UT4: Lenguaje de marcas para el almacenamiento y transmisión de información

Objetivos de UT4 y UT5

- Describir la información transmitida en los documentos XML y sus reglas.
- Identificar las tecnologías relacionadas con la definición de documentos XML.
- Analizar la estructura y sintaxis específica utilizada en la descripción.
- Crear descripciones de documentos XML.
- Utilizar descripciones en la elaboración y validación de documentos XML.
- Asociar las descripciones con los documentos.
- Utilizar herramientas específicas.
- Documentar las descripciones.

Índice

Objetivos de UT4 y UT5 1
Estructura y sintaxis de documentos XML 2
Reglas XML 3
Elemento XML 3
XML bien formado y válido 3
Declaración XML 3
Espacio de nombres 4
DTD 6
Declaración del tipo de documento 6
Declaraciones de elementos 6
Declaraciones de atributos 8
Declaraciones de entidades (ENTITY) 8
Declaraciones de notaciones (NOTATION) 9

Estructura y sintaxis de documentos XML

Repaso UT1:

- XML son las siglas en inglés de eXtensible Markup Language (lenguaje de marcas extensible).
- Propuesta para simplificar SGML.
- Desarrollado por el World Wide Web Consortium (W3C). Es un metalenguaje. Permite definir otros lenguajes.
- Pensado para **describir datos**, no para mostrar datos.
- Suficientemente flexible para ser utilizado en distintos ámbitos:
  - XHTML.
  - RSS, Atom, OPML.
  - MathML para notación matemática.
  - SVG para describir gráficos vectoriales: https://developer.mozilla.org/en-US/docs/Web/SVG
  - KML para representar datos geográficos: https://developers.google.com/kml/documentation/?hl=es

  http://en.wikipedia.org/wiki/List_of_XML_markup_languages
- Un documento XML es generalmente texto plano.
- Consta de etiquetas o marcas: marcas de inicio y de fin:
  - Las de inicio delimitadas por < y >
  - Las de fin delimitadas por </ y >
  - Las etiquetas permiten definir elementos: <etiqueta>Valor</etiqueta>
- Cada elemento puede contener otros elementos.
- Los elementos pueden contener atributos: <etiqueta atributo="valor">Valor</etiqueta>
- En XML hay caracteres reservados:
  - " "
  - ' '
  - < <
  - > >
  - & &
- Instrucciones de procesamiento. <?aplicación datos ?>
- Comentarios: <!-- comentario -->
- Secciones CDATA: <![CDATA["datos"]]>
- Los documentos XML forman una estructura de árbol.

```
<raíz>
    <padre>
        <hijo> ..... </hijo>
        <hermano> .... </hermano>
    </padre>
</raíz>
```

Reglas XML

- Todos los elementos XML deben tener una etiqueta de apertura y otra de cierre.
- Si el elemento no tiene contenido podremos sustituir las etiquetas por <etiqueta />.
- Las etiquetas XML distinguen entre mayúsculas y minúsculas.
- En XML todos los elementos deben estar anidados correctamente. No pueden estar entrelazados.
- Los documentos XML deben tener un único elementos raíz.
- En XML los elementos pueden contener atributos en la etiqueta inicial.
- En XML los valores de los atributos deben ir entre comillas (simples o dobles).
- En XML un elemento no puede tener dos atributos con el mismo nombre.
- En XML se deben sustituir los caracteres reservados por sus respectivas entidades.
- Los comentarios y las instrucciones de procesamiento no pueden estar dentro de etiquetas.

Elemento XML

Un elemento está formado por la etiqueta de inicio, la etiqueta de fin y todo lo contenido entre ellas.
Podrá contener:

- Texto
- Otros elementos
- Atributos

Reglas para los nombres de elementos y atributos:

- Pueden contener letras, números y otros caracteres (guion bajo, guion, punto).
- No pueden contener un espacio en blanco.
- Deben comenzar por letras o el carácter de subrayado.
- No pueden comenzar por un número, un guion o un punto.
- No pueden comenzar por xml (ya sea en minúsculas o en mayúsculas).
- No hay limitación en cuanto a la longitud.

Evitar en la medida de lo posible el uso de atributos para caracterizar información. Utilizar los atributos para los metadatos.

XML bien formado y válido

Los documentos bien formados son aquellos que son sintácticamente correctos, contiene un único elementos raíz y todas las etiquetas están correctamente anidadas.
Un XML válido, debe:

- Estar bien formado.
- La estructura debe encajar con la definición del tipo de documento (DTD) o esquema (XML Schema, Relax NG). Se comprobará:
  - Que elementos y atributos se permiten.
  - Estructura de los elementos y los atributos.
  - Orden de los elementos.
  - Valores de los datos de los elementos y los atributos.
  - Unicidad de valores dentro de un documento.

Declaración XML

Los documentos XML deben empezar con una declaración XML:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

Debe estar al principio del documento.

- El atributo version debe tener el valor 1.0. (http://www.w3.org/TR/REC-xml/)
- El atributo encoding nos permite indicar la codificación de caracteres utilizada en el
- documento. Es opcional. Si se omite utiliza el conjunto de caracteres Unicode.

- El atributo standalone indica si el documento es independiente o se basa en la información de fuentes externas. Es opcional. Si se omite se supone que el valor es no. Si el valor es no puede requerirse un DTD externo.

```
<?xml version=”1.0” encoding=”UTF-8” standalone=”yes”?>
<alumno>
   <nombre>Francisco</nombre>
   <apellido>Rodríguez</apellido>
   <direccion>
         <calle>Federico Silva</calle>
         <cp>49600</cp>
   </direccion>
</alumno>
```

PRÓLOGO

CUERPO

Validar en: http://validator.w3.org/

Espacio de nombres

En XML los nombres de los elementos están definidos por el desarrollador. Muchas veces surgen conflictos con nombres de elementos al intentar mezclar documentos XML de distintas fuentes. Para evitar el problema se le añade un prefijo al elemento que identifica el espacio de nombres. Para poder utilizar el prefijo debe ser definido el espacio de nombres.
El **espacio de nombres** es definido por el atributo xmlns en la etiqueta inicial de un elemento.
xmlns:prefijo="URI"
El espacio de nombres se puede declarar en los elementos donde se utiliza o en el elemento raíz.
http://www.w3.org/TR/REC-xml-names/

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

<html:html xmlns:html='http://www.w3.org/1999/xhtml'>
<html:head>
      <html:title>Virtual Library</html:title>
</html:head>
<html:body>
      <html:p>Moved to
      <html:a href=”http://vlib.example.org'>vlib.example.org</html:a>
      </html:p>
</html:body>
</html:html>
```

La declaración del espacio de nombres se aplica al elemento en que está especificada y a todos los elementos pertenecientes al contenido de ese elemento.

Se pueden declarar varios prefijos de espacios de nombres como atributos de un mismo elemento.

```
<?xml version="1.0"?>
<!-- both namespace prefixes are available throughout -->
<bk:book xmlns:bk='urn:loc.gov:books'
         xmlns:isbn='urn:ISBN:0-395-36341-6'>
      <bk:title>Cheaper by the Dozen</bk:title>
      <isbn:number>1568491379</isbn:number>
</bk:book>
```

Se considera que se aplica un espacio de nombres por defecto al elemento en que está declarado (si ese elemento no tiene prefijo de espacio de nombres), y a todos los elementos sin prefijo pertenecientes al contenido de ese elemento.

```
<?xml version="1.0"?>
<!-- initially, the default namespace is "books" -->
<book xmlns='urn:loc.gov:books'
      xmlns:isbn='urn:ISBN:0-395-36341-6'>
      <title>Cheaper by the Dozen</title>
      <isbn:number>1568491379</isbn:number>
      <notes>
            <!-- make HTML the default namespace for some commentary -->
            <p xmlns='http://www.w3.org/1999/xhtml'>
                  This is a <i>funny</i> book!
            </p>
      </notes>
</book>
```

DTD

Document Type Definition (definición de tipo de documento). Especifica que elementos pueden aparecer en un documento y dónde, así como el contenido y los atributos. Un documento válido incluye una declaración de tipo de documento que identifica la DTD que satisface el documento.
Hay que distinguir entre declaración de tipo de documento y definición de tipo de documento.
La DTD presenta una lista de todos los elementos, atributos y entidades que utiliza el documento.
Cuando un documento tiene una declaración de tipo de documento y el documento se ajusta a la DTD se dice que el documento es válido.

En los próximos apartados vamos a ver:

- Declaración del tipo de documento.
- Declaraciones de elementos.
- Declaraciones de atributos.
- Declaraciones de entidades.
- Declaraciones de notaciones.

Declaración del tipo de documento

Un documento válido incluye una referencia a la DTD con la que se debe comparar. Se incluye en el prólogo del documento XML, después de la declaración XML. Puede tratarse de una referencia interna o de una referencia externa.
Sintaxis para declaración interna:

```
<!DOCTYPE elemento_raiz [ declaración de elementos ]>
```

Sintaxis para declaración externa:

```
<!DOCTYPE elemento_raiz SYSTEM "url_dtd">
<!DOCTYPE elemento_raiz PUBLIC "identificador_publico" "url_dtd">
```

Existe la posibilidad de que haya declaración interna y externa a la vez.

```
<!DOCTYPE elemento_raiz SYSTEM "url_dtd" [ declaración de elementos]>
```

Declaraciones de elementos

Todos los elementos usados en un documento válido, deben declararse en la DTD de un documento con una declaración de elemento.
Sintaxis:

```
<!ELEMENT nombre_del_elemento especificación_de_contenido>
```

El nombre del elemento puede ser cualquier nombre XML.
Especificación de contenido:

- Datos de tipo carácter (#PCDATA).
- Elementos hijos (nombre_elemento_hijo).
- Elemento vacío EMPTY.
- Sin restricciones ANY.
- Secuencias. Un elemento suele tener más de un hijo. (nombre_elemento_hijo1, nombre_elemento_hijo2).
- Opciones. Las instancias de un elemento pueden contener hijos distintos. (nombre_elemento_hijo1 | nombre_elemento_hijo2).
- Número de hijos. No todas las instancias de un elemento tienen que tener los mismos hijos. Sufijos a añadir al nombre del elemento hijo:
  - ? el elemento hijo puede aparecer cero o una vez. (nombre_elemento_hijo?)
  - * el elemento hijo puede aparecer cero o más veces (nombre_elemento_hijo*)
  - + el elemento hijo puede aparecer una o más veces (nombre_elemento_hijo+)

- Paréntesis, para aplicar a:
  - Las secuencias, las opciones y los sufijos suelen aparecer combinados.
  - Una secuencia o una opción pueden aparecer entre paréntesis y tener un sufijo.
  - Los paréntesis pueden estar anidados.
- Contenido mixto (#PCDATA | nombre_elemento_hijo)*.

Ejemplo:

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<!DOCTYPE alumno [
      <!ELEMENT nombre (#PCDATA)>
      <!ELEMENT apellido (#PCDATA)>
      <!ELEMENT calle (#PCDATA)>
      <!ELEMENT cp (#PCDATA)>
      <!ELEMENT direccion (calle, cp)>
      <!ELEMENT alumno (nombre, apellido, direccion)>
]>

<alumno>
      <nombre>Francisco</nombre>
      <apellido>Rodríguez</apellido>
      <direccion>
            <calle>Federico Silva</calle>
            <cp>49600</cp>
      </direccion>
</alumno>
```

O por separado:
DTD:

```
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT apellido (#PCDATA)>
<!ELEMENT calle (#PCDATA)>
<!ELEMENT cp (#PCDATA)>
<!ELEMENT direccion (calle, cp)>
<!ELEMENT alumno (nombre, apellido, direccion)>
```

XML:

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<!DOCTYPE alumno SYSTEM “alumno.dtd”>
<alumno>
      <nombre>Francisco</nombre>
      <apellido>Rodríguez</apellido>
      <direccion>
            <calle>Federico Silva</calle>
            <cp>49600</cp>
      </direccion>
</alumno>
```

Declaraciones de atributos

Además de declarar sus elementos, un documento válido tiene que declarar todos los atributos de dichos elementos.
Sintaxis:
`<!ATTLIST elemento nombre_atributo tipo_atributo valor_predeterminado>`
Una sola declaración ATTLIST puede declarar múltiples atributos para el mismo elemento. Si el mismo atributo se repite en varios elementos, debe declararse en cada elemento donde aparece.
Valores predeterminados:

- #IMPLIED. El atributo es opcional. No se proporciona valor predeterminado.
- #REQUIRED. El atributo es obligatorio. No se proporciona valor predeterminado.
- #FIXED. El valor del atributo no puede cambiar. Aparece entre comillas.
- Valor literal. El valor predeterminado aparece entre comillas.

Tipos de atributo:

- CDATA. Puede contener cualquier cadena de texto
- NMTOKEN. Debe contener los mismos caracteres que un nombre XML
- NMTOKENS. Contiene uno o más NMTOKEN separados por un espacio en blanco
- ID. Debe contener un nombre XML único dentro del documento. Cada elemento solo puede tener un atributo ID
- IDREF. Debe ser un nombre XML. Se refiere al atributo de tipo ID de algún elemento en el documento. Se utilizan para establecen relaciones entre los elementos
- IDREFS. Contiene una lista de nombres XML separados por un espacio en blanco. Cada uno de los nombres XML se refiere al ID de un elemento.
- ENTITY. Contiene el nombre de una entidad sin analizar, declarada en alguna
- parte en la DTD.
- ENTITIES. Contiene los nombres de una o más entidades sin analizar, declaradas en cualquier parte en la DTD, separados por un espacio en blanco.
- NOTATION. Contiene el nombre de una notación declarada en la DTD del documento Enumeración Lista de todos los valores posibles para el atributo, separados por barras verticales. Cada valor debe tener los mismos caracteres que un nombre XML.
  <!ATTLIST elemento atributo (valor1 | valor2 | valor3) valor_predeterminado>

Declaraciones de entidades (ENTITY)

- XML predefine cinco entidades:
  < (<) > (>) & (&) " (") ' (')
- Las referencias de entidad se definen con la declaración ENTITY.
- Una entidad ofrece una entrada abreviada para el documento XML o la DTD.
- Las entidades de clasifican en:
  - Internas o externas.
  - Analizadas o no analizadas.
  - Generales o Parámetro.

Entidades generales internas:

- Son abreviaturas definidas en la DTD del documento XML.
- Consta de nombre de la entidad y texto de reemplazo de la entidad.
- El nombre de la entidad deber ser un nombre XML.
- Son siempre entidades analizadas.
- Para referenciar la entidad en el documento XML: &nombre_entidad;
- Una vez reemplazada la referencia a la entidad por su contenido, pasa a ser parte del documento XML y es analizada por el procesador XML.

Ejemplo:

```
<!ENTITY nombre "texto de sustitución">
<etiqueta>Contenido: &nombre;</etiqueta>
```

Entidades generales externas analizadas:

- Obtienen su contenido de otro sitio.
- Consta de nombre de la entidad, la palabra SYSTEM y el URI (*Universal Resource Identifier*).
- El nombre de la entidad deber ser un nombre XML.
- La entidad externa no contendrá prólogo, es decir, ni declaración XML ni declaración de tipo de documento.
- Puede que el analizador no reemplace la referencia de la entidad con el documento en el URI.

Ejemplo:

```
<!ENTITY nombre SYSTEM "archivo.xml">
&nombre;
```

Entidades no analizadas:

- Si el contenido de la entidad es un archivo de cualquier tipo no XML, el procesador XML no debe analizarlo.
- Siempre son entidades generales y externas.
- Consta de nombre de la entidad, la palabra SYSTEM y el URI (*Universal Resource Identifier*) .
- El nombre de la entidad deber ser un nombre XML.
- Una entidad sin analizar no puede referenciarse. Las referencias de entidad se usan sólo con entidades analizadas.
- XML no garantiza ningún comportamiento determinado de ninguna aplicación que se encuentre entidades sin analizar.

Ejemplo: `<!ENTITY nombre SYSTEM "logo.jpg" NDATA jpg>`

Entidades de parámetro internas:

- Se utilizan en la DTD y no en el documento XML.
- Se pueden utilizar para agrupar ciertos elementos de la DTD que se repiten mucho.
- Se declara de forma parecida a la referencia de entidad general.
- Aparece el carácter % entre ENTITY y el nombre de la entidad.
- Las referencias de entidades de parámetro sustituyen el carácter & por el carácter %.

Ejemplo:

```
<!ENTITY % nombre "<!ELEMENT ......>">
%nombre;
```

Entidades de parámetro externas:

- Se utilizan en la DTD y no en el documento XML.
- No pueden ser utilizadas en DTD internas.
- Se pueden utilizar para agrupar ciertos elementos de la DTD que se repiten mucho.
- Se declara de forma parecida a la referencia de entidad general.
- Aparece el carácter % entre ENTITY y el nombre de la entidad.
- Las referencias de entidades de parámetro sustituyen el carácter & por el carácter %.

Ejemplo:

```
<!ENTITY % nombre SYSTEM "archivo.dtd">
%nombre;
```

Declaraciones de notaciones (NOTATION)

Permiten definir el formato de archivos externos no XML.

Ejemplo:

```
<!NOTATION jpg SYSTEM "image/jpeg">
<!ENTITY nombre SYSTEM "logo.jpg" NDATA jpg>
<!ELEMENT imagen EMPTY>
<!ATTLIST imagen img ENTITY #REQUIRED>
<imagen img="nombre" />
```