T3
Llenguatges de publicació dels continguts de la xarxa

Xarxes de Computadors i Aplicacions

PAU ARTIGAS, DAVID CARRERA i JORDI TORRES
Departament d'Arquitectura de Computadors
UPC, setembre – 2009

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
- XHTML

Contingut del tema

- Descripció dels esquemes per representar informació estructurada en documents i els mecanismes per restringir-la i transformar-la.
- Tecnologies de xarxa estretament relacionades que es presentaran:
  - HTML
    - es descriuen els conceptes bàsics i els mecanismes per separar el contingut de la presentació: fulls d'estil (css).
  - XML
    - es descriu la sintaxi bàsica i a continuació els mecanismes per restringir un document : DTD I SCHEMA.
  - XSL i XSLT
    - Es descriu com transformar documents xml
  - XHTML
    - Es descriu la unificació de xml i html

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
- XHTML

Antecedents i evolució

- Etiquetes:
  - Delimiten un conjunt de text / dades
  - Dónen semàntica al text que delimiten
  - En general, han de tenirun inici i un final (hi ha excepcions)

  - Inici zona delimitada per una etiqueta: <etiqueta>
  - Final zona delimitada per una etiqueta: </etiqueta>

  - Exemple: <etiqueta> text delimitat </etiqueta>

- Atributs:
  - Completen la semàntica d'una etiqueta

  - Forma: <etiqueta atribut1="valor" atribut2="valor"> text </etiqueta>

Antecedents i evolució

- SGML: Standard Generalized Markup Language

  - Origen: GML el 1969.

  - Textual, basat en tags …
    - Un llenguatge amb marcadors i identificadors d'elements del document que poden també contenir altres elements i que ja afegia el concepte de definició formal de tipus de documents.

  - SGML és l'estàndar (ISO 8879) per a la representació de dades i documents estructurats. Inventat en 1974.

Antecedents i evolució

- HTML: Hiperdocuments WWW
  - HTML ha seguit una evolució molt ràpida que ha convertit el llenguatge en un objecte d'una gran disputa entre fabricants de navegadors (recordeu Netscape i Microsoft) i el consorsi Web encarregat de posar ordre a l'evolució del llenguatge.
- Això ha fet d'HTML un llenguatge en que el format de presentació està barrejat amb l'estructura (o contingut) del document.
- Cap allà el 1996 html començava a estar fora de control i es va recuperar les idees de:

  SGML – (complicacions de SGML) = XML

Antecedents i evolució

XML: SGML lleuger …

- El 1998 apareix XML 1.0 amb les seves fulles d'estil. Està pensat no només per a les persones que miren una finestra de navegador, també està pensat com a llenguatge d'intercanvi de dades entre màquines
  - En ocasions es precís transformar els documents xml d'un format a un altre i per això s'utilitzen els fulls d'estil de transformacions (xslt)
- La ràpida introducció de la web i del xml ha estat possible per la facilitat i gratuïtat d'eines desenvolupades per la comunitat de manera participativa:
  - Universitat d'Illinois al 1993 : Mosaic (html 2.0)
  - Diverses llibreries ("parsers") per tractar documents xml s'han fet públiques, fet que ha comportat que xml fos un format "de moda".

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

HTML

- Presentació
- Estructuració d'un document HTML
  - Etiquetes d'estructuració del document
  - Etiquetes d'estructuració del contingut
  - Etiquetes d'entrada de dades
  - Etiquetes de formateig de dades
- Scripting de client
- CSS: Cascade Style Sheets

HTML: introducció

- Llenguatge per codificar documents de la WWW
  - Sintaxis i semàntica definida per especificació
  - Consorci World Wide Web (W3C) http://www.w3.org
- Característiques:
  - Basat en etiquetes (tags): <tag> … </tag>
  - Codificació d'estructura lògica (no presentació)
  - Enllaços a altres objectes (per valor o "inline"/per ref a URL)
- Cada vegada més complex,…
  - menor codificació manual -> més estricte:
  - format intern d'editors (de html o genérics)
  - Netscape, Macromedia Dreamweaver, MS Office, MS FrontPage…

HTML: introducció

- HTML was originally developed by Tim Berners-Lee
- HTML 2.0 (November 1995, see [RFC1866]) was developed under the aegis of the Internet Engineering Task Force (IETF)
- HTML+ (1993) and HTML 3.0 (1995, see [HTML30]) proposed much richer versions of HTML
- Efforts of the World Wide Web Consortium's HTML Working Group to codify common practice in 1996 resulted in HTML 3.2 (January 1997, see [HTML32]).

HTML: introducció

- HTML 4 (1998) extends HTML with mechanisms for style sheets, scripting, frames, embedding objects, improved support for right to left and mixed direction text, richer tables, and enhancements to forms, offering improved accessibility for people with disabilities.
- HTML 4.01 (1999) is a revision of HTML 4.0 that corrects errors

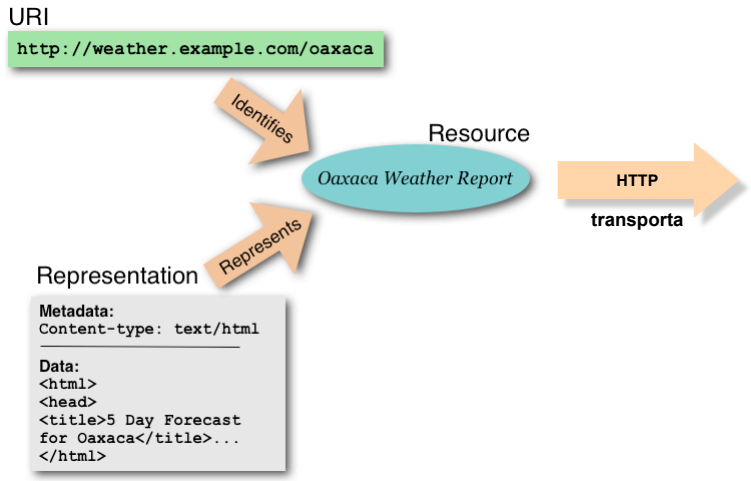
Font:

http://www.w3.org/TR/REC-html40/intro/intro.html#h-2.2.1

HTML: introducció

- Arquitectura de les aplicacions web:
  - El protocol **HTTP** per transferir documents web del servidor al client.
  - El llenguatge **HTML** de representació de la documentació web, que inclou a més de text, les regles de format i presentació visual.
  - La **URL** (uniform resource locator) per referènciar unívocament documents web en la xarxa.

HTML: introducció



HTML: estructura

Etiquetes amb funcions estructurals de document:

- Inici i final del document HTML:
  - <html> i </html>
- Capçalera:
  - <head> i </head>
- Cos:
  - <body> i </body>

Exemple de l'estructura bàsica d'un document:

```
<html>
   <head>
      <title>Document bàsic</title>
   </head>
   <body>
      Un document molt senzill
   </body>
</html>
```

HTML: estructura

- Etiquetes que estructuren el contingut:
  - Marcs (frames): Divideixen un document en subdocuments

```
<frameset cols="10%,*">
  <frame src="frame1.html">
  <frame src="frame2.html">
</frameset>
```

  - Taules (tables): Estructuren les dades d'un document

```
<table width="100%">
  <tr>
    <td>A</td>
    <td>B</td>
  </tr>
  <tr>
    <td colspan=2>C</td>
  </tr>
</table>
```

A	B
C	

HTML: estructura

- Etiquetes d'entrada de dades:
  - Formularis (forms): Permeten a l'usuari la introducció de dades

```
<form method="post" action="/bin/cmd">
  Importe
  <input type="text" name="valor">
  Verificar
  <input type="checkbox" name="ver" value="1">
  <input type="submit" value="Comprar">
</form>
```

Importe
1500€
Verificar ☑ Comprar

HTML: estructura

- Etiquetes de formateig de dades:
  - Enllaços

```
<a href="http://www.upc.es"> text </a>
```

  - Imatges:

```
<img src=pdf.gif>
```

  - Llistes:

```
Llista ordenada:
<ol>
  <li>Primer</li>
  <li>Segon</li>
</ol>
Llista no ordenada:
<ul>
  <li>Primer</li>
  <li>Segon</li>
</ul>
```

Llista ordenada:
1. Primer
2. Segon

Llista no ordenada:
- Primer
- Segon

HTML: estructura

<!--> <DEL> <INPUT> <SAMP>
< <DFN> <INS> <SCRIPT>
<A> <DIR> <ISINDEX> <SELECT>
<ABBREV> <DIV> <KBD> <SMALL>
<ACRONYM> <DL> <LANG> <SPACER>
<ADDRESS> <DT> <LH> <SPOT>
<APPLET> <DD> <LI> <STRIKE>
<AREA> <EM> <LINK> <STRONG>
<AU> <EMBED> <LISTING> <SUB>
<AUTHOR> <FIG> <MAP> <SUP>
<B> <FN> <MARQUEE> <TAB>
<BANNER> <FONT> <MATH> <TABLE>
<BASE> <FORM> <MENU> <TBODY>
<BASEFONT> <FRAME> <META> <TD>
<BGSOUND> <FRAMESET> <MULTICOL> <TEXTAREA>
<BIG> <H1> <NOBR> <TEXTFLOW>
<BLINK> <H2> <NOFRAMES> <TFOOT>
<BLOCKQUOTE> <H3> <NOTE> <TH>
<BQ> <H4> <OL> <THEAD>
<BODY> <H5> <OVERLAY> <TITLE>

 <H6> <P> <TR>
<CAPTION> <HEAD> <PARAM> <TT>
<CENTER> <HR> <PERSON> <U>
<CITE> <HTML> <PLAINTEXT> <UL>
<CODE> <I> <PRE> <VAR>
<COL> <IFRAME> <Q> <WBR>
<COLGROUP> <IMG> <RANGE> <XMP>
<CREDIT>

HTML: scripting de client

- En certes circumstàncies interessa que el client (navegador habitualment) pugui realitzar un cert processat del document
  - Validació de camps (ex: camp nom no buit)
  - Càlcul de valors (ex: conversió d'unitats)
  - …
- En aquests casos on el client és qui fa els càculs parlem d'scripting de client
  - Cal incorporar llenguatges que permetin implementar aquestes tasques

HTML: scripting de client

- Javascript:
  - Llenguatge d'scripting de client basat en Java
  - Disposa de diferents "dialectes" en funció del client (navegador)
  - Exemple:
    - En fer click en un enllaç s'obre un avís:

```
<script>
   <a href="pagina.html" onClick="alert('Has fet click');"> Text enllaç </a>
</script>
```

HTML: scripting de client

Exemple:

```
<html>
<head>
<script language="javascript"><!--
function checkform()
{
  if (document.myform.txtfirstname.value == "")
  {
    alert("please enter your first name");
    document.myform.txtfirstname.focus();
  }
  else
  {
    document.myform.submit();
  }
}
//--></script>
</head>
<body>
<form name="myform" action="somecgiprogram" method="post">
first name:
<input type="text" name="txtfirstname"><br>
<input type="button" name="btn" value="submit form" onclick="checkform();">
</form>
</body>
</html>
```

HTML: css

- HTML no hauria de fer-se servir per a donar estil a les dades que conté
- Habitualment, però, la informació de format i estructura està barrejada.
  - Per exemple s'usen marques com <font> per canviar l'aspecte d'una frase, enlloc d'indicar el seu significat estructural (és un títol de primer nivell)
  - O s'usen taules per col·locar un text en una cantonada de la pàgina, enlloc d'usar-les per indicar l'organització de dades de manera tabulada.
- La solució correcta és usar fulls d'estil:
  - Cascade Style Sheets (CCS)

HTML: css

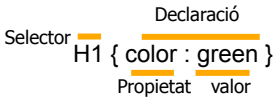
- Els fulls d'estil es creen per a controlar l'aparença (en el sentit més visual de la paraula) d'un document HTML:
  - Incorporat en HTML 4.0
  - Complementen la info estructural d' HTML
  - Important la separació estructura i estil
- Per què "en cascada"?
  - Info d'estil que es va superposant
  - Van "caient" (aplicant) sobre el document …
- També pot ser usat en altres markup languages (XHTML, XML)
  - Referència: http://www.w3.org/MarkUp/Guide/Style

HTML: css

- Des del 1996 el W3C ha anat fent noves aportacions:
  - CSS1 (Nivel 1), 12/96 Format senzill, presentació en pantalla, ~50 propietats: color, font-size, …
  - CSS2, 5/98 afegeix més de 70 propietats. Permet diverses presentacions (impresa, auditiva "aural", etc.).
  - CSS3 entre d'altres coses divideix CSS en mòduls

HTML: css

- Regla per a mostrar una etiqueta html



- ¿Què podem controlar?
  - Fonts (color, size, caps, font, etc)
  - Background (image, color, tiling properties)
  - Text (spacing, line-height, alignment, decoration, word-spacing)
  - Box properties (margin, border, float)
  - List properties (image for bullets)
  - Links (visited, hover, active, link)

HTML: css

- Els estils poden ser de diferents tipus:
  - Redefineixen l'estil de tots els tags d'un cert tipus a tot el document:
    H1 {font-style: bold; font-size: 12pt}
  - Creen un estil propi que es pot aplicar en parts del document
    .normal {color; black; font-style: bold; font-size: 12pt}
- També es poden aplicar estils a XML

HTML: css

- Exemples:

```
body { color: black; }
h1, h2 { color: maroon; }

body { font-family: "Times New Roman"; }
h1 { font-family: Helvetica; }
h2 { font-family: Arial; }

h1 { font-style: normal; font-weight: bold; }
h2 { font-style: italic; font-weight: normal; }
p { font-weight: bold; }
```

HTML: css

- Hi ha diferents maneres d'associar estils:
  - *Inline* amb HTML, com a atribut *style*

    <h1 **style="color: blue; font-style: italic"**>COLOR BLAU</h1>
  - Per a tot el document (*document level*), element *style*

    <head> …

    <style type="text/css"> <!-- H1 {color: blue; font-style: italic}**--> </style>** …

    <!--…--> Per a clients web que no entenen CSS (sintaxi ≠ html)
  - Extern, full d'estil extern

    estil_gral.css:

    H1 {color: blue; font-style: italic}

    .normal {color; black; font-style: bold; font-size: 12pt}

    doc html:

    <head>…<link rel=stylesheet type="text/css" href="**http::://x.org/estilo_gral.css**">

    </head>

     Text de paràgraf

! compartit per tots els document d'una site web, permetent que **un canvi en el document estil afectarà automàticament a tots els documents que usin aquest fitxer d'estil**.

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

XML

- XML: eXtensible Markup Language
  (llenguatge d'etiquetatge extensiu)
- A partir d'elements <x> … </x>
  - Proposat pel W3C ("SGML lleuger")
    - Basat en alfabet universal Textual (Unicode UTF-8)
- XML
  - permet als usuaris crear els seus tags
  - permet separar el contingut de la presentació

XML: Motivació

- Separació de presentació i dades:

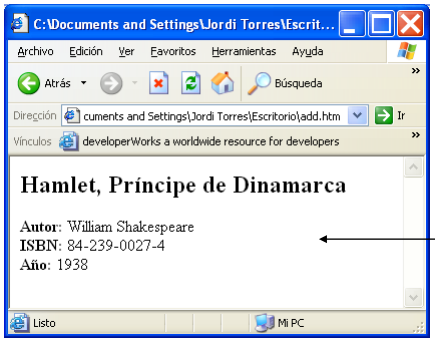
```
<h2>Hamlet, Príncipe de Dinamarca</h2><p>
<b>Autor: </b>William Shakespeare<br>
<b>ISBN: </b>84-239-0027-4<br>
<b>Año: </b>1938</p>
```

HTML **barreja dades i presentació**.

P.ex:

-Nom d'un autor

-Text en negreta



La versió HTML pot ser interpretada per persones mitjançant els browsers, **però els programes NO poden extreure fàcilment** la informació del document HTML

XML: Motivació

- XML estructura les dades i elimina, dins del document, informació sobre la presentació

```
<?xml version="1.0"?>
<libro>
  <autor> <apellido>Shakespeare</apellido>
        <nombre>William</nombre>
  </autor>
  <titulo>Hamlet  Principe de Dinamarca</titulo>
  <isbn>84-239-0027-4</isbn>
  <fecha>1938</fecha>
</libro>
```

- Ara amb XML una persona o **o un programa** podrà obtenir la informació (apellido, nombre, título, isbn, fecha) amb facilitat

XML: usos habituals

- Usos de XML:
  - Per transportar informació entre bases de dades
  - Per enviar informació que es mostra a l'usuari
  - Com format llegible per expressar dades estructurades
  - Per codificar dades en la xarxa
  - Com a alternativa a formats de dades binàries en RPC

- Restriccions:
  - Les dades binàries han de ser enviades com a Base64 o enviar a banda del document XML (un enllaç, com fa HTML per exemple)
  - Força text! (pot comprimir-se)

XML: components

- Per a poder estructurar la informació, XML es basa en els següents components:
  - La **sintaxi** del llenguatge (fixada pel W3C):
    - Cada etiqueta ha d'obrir-se i tancar-se
    - Les etiquetes són "case-sensitive" (majúscules/minúscules)
    - …
  - El **conjunt d'elements** vàlid (fixat per l'usuari):
    - **espai de noms** vàlid per als elements (etiquetes)
      - <libro>, <autor>, <apellido>, <nombre>, <titulo> …
    - les **restriccions d'us**:
      - quan es poden usar i quins valors pode prendre

XML: sintaxi

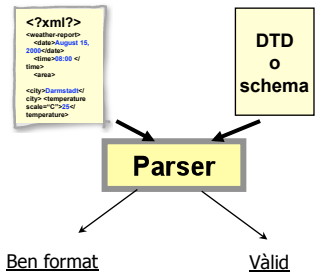
- Elements poden estar imbricats
  <libro><autor>William S.</autor>…</libro>
- Elements poden tenir atributs
  <libro autor="William S" titulo="Hamlet">
- Els elements s'han de tancar sempre (a HTML no sempre)
  (Incorrecte) <libro><autor>William S.</libro>
- Elements sense dades poden tancar-se al final de la etiqueta
  <libro autor="William S" titulo="Hamlet"/>
  <libro autor="William S" titulo="Hamlet"></libro>
- Codificació de caràcters especials
  <libro titulo="El "Aleph""/>

XML: sintaxi

- XML és més restrictiu que HTML
- Alguns exemples:
  - Acceptat a HTML i XML:
    Paràgraf 1** **Paràgraf 2
    ****Negreta i Itàlica****
    <font color=**"#ffff66"**>
  - Acceptat en HTML però no a XML:
    Paràgraf 1** **Paràgraf 2
    ****Negreta i Itàlica****
    <font color=**#ffff66**>

XML: validesa

- DEFINICIONS:
  - Document XML "ben format":
    - Un document que **compleix amb la sintaxi** de xml
  - Document XML "vàlid":
    - Si a més de ser ben format, **compleix amb un conjunt de restriccions especificades en un fitxer de restriccions**



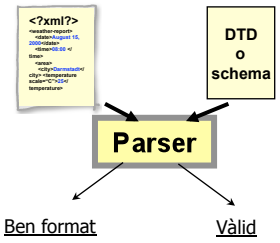
XML: validesa

- Els fitxers de restriccions expressen carácterístiques sobre el domini de dades que és representat pel document XML
- Els creen els usuaris o les comunitats
- Expresen gramàtiques que han de complir els elements dels documents XML (a part de la sintaxi genèrica del llenguatge XML)

XML: validesa

- Existeixen diversos formats per als fitxers de restriccions, proposats pel W3C:

1. DTD:
   - Definició Tipus Document;
   - 1ª generació: heretat de SGML,
   - sintaxi no XML, sense tipus de dades, …
2. XML Schema:
   - format XML,
   - tipus de dades,
   - més restriccions



Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

DTD

- DTD: Document Type Definition
  - Defineix els blocs constructius d'un document XML.
  - En defineix l'estructura i la llista legal d'elements
- El DTD es pot declarar inline en el document XML o com a referència externa

DTD

- Elements
  - Són els principals blocs constructius de XML
- Attributs
  - Ofereixen informació addicional sobre els elements
- Entitats
  - Variables usades per a definir text d'ús habitual
- #PCDATA
  - PCDATA significa "parsed character data" (text processat)
- #CDATA
  - CDATA significa "character data" (text no processat)

DTD: Elements

- Els elements que poden aparèixer en el document XML es declaren en el DTD a través d'una declaració de tipus ELEMENT:

<!ELEMENT *nom_element* (*contingut*)>

Exemple:

- Un llibre té autors

<!ELEMENT llibre (autor+)>

- XML associat que compleix el DTD:

<llibre> <autor> Emili Teixidor </autor> </llibre>

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- El *contingut* d'un element pot ser:
  - EMPTY: No té informació entre els tags
    <!ELEMENT element-name EMPTY>
  - #PCDATA:
    <!ELEMENT element-name (#PCDATA)>
  - #CDATA:
    <!ELEMENT element-name (#CDATA)>
  - ANY:
    <!ELEMENT element-name ANY>
  - Estructures més complexes (amb subelements)

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Una sola instància del subelement:

<!ELEMENT element-name (child-element-name)>

DTD

```
<!ELEMENT llibre (títol)>
<!ELEMENT títol (#PCDATA)>
```

XML vàlid

```
<llibre>
   <títol>El llibre de les mosques</títol>
</llibre>
```

XML no vàlid

```
<llibre>
   <títol>El llibre de les mosques</títol>
   <títol>Retrat d'un assassí d'ocells</títol>
</llibre>
```

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Seqüència: Mateix ordre i un sol cop cadascun

<!ELEMENT element-name (child-element-name,child-element-name,.....)>

DTD

```
<!ELEMENT autor (nom, cognom)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom (#PCDATA)>
```

XML vàlid

```
<autor>
   <nom>Emili</nom>
   <cognom>Teixidor</cognom>
</autor>
```

XML no vàlid

```
<autor>
   <nom>Emili</nom>
</autor>
```

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Mínim una instància del subelement:

<!ELEMENT element-name (child-element-name+)>

DTD

```
<!ELEMENT autor (llibre+)>
<!ELEMENT llibre (#PCDATA)>
```

XML vàlid

```
<autor>
   <llibre>El llibre de les mosques</llibre>
   <llibre>El príncep Alí</llibre>
</autor>
```

XML no vàlid

```
<autor>
</autor>
```

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Zero o més instàncies del subelement:

<!ELEMENT element-name (child-element-name*)>

DTD

```
<!ELEMENT pare (fill*)>
<!ELEMENT fill (#PCDATA)>
```

XML vàlid

```
<pare>
   <fill>Pau</fill>
   <fill>David</fill>
</pare>
```

XML no vàlid

```
??
```

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Zero o una instància del subelement:

<!ELEMENT element-name (child-element-name?)>

DTD

```
<!ELEMENT fill (mare?)>
<!ELEMENT mare (#PCDATA)>
```

XML vàlid

```
<fill>
   <mare>Montse</mare>
</fill>
```

XML no vàlid

```
<fill>
   <mare>Montse</mare>
   <mare>Yolanda</mare>
</fill>
```

DTD: Elements

<!ELEMENT *nom_element* (*contingut*)>

- Estructures d'elements amb subelements:
  - Una de les instàncies dels subelements:

<!ELEMENT element-name (child-element-name | child-element-name)>

DTD

```
<!ELEMENT operació (get | put)>
<!ELEMENT get EMPTY>
<!ELEMENT put EMPTY>
```

XML vàlid

```
<operació>
   <get/>
</operació>
```

XML no vàlid

```
<operació>
   <get/>
   <put/>
</operació>
```

DTD

- Blocs constructius d'un document DTD:
  - Elements
  - Attributs
  - Entitats
  - #PCDATA
  - #CDATA

DTD: Atributs

- Un element pot tenir una sèrie d'atributs que en completin el valor semàntic
- Els atributs que poden aparèixer en un element es declaren com una llista en el DTD:

```
<!ATTLIST element-name attribute-name attribute-type default-value>
```

Exemple:

- Un llibre té autors

```
<!ATTLIST llibre títol CDATA #REQUIRED>
```

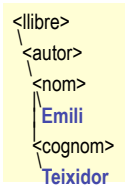
- XML associat que compleix el DTD:

```
<llibre títol="El llibre de les mosques">
```

DTD: Atributs

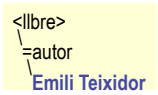
Un document XML pot imaginar-se com un arbre

```
<llibre>
 <autor>
  <nom>Emili</nom>
  <cognom>Teixidor</cognom>
 </autor>
</llibre>
```



en front de

```
<llibre autor="Emili Teixidor">
```



- Elements:
  branques que poden **subdividir-se** (y extendre's)
- Atributs:
  fulles (**terminals**) que no poden dividir-se més (no estructurats)
  sense modificar el document anterior.

DTD: Entitats

- Les entitats representen seqüències de text que es referencien des del document XML
- S'usen a mode de macro per a reduir duplicacions de continguts
- Els caràcters especials de XML es defineixen com a entitats
- El format és el següent:

```
<!ENTITY entity-name "entity-value">
```

" → "
& → &
< → <
> → >
' → '

Exemple:

- En un document es referencia molts cops la FIB:

```
<!ENTITY FIB "Facultat d'Informàtica de Barcelona">
```

- En el XML es fa la referència a l'entitat:

```
<desc>XCA és una assignatura de la &FIB;</desc>
```

DTD: CDATA i PCDATA

- #PCDATA significa dades processades.
  - **És text que serà processat per un parser.**
  - Les etiquetes contingudes pel text seran tractades com a entitats i seran expandides
- #CDATA also means character data.
  - **És text que NO serà processat per un parser.**
  - Les etiquetes contingudes pel text NO seran tractades com a entitats i NO seran expandides

DTD: CDATA i PCDATA

Sobre #PCDATA, #CDATA i CDATA

- #PCDATA: Dades **processades**
  - El següent exemple no és vàlid (no compleix el DTD!)

```
<!ELEMENT doc-html (#PCDATA)>
```

```
<doc-html> <html> exemple </html> </doc-html>
```

- #CDATA: Dades **no processades**
  - Però cal protegir els caràcters especials!

```
<!ELEMENT doc-html (#CDATA)>
```

```
<doc-html><![CDATA[<html> exemple </html>]]></doc-html>
```

- CDATA: Indica seqüència de caràcters.
  - S'usa en atributs
  - Les dades **es processe**

```
<!ATTLIST llibre títol CDATA>
```

DTD: Exemple final

xca.dtd

```
<!ELEMENT biblioteca (llibre*) >

<!ELEMENT llibre (autor+) >

<!ATTLIST llibre
            titol CDATA #REQUIRED
            any CDATA #REQUIRED >

<!ELEMENT
        autor (nom,cognom,cognom?) >

<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom (#PCDATA)>
```

```
<?xml version="1.0“encoding="UTF-8"?>
<!DOCTYPE biblioteca SYSTEM "xca.dtd">
<biblioteca>
  <llibre titol="El Aleph" any="1968">
     <autor>
       <nom>Jorge Luis</nom>
       <cognom>Borges</cognom>
     </autor>
  </llibre>
  <llibre titol="Rayuela" any="1963">
     <autor>
       <nom>Julio</nom>
       <cognom>Cortazar</cognom>
     </autor>
  </llibre>
</biblioteca>
```

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

Schema

Limitacions dels DTDs:

- La sintaxi és diferent de XML
  - No pot ser parsejada amb un parser XML
  - No és còmode treballar amb dos tipus de sintaxi diferentes
- DTDs no permeten especificar tipus particulars de dades.

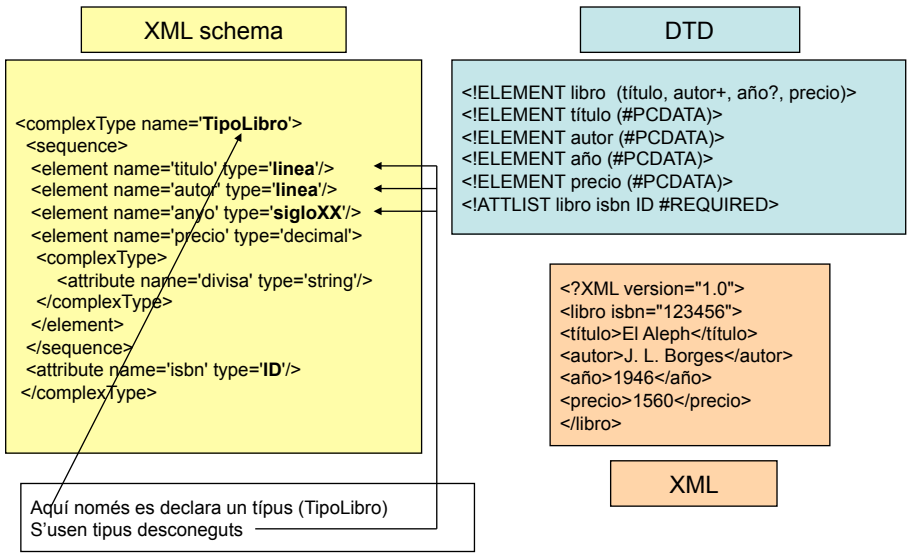
XML Schema Definition (XSD)

- Basat en la sintaxi, eines i tecnologia de XML
- La definició va d'acord amb la sintaxi de XML
- Permet la definició i tipus de dades complexes

Schema

- Funcions:
  - Definir/restringir el contingut i estructura de documents XML usant XML
  - Substituir DTD
- Característiques:
  - Facilita mapeigs amb estructures de dades de programa
  - És un document XML (parsers XML el poden tractar)
- Elimina bona part de les limitacions dels DTDs:
  - Tipus de dades definits per l'usuari
  - Quantificadors (minOccurs, maxOccurs)
  - Refinament (suporta herència de tipus)

Schema: exemple



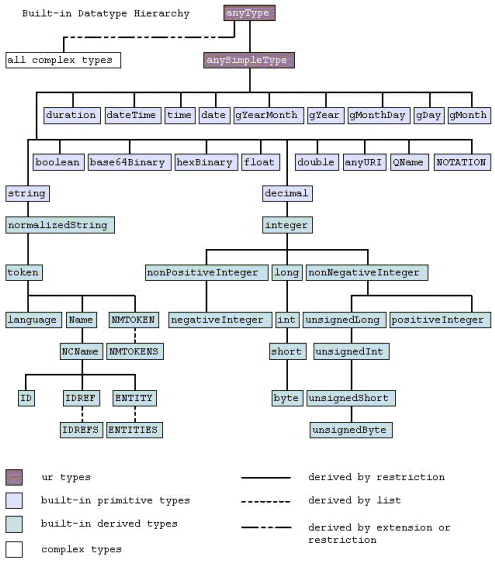
Schema: exemple

```
<schema>
<simpleType name='linea'>
  <restriction base='string'> <maxLength value='50'/> </restriction>
</simpleType>

<simpleType name='sigloXX'>
  <restriction base='integer'> <minInclusive value='1900'/> <maxInclusive value='1999'/> </restriction>
</simpleType>

<element name='libro' type='TipoLibro'/>

<complexType name='TipoLibro'>
  <sequence>
   <element name='titulo' type='linea'/>
   <element name='autor' type='linea'/>
   <element name='anyo' type='sigloXX'/>
   <element name='precio' type='decimal'>
    <complexType>
      <attribute name='divisa' type='string'/>
    </complexType>
   </element>
  </sequence>
  <attribute name='isbn' type='ID'/>
 </complexType>
</schema>
```

Schema: tipus de dades

- Simple:
  - No té atributs
  - No conté altres elements
  - Són els tipus de dades natius, amb o sense restriccions
- Complex:
  - No simple (conté atributs i/o subelements)
- Als tipus de dades se'ls poden afegir restriccions:
  - length
  - minLength
  - maxLength
  - pattern
  - enumeration
  - whiteSpace
  - maxInclusive
  - maxExclusive
  - minExclusive
  - minInclusive
  - totalDigits
  - fractionDigits

Schema: tipus simples

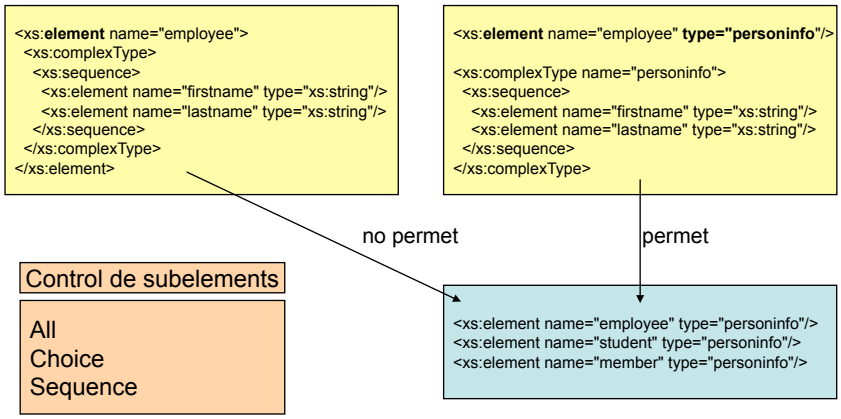


Els tipus de dades simples són aquests i restriccions sobre aquests

Existeixen molts tipus de dades natius!

Schema: tipus complexes

- Poden declarar-se implícits o com a tipus de dades “amb nom”



Schema: exemple final

```
<?xml version="1.0" encoding="utf-8"?>
<book isbn="0836217462">
   <title>Being a Dog Is a Full-Time Job</title>
   <author>Charles M. Schulz</author>
   <character>
      <name>Snoopy</name>
      <friend-of>Peppermint Patty</friend-of>
      <since>1950-10-04</since>
      <qualification>extroverted beagle</qualification>
   </character>
   <character>
      <name>Peppermint Patty</name>
      <since>1966-08-22</since>
      <qualification>bold, brash and tomboyish</qualification>
   </character>
</book>
```

Schema: exemple final

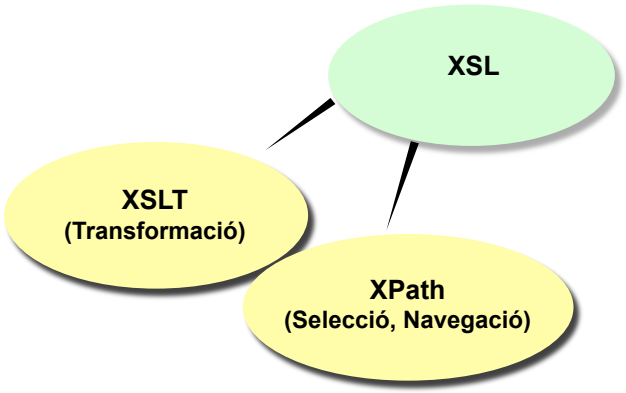
```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="book">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="title" type="xsd:string"/>
        <xsd:element name="author" type="xsd:string"/>
        <xsd:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="name" type="xsd:string"/>
              <xsd:element name="friend-of" type="xsd:string“
                                minOccurs="0" maxOccurs="unbounded"/>
              <xsd:element name="since" type="xsd:date"/>
              <xsd:element name="qualification" type="xsd:string"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="isbn" type="xsd:string"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

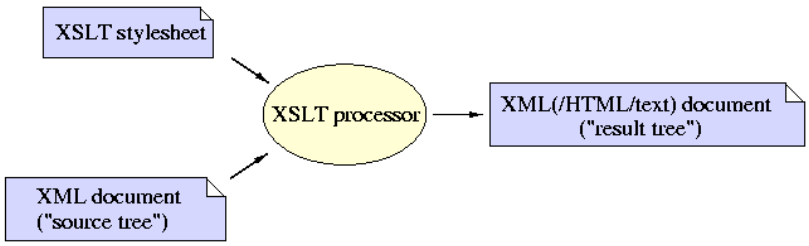
XSL

- XSL: Extensible Stylesheet Language
  http://www.w3.org/Style/XSL



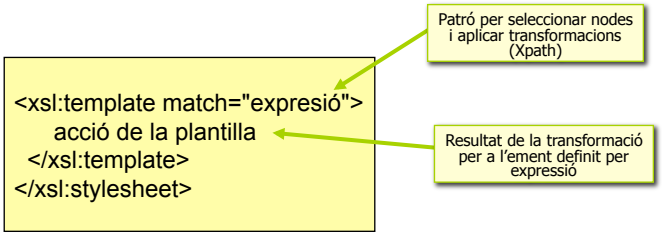
XSL

- Esquema conceptual de funcionament



XSL

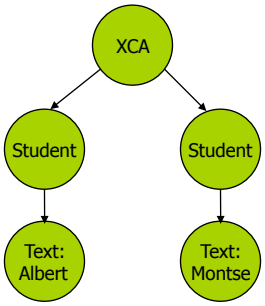
- Procés de transformació de documents:
  - Un conjunt de regles que s'apliquen als elements.
  - Cada regla descriu quina sortida hi haurà per cada element del XML original
    - Cal poder identificar els elements! (Xpath)



XSL: Xpath

- XPath: permet navegar en l'arbre/fitxer XML

```
<XCA>
  <Student>Albert</Student>
  <Student>Montse</Student>
</XCA>
```



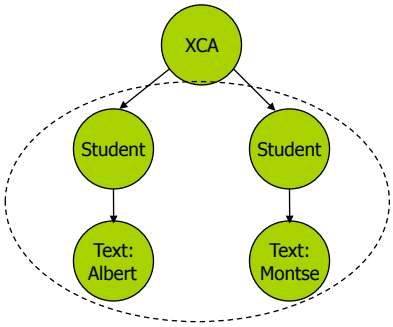
XSL: Xpath

- Exemple:

  /XCA/Student

  selecciona tots els estudiants (cadascun d'ells)

```
<XCA>
  <Student>Albert</Student>
  <Student>Montse</Student>
</XCA>
```

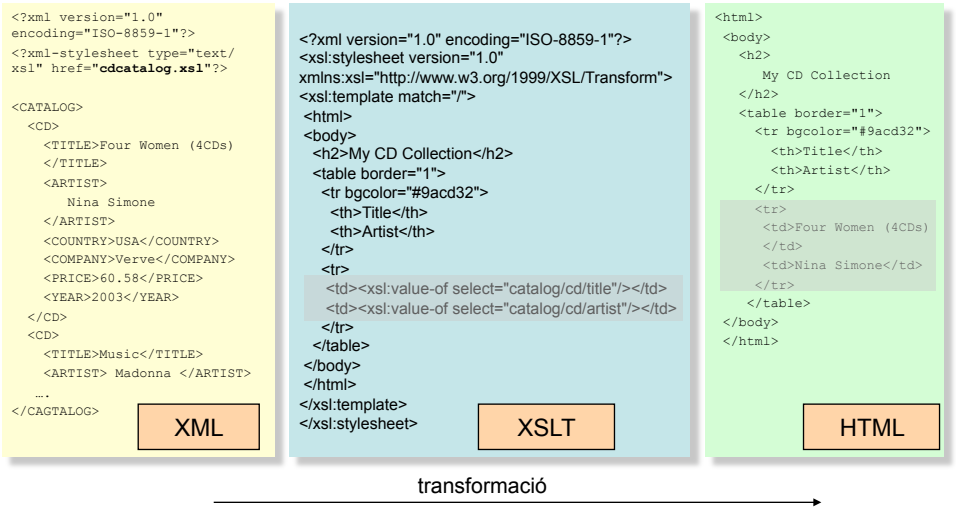


XSL: XSLT

- Definició del patró de transformació:
  - Element <xsl:template>
    - Un full d'estil XSL consisteix en un conjunt de regles anomenades template.
    - Cada element <xsl:template> conté regles per aplicar quan un node determinat fa matching.
    - Atribut match=
      - S'usa per associar el template amb un element XML.
      - També pot ser usat per definir templates per tota una branca d'un document XML.
  - Element <xsl:value-of>
    - Extreu el valor del node seleccionat.
    - L'atribut pot contenir una expressió
  - Element <xsl:for-each>
    - ens permet realitzar bucles en XSL (selecciona cada element XML d'un conjunt de nodes especificat).

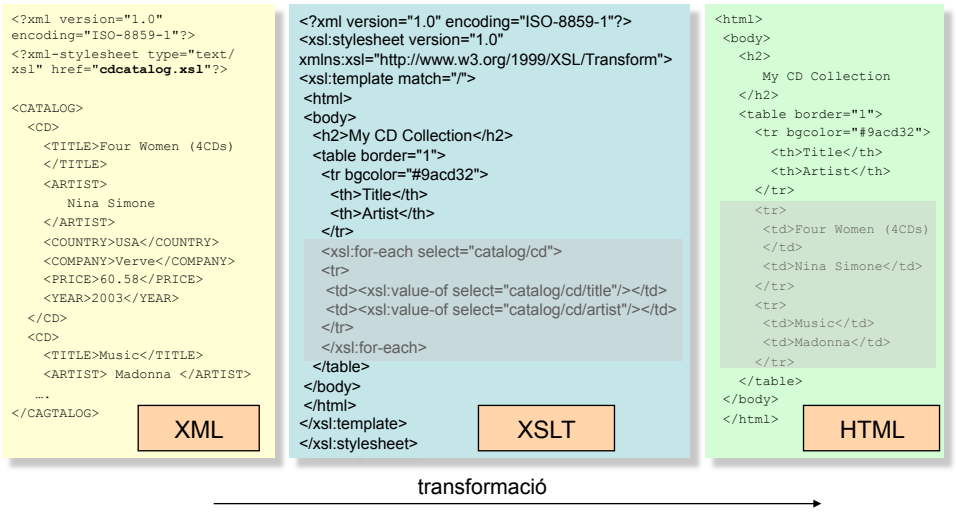
XSL: XSLT

- Exemple 1:



XSL: XSLT

- Exemple 2 (ús de for-each):



Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

XML Namespaces

- Situació:
  - Es volen fusionar dades de dos XMLs que tenen elements amb semàntica diferent però mateix nom

```
<?xml version="1.0" ?>
…
<Addr>
    <Street>Gran</Street>
    <Number>65</Number>
    <City>Argentona</City>
    <Zip>08310</Zip>
</Addr>
…
```

```
<?xml version="1.0" ?>
…
<Server>
    <Name>General</Name>
    <Addr>192.168.12.1</Addr>
</Server>
…
```

XML Namespaces

- Solució dolenta:
  - Canviar el nom d'un dels documents
    - Fins i tot pot no ser possible fer-ho!

```
<?xml version="1.0" ?>
…
<Addr>
    <Street>Gran</Street>
    <Number>65</Number>
    <City>Argentona</City>
    <Zip>08310</Zip>
</Addr>
…
<Server>
    <Name>General</Name>
    <AddrServ>192.168.12.1</AddrServ>
</Server>
```

XML Namespaces

- Solució bona:
  - Declarar espais de noms
    - Cada element pertany a un espai (definit per un sol fitxer de restriccions)
    - Aquell element és validat segons les restriccions del fitxer que el defineix

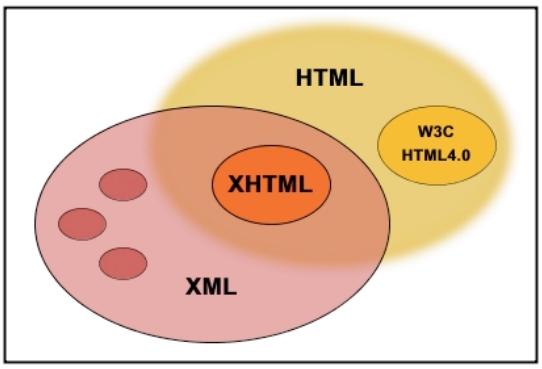
```
<h:html xmlns:xdc="http://www.xml.com/books"
        xmlns:h="http://www.w3.org/HTML/1998/html4">
 <h:head><h:title>Book Review</h:title></h:head>
 <h:body>
   <xdc:bookreview>
   <xdc:title>XML: A Primer</xdc:title>
     <h:table>
     <h:tr align="center">
        <h:td>Author</h:td><h:td>Price</h:td>
        <h:td>Pages</h:td><h:td>Date</h:td></h:tr>
     <h:tr align="left">
        <h:td><xdc:author>Simon St. Laurent</xdc:author></h:td>
        <h:td><xdc:price>31.98</xdc:price></h:td>
        <h:td><xdc:pages>352</xdc:pages></h:td>
        <h:td><xdc:date>1998/01</xdc:date></h:td>
     </h:tr>
     </h:table>
   </xdc:bookreview>
 </h:body>
 </h:html>
```

Contingut

- Antecedents i evolució
- HTML
  - CSS
- XML
  - DTD
  - SCHEMA
  - XSL
  - Espais de noms
- XHTML

XHTML

- Formulació de HTML 4.01 en XML
  - Introducció de restriccions basades en DTD o XSD



XHTML

- Alguns exemples:
  - Acceptat a HTML i XHTML:

    Paràgraf 1** **Paràgraf 2
    ****Negreta i Itàlica****
    <font color=**"#ffff66"**>

  - Acceptat en HTML però no a XHTML:

    Paràgraf 1** **Paràgraf 2
    ****Negreta i Itàlica****
    <font color=**#ffff66**>

Diagrama relació de llenguatges