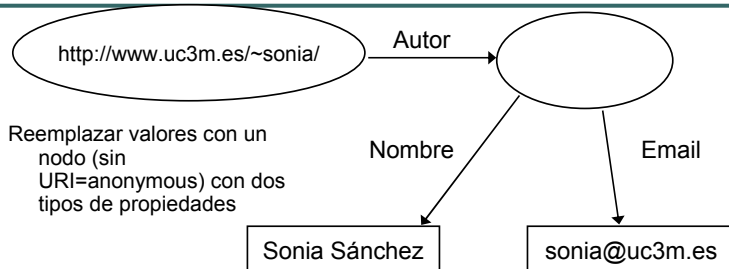
```
<DC:Subject> Metadata, RDF, Dublin Core </DC:Subject>
```

```
</Description>
```

```
</RDF>
```

66

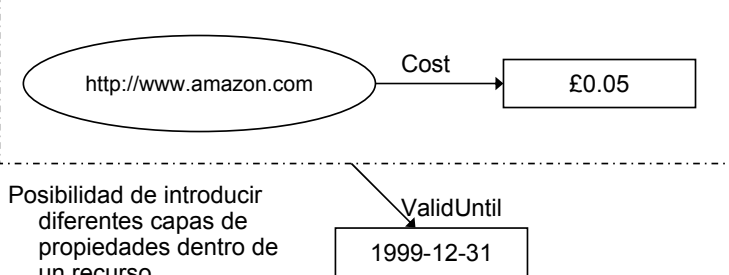
RDF - estructuración



```
<DC:Creator parseType="Resource">
  <vCard:FN> Pepe </vCard:FN>
  <vCard:TITLE> profe </vCard:TITLE>
  <vCard:EMAIL> sonia@uc3m.es
</vCard:EMAIL>
</DC:Creator>
...
```

67

RDF - reification



```
...
<Description about =
http://www.amazon.com
  bagID = "ID001" >
  <DC:Title> Ontologias </DC:Title>
  <DC:Creator> Ruben
  Prieto</DC:Creator>
  <ECOMM:Price>£0.05</ECOMM:Price>
</Description>

<Description aboutEach = "#ID001" >
  <ADMIN:ValidFrom> 1998-01-01
</ADMIN:ValidFrom>
  <ADMIN:ValidTo> 1999-12-31
</ADMIN:ValidTo>
</Description>
```

68

RDF - múltiples propiedades

```
...
<DC:Creator>
  <Bag>
    <li> Maddie Azzurii
  </li>
    <li> Corky Brown </li>
    <li> Jacky Crystal </li>
  </Bag>
</DC:Creator>.....
```

```
...
<DC:Creator>
  <Seq>
    <li> Maddie Azzurii </li>
    <li> Corky Brown </li>
    <li> Jacky Crystal </li>
  </Seq>
</DC:Creator>
...
```

```
...
<SOFT:Location>
  <Alt>
    <li> ftp://soft-sales.com.us/abc.exe </li>
    <li> ftp://soft-sales.com.au/abc.exe </li>
    <li> ftp://soft-sales.com.de/abc.exe </li>
    <li> ftp://soft-sales.com.uk/abc.exe </li>
  </Alt>
</SOFT:Location>...
```

69

RDF - namespaces

- Utilizados en XML para representar atributos
- Identifican Tipos de Propiedades
- Deben especificarse previamente
- Precedidos de dos puntos
 - `<DC:Title>Título del recurso</DC:Title>`
- Tienen asociados un URI

70

Ejemplo RDF

```
<?xml version="1.0"
<RDF xmlns:"http://w3.org/TR/1999/PR-rdf-syntax-
19990222#"
      xmlns:DC="http://purl.org/DC#">

  <Description about="http://www.ugr.es">
    <DC:Title> Web de la Universidad de Granada
    </DC:Title>
    <DC:Creator>Servicio de informática
    </DC:Creator>
    <DC:Date> 1998-02-08 </DC:Date>
    <DC:Description> Resumen del contenido del
    sitio</DC:Description>
  </Description>
</RDF>
```

71

RDF con varios vocabularios de metadatos

```
<?xml version="1.0"
<RDF xmlns:"http://w3.org/TR/1999/PR-rdf-syntax-19990222#"
      xmlns:DC="http://purl.org/DC#"
      xmlns:AGLS="http://na.gov.au/AGLS#"

  <Description about="http://www.ugr.es">
    <DC:Title> Web de la Universidad de Granada
    </DC:Title>
    <DC:Creator>Servicio de informática
    </DC:Creator>
    <DC:Date> 1998-02-08 </DC:Date>
    <AGLS:Function> Information managemen - Internet
  </AGLS:FunctionDescription>
  </Description>
</RDF>
```

72

¿Qué es RSS?

- RSS = “Really Simple Syndication”
 - RSS = “Rich Site Summary”
 - RSS = “RDF Site Summary”.
- Básicamente: RSS es un lenguaje XML para publicar simultáneamente (*sindicate*) noticias en Internet.

73

Posibilidades

- Noticias (prensa, anuncios)
 - Eventos
 - Información de proyectos
 - Bibliografías
 - Información de contacto
 - ...
- La principal ventaja: La fuente informa cuando se producen cambios.

74

Un poco más técnico ...

- Los archivos RSS se actualizan de forma regular y contienen **metadatos** sobre una fuente de noticias determinada y su contenido.
- Consta fundamentalmente de:
 - **Channel**: que representa la fuente de las noticias.
 - **Title**: título del canal.
 - **Link**: vínculo del canal.
 - **Description**: descripción del canal.
- Además, consta de uno o varios elementos **item** que representan elementos de noticias individuales, cada uno de los cuales debe disponer de un campo **title**, **link** o **description**.

75

Ejemplo

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
  <channel>
    <title> BCR: The Third Indicator</title>
    <link>http://www.bcr.org/publications/thirdind/</link>
    <description>The Third Indicator... included.</description>
    <lastBuildDate>Tue, 21 Sep 2004 21:37:39 GMT</lastBuildDate>
    <generator>ListGarden Program 1.01</generator>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <item>
      <title>WorldCat Resource Sharing Training</title>
      <link>http://www.bcr.org/publications/thirdind/2004/august/augsharetra
in04.html</link>
      <description>If you'd like to see what WorldCat Resource Sharing ...
www.oclc.org/ill/migration/ or view the WorldCat Resource Sharing
tutorial</description>
      <pubDate>Tue, 21 Sep 2004 19:29:47 GMT</pubDate>
      <guid isPermaLink="false">thirdind-2004-08-21-19-29-47</guid>
    </item> </channel></rss>
```

76

¿Cómo funciona RSS?

- El autor crea un fichero RSS.
- Los usuarios se suscriben al fichero a través de un lector de noticias o agregador.
- Cuando el autor actualiza el fichero RSS, los nuevos elementos se notifican automáticamente a los usuarios quedando a su disposición para su lectura.

77

¿Qué es un canal?

- Un canal, bitácora o *blog* (abreviatura de *weblog*, pronunciado “*we blog*”,) es un conjunto de noticias *online*, representadas en orden cronológico inverso e incluidas en un fichero RSS.
- También se utiliza para denominar el sistema que aloja y sirve un conjunto de canales.
- Típicamente, este tipo de sistemas incluyen enlaces entre ellos, proporcionando información adicional sobre los temas que incluyen.

78

¿Qué es un canal?

- Un canal, bitácora o *blog* (abreviatura de *weblog*, pronunciado “*we blog*”,) es un conjunto de noticias *online*, representadas en orden cronológico inverso e incluidas en un fichero RSS.
- También se utiliza para denominar el sistema que aloja y sirve un conjunto de canales.
- Típicamente, este tipo de sistemas incluyen enlaces entre ellos, proporcionando información adicional sobre los temas que incluyen.

79

¿Qué es un agregador?

- Una aplicación o un servicio remoto que, periódicamente, lee un conjunto de fuentes o **canales** en formato XML.
- Cuando detecta nuevos elementos, muestra un resumen de los mismos en un listado ordenado cronológicamente, comenzando por el más moderno.
- La aplicación necesaria para leer ficheros RSS.

80

Tipos de agregadores

- Clientes/agentes independientes
 - **FeedReader**, Radio UserLand
- Complementos PIM (*Personal Information Manager*)
 - **Pluck**, NewsGator, intraVnews
- Complementos de Navegador
 - **Firefox 1.0**, Sage
- Sitios Web
 - **Bloglines**, NewsIsFree

➔ Listado de agregadores
<http://www.lights.com/weblogs/rss.html>

81

Mas información

- Introducción a RSS
<http://www.maestrosdelweb.com/editorial/sindicacion/>
- RSS esquema <http://www.clikear.com/xsd/rss2.xsd>
- RDF Site Summary <http://web.resource.org/rss/1.0/spec>

- cANALES

- Librarian.net www.librarian.net
- Librarian's Rant blog.jalcorn.net
- LISNews www.lisnews.com
- The Shifted Librarian www.theshiftedlibrarian.com
- Travelin' Librarian travelinlibrarian.blogspot.com
- Unshelved www.overduemedia.com
- Free Range Librarian freerangelibrarian.com
- Crime in the Library crimeinthelibrary.blogspot.com
- Tame the Web www.tametheweb.com/ttwblog
- LibraryTectonics www.librarytectonics.info

82