XPATH

Conversión y adaptación de documentos XML

- Todo el procesamiento realizado con un fichero XML está basado en la posibilidad de direccionar o acceder a cada una de las partes que lo componen, de modo que podamos tratar cada uno de los elementos de forma diferenciada.
- La forma de seleccionar información dentro de él es mediante el uso de XPath, que es la abreviación de lo que se conoce como XML Path Language. Con XPath podremos seleccionar y hacer referencia a texto, elementos, atributos y cualquier otra información contenida dentro de un fichero XML.
- XPath sirve para decir cómo debe procesar una hoja de estilo el contenido de una página XML, pero también para poder poner enlaces o cargar en un navegador zonas determinadas de una página XML, en vez de toda la página.
- XPath en sí es un lenguaje sofisticado y complejo, de tipo declarativo, pero distinto de los lenguajes procedurales que solemos usar (C,C++, Basic, Java...).

Introducción a XPath

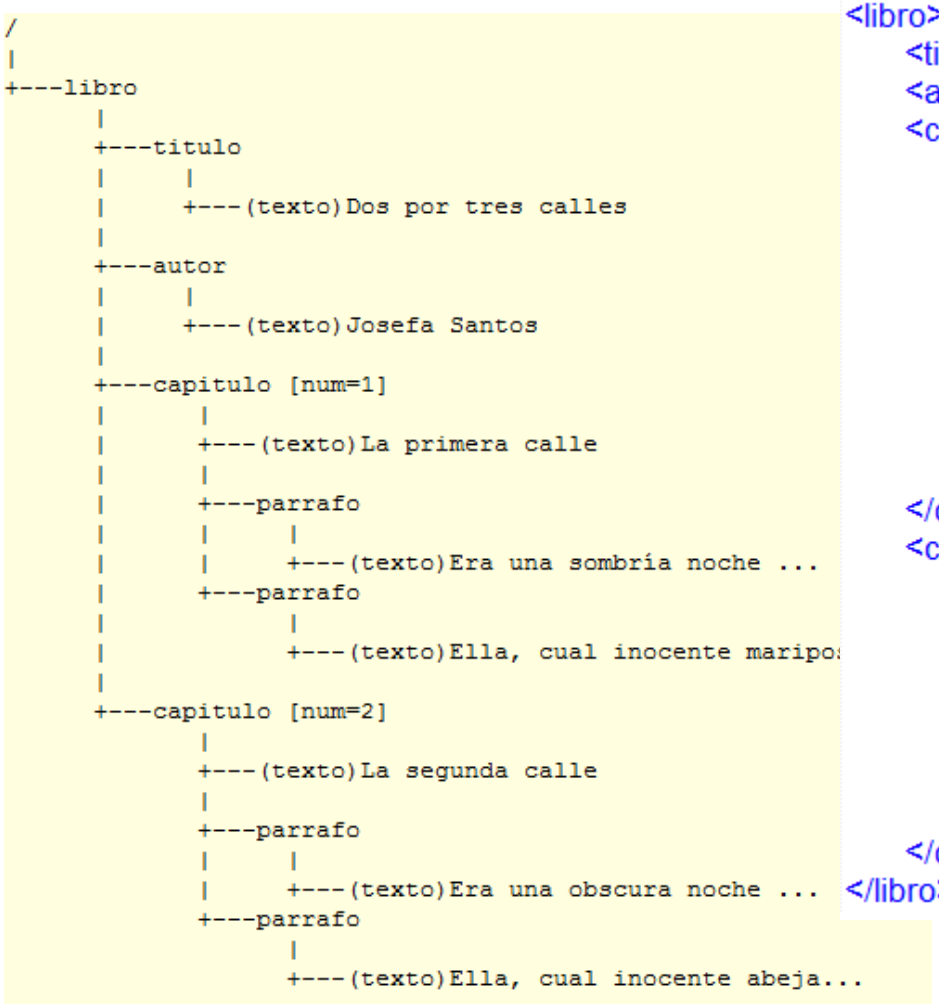
- XPath es una especificación del W3C.
- Define cómo acceder a partes de un documento XML.
- Se basa en relaciones de “parentesco” entre nodos.
- Su estilo de notación es similar a las rutas de los ficheros, pero se refiere a nodos en un documento XML:
  - Ejemplo: /fecha/dia
- XPath se usa en XSLT, pero también en XSL-FO, XPointer, XLink, y otros.

- En XSLT, XPath se utiliza en los valores de atributos (atributos tales como match o select, empleados en las etiquetas xsl:value-of y xsl:template respectivamente)

Modelo de datos de Xpath (1)

- Un documento XML es procesado por un analizador (o parser) construyendo un árbol de nodos.
  - Este árbol comienza con un elemento raíz, que se diversifica a lo largo de los elementos que cuelgan de él y acaba en nodos hoja, que contienen sólo texto, comentarios, instrucciones de proceso o incluso que están vacíos y sólo tienen atributos.
- El árbol de nodos:
  - Comienza en el nodo raíz
  - Acaba en los nodos hoja
- XPath selecciona partes del documento XML basándose en la estructura en árbol.

Modelo de datos de Xpath (2)




```
<?xml version="1.0" encoding="UTF-8"?>
<libro>
    <titulo>Dos por tres calles </titulo>
    <autor>Josefa Santos</autor>
    <capitulo num="1">
        La primera calle
        <parrafo>
            Era una sombria noche del mes de agosto.....
        </parrafo>
        <parrafo destacar="si">
            Ella, cual inocente
            <enlace href="http://www.enlace.css">mariposa</enlace>
            que surca el cielo en busca de .....
        </parrafo>
    </capitulo>
    <capitulo num="2">
        La segunda calle
        <parrafo>
            Era una oscura noche del mes de octubre.....
        </parrafo>
        <parrafo >
            Ella, cual inocente abeja ..
        </parrafo>
    </capitulo>
</libro>
```

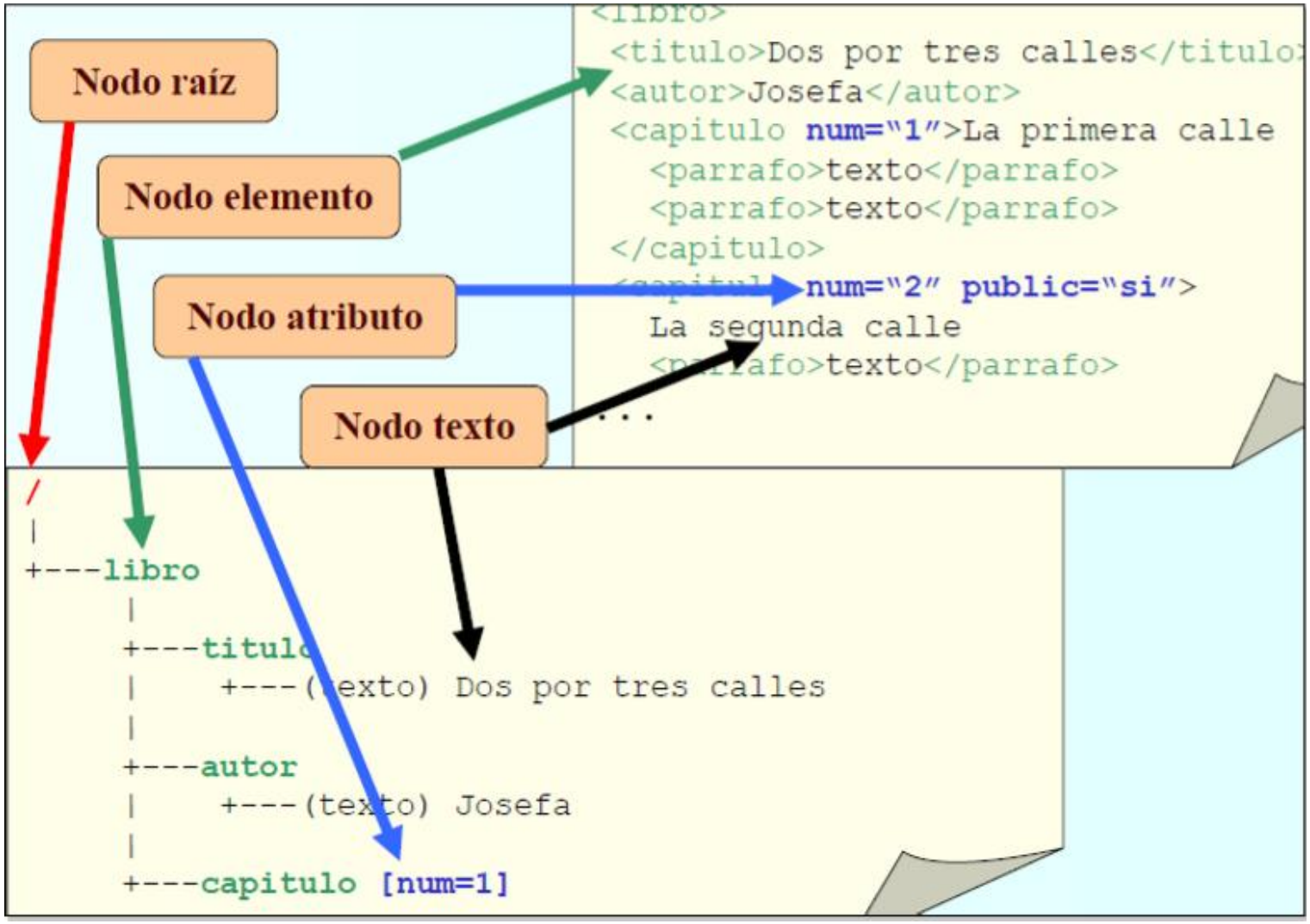
Tipos de nodos (1)

- **Nodo Raíz**
  - Se identifica por / . No se debe confundir el nodo raíz con el elemento raíz del documento.
    - El documento XML de nuestro ejemplo tiene por elemento raíz libro, éste será el primer nodo que cuelgue del nodo raíz del árbol, el cual es: /.
- **Nodo Elemento**
  - Cualquier elemento de un documento XML se convierte en un nodo elemento dentro del árbol.
  - Cada elemento tiene su nodo padre. El nodo padre de cualquier elemento es, a su vez, un elemento, excepto el elemento raíz, cuyo padre es el nodo raíz.
  - Los nodos elemento tienen a su vez hijos, que puede ser nodos de cualquier tipo excepto raíz.

Tipos de nodos (2)

- **Nodos Texto**
  - referencia a todos los caracteres del documento que no está marcados con alguna etiqueta.
- **Nodo Atributo**
  - Más que hijos del nodo elemento que los contiene son como etiquetas añadidas a dicho nodo.
  - Cada nodo atributo consta de un nombre, un valor (que es siempre una cadena).
- **Nodos comentario, y de Instrucciones de proceso.**
  - Al contenido de estos nodos se puede acceder con la propiedad string-value

Tipos de nodo (3)



Expresiones XPath (1)

- Una "instrucción" en lenguaje XPath se denomina **expresión**.
- Hay que considerar una expresión XPath como un “predicado”, que devuelve todo lo que encaja con dicho predicado.
- Lo que devuelve es procesado por la regla XSL.
- Las expresiones XPath se usan sobre todo en los atributos match, select (y test).

Expresiones XPath (2)

- Una expresión XPath arroja (tras ser evaluada) una expresión que puede ser de 4 tipos posibles: **conjunto de nodos** (node-set), **booleano**, **número**, ó **cadena.**
- Tokens válidos en una expresión Xpath
  - Paréntesis y similares: ( ) { } [ ]
  - Elemento actual . y elemento padre ..
  - Atributo @, * y separador ::
  - La coma ,
  - El nombre de un elemento
  - Tipo de nodo (comment, text, processing instruction, node)
  - Operadores: and, or, mod, div, *, /, //, |, +, -, =, !=, <, <=, >, >=
  - Nombres de función
  - Nombre de eje (axis): ancestor, ancestor-or-self, attribute, child, descendant, descendant-orself, following, following-sibling, namespace, parent, preceding, preceding-sibling, self
  - Literales, entre comillas dobles o simples (se pueden anidar alternadas)
  - Números
  - Referencias a variables ($nombreVariable)
- Nos centraremos en los que veamos en los ejemplos y ejercicios siguientes.

Location Paths

- Un **location path** es la más importante de los tipos de expresiones que se pueden especificar en notación XPath.

- La sintaxis de un location path es similiar a la usada a la hora de describir los directorios que forman una unidad de disco en Unix.

Location Paths. Sintaxis (1).

Una ruta de directorio de Linux indica la ruta a un archivo particular:

- Ejemplo: /home/juan/documentos
  - Referencia a un único directorio llamado documentos que cuelga de /home/juan

- Location Paths se corresponde con la idea intuitiva de “ruta de directorio”, pero un location path siempre devuelve un node-set:
  - XPath indica la ruta hasta varios nodos, basándose en la estructura del documento XML
  - Ejemplo: /libro/capitulo/parrafo
    - Referencia a todos los elementos parrafo que cuelguen de cualquier capitulo del libro

```
<?xml version="1.0" encoding="UTF-8"?>
<libro>
    <titulo>Dos por tres calles </titulo>
    <autor>Josefa Santos</autor>
    <capitulo num="1">
        La primera calle
        <parrafo>
            Era una sombria noche del mes de agosto.....
        </parrafo>
        <parrafo destacar="si">
            Ella, cual inocente
            <enlace href="http://www.enlace.css">mariposa</enlace>
            que surca el cielo en busca de .....
        </parrafo>
    </capitulo>
    <capitulo num="2">
        La segunda calle
        <parrafo>
            Era una oscura noche del mes de octubre.....
        </parrafo>
        <parrafo >
            Ella, cual inocente abeja ..
        </parrafo>
    </capitulo>
</libro>
```

Location Paths. Sintaxis. (2)

- Si se indica el camino completo, la búsqueda comienza en el nodo raíz.
- Si se indica un camino completo, se entiende que el path comienza en el nodo que en cada momento se está procesando.
- Ejemplo:
  - /libro/capitulo/parrafo
    - Al leer / se selecciona el nodo raíz como nodo contexto.
    - Al leer **libro** se seleccionan los elementos libro que cuelgan del contexto (/)
    - Al leer **capitulo** se seleccionan los elementos capitulo que cuelgan del contexto (en ese momento es **libro**)
    - Al leer **parrafo** se seleccionan los elementos parrafo que cuelgan del contexto (en ese momento es **capitulo**)

Location Paths. Nodo Contexto (1)

- Un location path siempre tiene un nodo contexto
  - Es similar al concepto de directorio actual:
    - ls ./juan/documentos
- En XPath, si la expresión comienza por / estamos dando un path absoluto, partiendo del nodo raíz.
- Si no comienza por /, estamos dando un camino relativo desde el nodo actual (nodo contexto).

Location Paths. Nodo Contexto (2)

- Nodo actual (**current node**)
  - Es un nodo que está seleccionado cuando se va a evaluar una expresión XPath
  - Constituye el punto de partida al evaluar la expresión
- Nodo **contexto** (context node)
  - Para evaluar una expresión, se van evaluando subexpresion parciales
  - Cada vez que se evalúa una subexpresión se obtiene un nuevo conjunto de nodos (node-set) que es el nuevo contexto para evaluar la siguiente subexpresión
- Tamaño del contexto (context size)
  - El número de nodos que se están evaluando en un momento dado. Luego el Nodo contexto puede ser en si un conjunto de nodos, para los que se evaluaría una expresión empezando por uno de ellos, que será el Current node en el momento de evaluar la expresión.

Location Paths. Nodo Contexto (3)

- **Ejemplo con /libro/capitulo/parrafo**
- El analizador comienza por leer /, lo cual le dice que debe seleccionar el nodo raíz, independientemente del nodo contexto que en ese momento exista. En el momento en que el evaluador de XPath localiza el nodo raíz, éste pasa a ser el nodo contexto de dicha expresión.
- El analizador leería ahora **libro**, lo cual le dice que seleccione TODOS los elementos que cuelgan del nodo contexto (que ahora mismo es el nodo raíz) y que se llamen libro. En este caso solo hay uno... porque solo puede haber un elemento raíz. El nodo contexto pasa a ser libro.
- A continuación el analizador leería capitulo, entonces selecciona TODOS los elementos que cuelgan del nodo contexto (ahora es el nodo libro).
  - En un disco sería imposible que hubiera dos directorios con el mismo nombre colgando de un mismo directorio padre.
  - En estos momentos hay dos elementos que encajan con el patrón /libro/capitulo.
- El analizador continua leyendo **parrafo**. Con lo que selecciona TODOS los elementos parrafo que cuelgan del nodo contexto...¡¡pero NO hay un nodo contexto, sino DOS!!. El evaluador de expresiones lo va a recorrer uno por uno haciendo que, mientras evalúa un determinado nodo, ése sea el nodo contexto de ese momento.

Location Paths.

Un location path consta de:

- Eje (axis). Es la relación entre el nodo de contexto y el Location Path
  - El eje a veces está implícito (no se pone, y entonces se sobreentendera “es hijo de”).
  - Para nosotros esto será lo común.
- Prueba de nodo (node test). Es el “nombre de directorio”
- Predicado (predicate). Expresión XPath entre corchetes.
  - El predicado es opcional

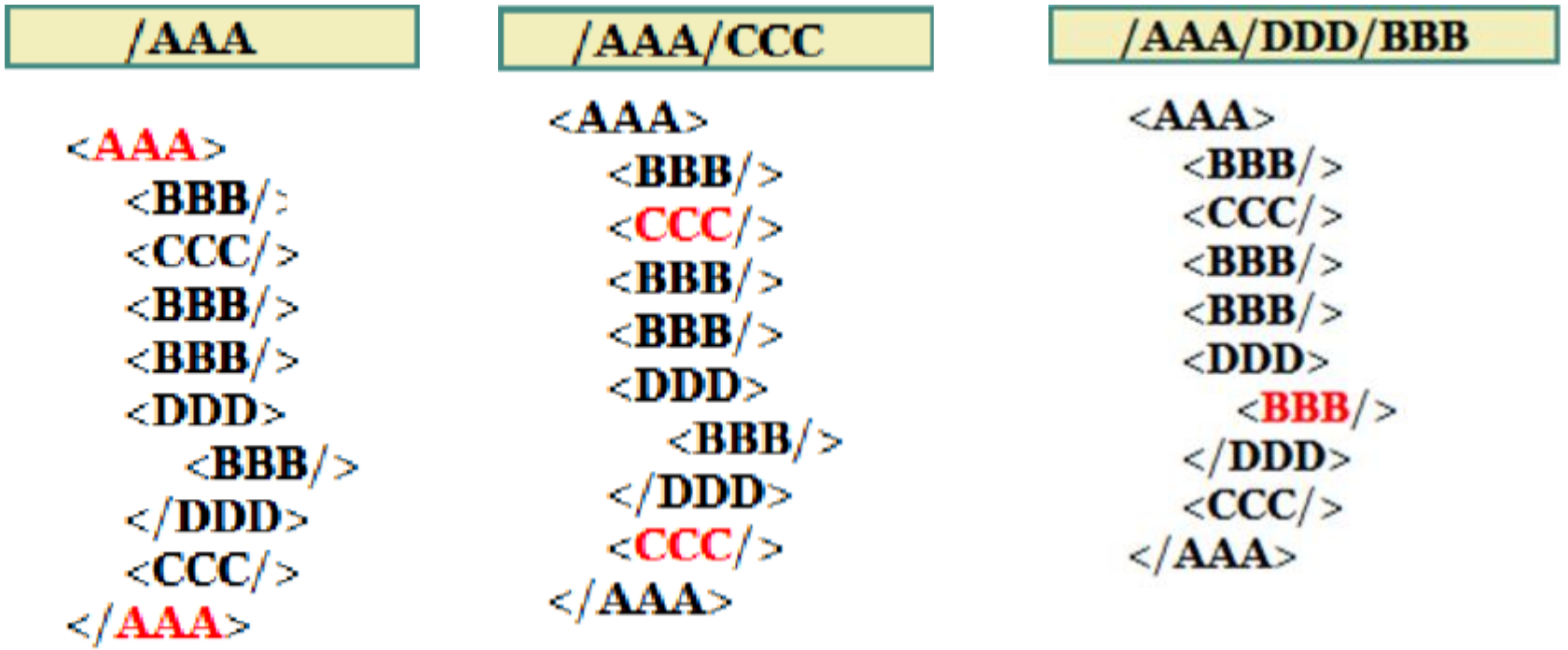
    eje::pruebanodo[predicado]

Location Paths: Prueba de nodo (1)

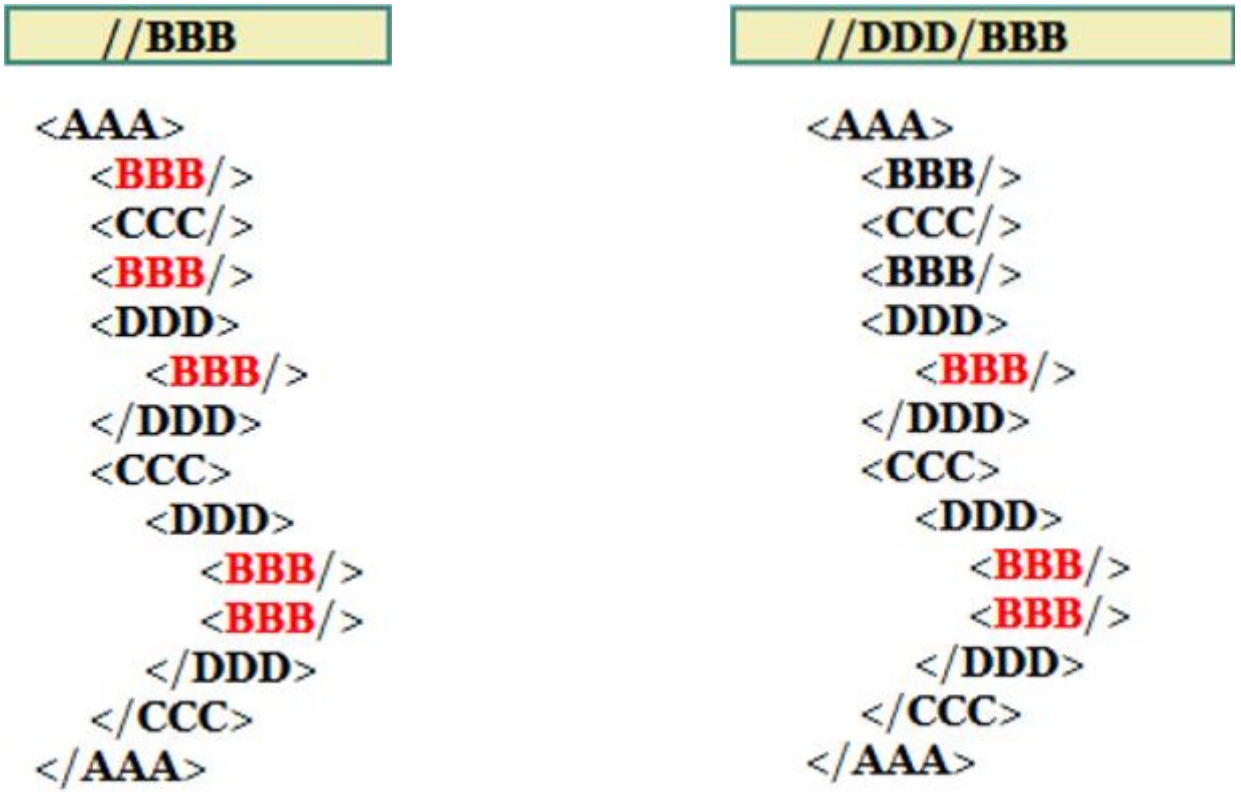
eje::**pruebanodo**[predicado]

- La forma más simple es escribir simplemente el nombre del nodo (su etiqueta)
- * que simboliza cualquier nombre
- Ejemplos:
  - /libro/capitulo/parrafo
    - Encaja con cualquier nodo “parrafo” que sea hijo de un nodo “capitulo” que sea hijo de un nodo “libro” que será el nodo raíz
  - /libro/*
    - * simboliza cualquier nombre: encaja con cualquier nodo que sea hijo del nodo “libro” que será el hijo del nodo raiz.
  - capitulo/*
    - Encaja con cualquier nodo que sea hijo de un nodo “capitulo” que sea hijo del nodo de contexto
- IMPORTANTE: // indica “que sea hijo de cualquiera”
  - Los atributos se especifican con el prefijo @. Su acceso ser vera más adelante.

Location Paths: Prueba de nodo (2)

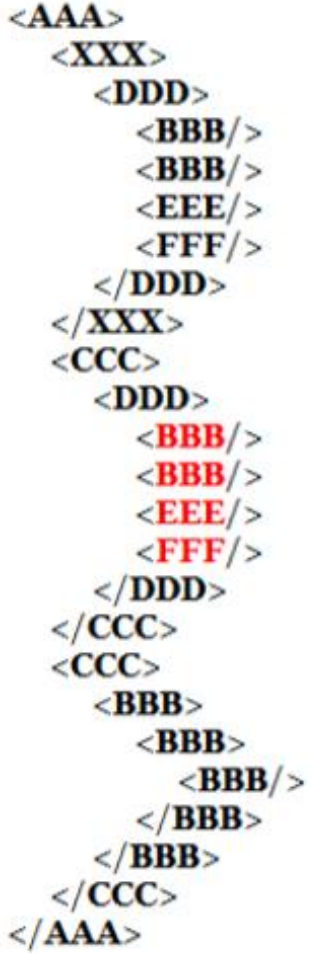


Location Paths: Prueba de nodo (3)

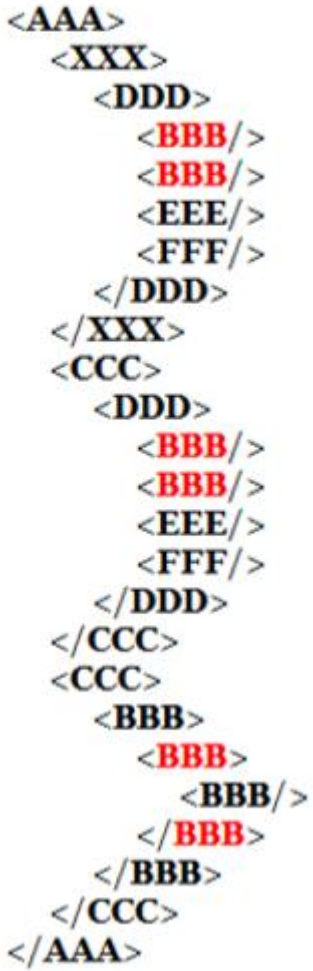


Location Paths: Prueba de nodo (4)

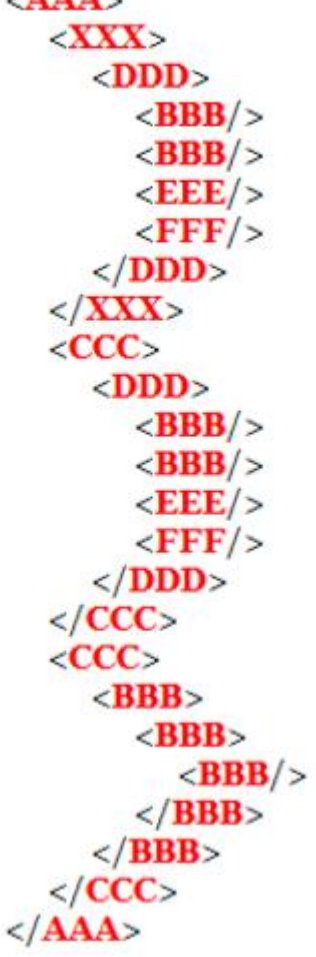
/AAA/CCC/DDD/*



/*/*/BBB



//*



Location Paths: Prueba de nodo (5)

*

- Devuelve todos los nodos de tipo principal (es decir, elemento, atributo o espacio de nombres), pero no nodos de texto, comentarios y de instrucciones de proceso.
- Ejemplo: Seleccionar todos los nodos principales descendientes de los parrafo:
  - //parrafo/*

Location Paths: Prueba de nodo (6)

node()

- El predicado node() devuelve todos los nodos de todos los tipos.

- Ejemplo: Seleccionar todos los nodos descendientes de los **parrafo**:
  - //parrafo/node()

Location Paths: Prueba de nodo (7)

text()

- Selecciona cualquier nodo de tipo texto.
- Ejemplo:
  - Seleccionar el texto de todos los nodos parrafo:
    - //parrafo/text()
  - Seleccionar TODO el texto que cuelga de todos los nodos parrafo:
    - //parrafo//text()

Location Path: Predicado (1)

eje::pruebanodo[predicado]

- Un predicado permite restringir el conjunto de nodos seleccionados a aquellos que cumplen cierta condición.
  - Dicha condición es una expresión XPath y se especifica entre corchetes.
  - Es una expresión booleana: un nodo del conjunto seleccionado la cumplirá, o no la cumplirá.
- Dada la prueba de nodo, y dado el eje (supongamos “es hijo de”), del conjunto de nodos resultante quedan sólo los que cumplan el predicado
- En el predicado pueden intervenir **funciones Xpath**.

Location Path: Predicado (2)

- Ejemplo: Seleccionar todos los **capitulo** que tengan un **parrafo** que tenga algún elemento con atributo href:
  - //capitulo[parrafo/*[@href]]
- Serán muchas las ocasiones en que deseemos que los nodos a seleccionar no cumplan una sino varias condiciones. En estos casos, los predicados se pueden suceder uno a otro haciendo el efecto de la operación AND. Como en el siguiente ejemplo.
- Ejemplo: Seleccionar todos los **capitulo** que tengan un **parrafo** que tenga algún elemento con atributo **href** y que ellos mismos (los **capitulo**) tengan el atributo **public** a valor **si**:
  - //capitulo [parrafo/*[@href]] [@public='si']

Location Path: Predicado (3)

- También se puede hacer uso del operador and encerrando entre paréntesis los distintos predicados lógicos. Ejemplo similar al anterior:
  - //capitulo[(parrafo/*[@href]) and (@public='si')]
  - También se puede hacer uso de la operación **OR** en vez de and.
- Existe otro tipo de operación **or** que utiliza la barra vertical: | separando no dos predicados, sino dos expresiones XPath.
- Ejemplo de |: Seleccionar todos los **capitulo** que tengan un **parrafo** que tenga algún elemento con atributo **href** o todos los **apendice**:
  - //capitulo[parrafo/*[@href]] | //apendice

Location Path: Predicado (4)

- Por último, también podemos especificar con **not** la negación de alguna de las negaciones del predicado.
- Ejemplo: Seleccionar todos los **capitulo** que no tengan el atributo **public**: //capitulo[not(@public)]
- **Predicados con funciones de cardinalidad**
  - Existen funciones que nos van a servir para restringir los nodos basándose en la posición del elemento devuelto.
    - **position**() → Ejemplo: //capitulo[position()=2] ó //capitulo[2]
    - **last**() → Ejemplo: //capitulo[last()] ó //capitulo[not(position()=last())] ó //capitulo[last()-1]
    - **id**(). → Ejemplo: id("capitulo_1")/parrafo

Más funciones Xpath (1)

- Hay una gran variedad de **funciones** que podemos usar en el predicado:
  - **boolean**(): convierte a booleano. Aplicada a un conjunto de nodos, devuelve true si no es vacío. not(), true()
  - **count**(): Devuelve el número de nodos en un conjunto de nodos.
  - **name**(): Devuelve el nombre de un nodo (su etiqueta). local-name(), namespace-uri()
  - Biblioteca de strings:
    - **normalize-space**(), **string**(), **concat**(), **stringlength**()
  - **sum**()
  - Recuerda también las de **cardinalidad**: id(), last(), position()

Más funciones Xpath (2)

//*[count(BBB)=2]

Selecciona elementos que tienen dos hijos BBB

```
<AAA>
  <CCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </CCC>
  <DDD>
    <BBB/>
    <BBB/>
  </DDD>
  <EEE>
    <CCC/>
    <DDD/>
  </EEE>
</AAA>
```

//*[count(*)=2]

selecciona elementos con dos hijos

```
<AAA>
  <CCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </CCC>
  <DDD>
    <BBB/>
    <BBB/>
  </DDD>
  <EEE>
    <CCC/>
    <DDD/>
  </EEE>
</AAA>
```

//*[count(*)=3]

selecciona elementos con tres hijos

```
<AAA>
  <CCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </CCC>
  <DDD>
    <BBB/>
    <BBB/>
  </DDD>
  <EEE>
    <CCC/>
    <DDD/>
  </EEE>
</AAA>
```

Más funciones Xpath (3)

//*[name()='BBB']

selecciona todos los elementos con nombre BBB, equivalente a //BBB

```
<AAA>
  <BCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </BCC>
  <DDB>
    <BBB/>
    <BBB/>
  </DDB>
  <BEC>
    <CCC/>
    <DBD/>
  </BEC>
</AAA>
```

//*[starts-with(name(),'B')]

selecciona todos los elementos cuyo nombre empieza por B

```
<AAA>
  <BCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </BCC>
  <DDB>
    <BBB/>
    <BBB/>
  </DDB>
  <BEC>
    <CCC/>
    <DBD/>
  </BEC>
</AAA>
```

//*[contains(name(),'C')]

selecciona todos los elementos cuyo nombre contenga la letra C

```
<AAA>
  <BCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </BCC>
  <DDB>
    <BBB/>
    <BBB/>
  </DDB>
  <BEC>
    <CCC/>
    <DBD/>
  </BEC>
</AAA>
```

Más funciones Xpath (4)

//*[string-length(name()) = 3]

Selecciona elementos con tres letras en el nombre

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

//*[string-length(name()) < 3]

selecciona elementos que tienen en el nombre uno o dos caracteres

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

//*[string-length(name()) > 3]

selecciona elementos con un nombre mayor de tres caracteres

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

Localization Paths. Ejes (1)

eje::pruebanodo[predicado]

- El eje denota la relación de un Localization Paths con su nodo de contexto
- Hay una serie de ejes posibles: ancestor, ancestor-or-self, attribute, child, descendant, descendant-or-self, following, following-sibling, namespace, parent, preceding, preceding-sibling, self.
- Equivale a “que es un”, pero sus argumentos se leen de derecha a izquierda
- **child está implícito** y casi nunca se pone.
  - Pero para el nodo raíz, está implícito self (self denota al nodo de contexto)
- Ejemplos:
  - /universidad/nombre
    - Equivale de manera implícita a /self::universidad/child::nombre
  - /universidad/nombre/following-sibling::*
    - Todos los nodos que son “hermanos después de” nombre (en el orden del documento) que es hijo de universidad

Localization Paths. Ejes (2)

Child

- Es el hacha utilizada por defecto. Se corresponde con la barra, / (aunque tiene una forma más larga que es: /child::).
- Ejemplo: Seleccionar todos los titulo de un libro:
  - /libro/titulo
- Sugerencia: Seleccionar el autor del libro.
- Sugerencia: Seleccionar todos los párrafos del libro.

Localization Paths. Ejes (3)

Attribute

- Se corresponde con el signo de la arroba, @ (o en su forma larga que es: attribute::). Mediante este operador podemos seleccionar aquellos nodos atributos que deseemos, indicando el nombre del atributo en cuestión.
- Ejemplo: Seleccionar el atributo **num** que posean los elementos **capitulo**
  - /libro/capitulo/@num
- Ejemplo: Seleccionar todos los elementos hijo de los **capitulo** que posean el atributo **public** (sin importar el valor asignado al mismo):
  - /libro/capitulo[@public]/*
- Ejemplo: Seleccionar todos los elementos hijo de **parrafo** cuyo atributo **destacar** sea igual a "si".
  - /libro/titulo/parrafo[@destacar="si"]

Attribute

//@id

selecciona todos los atributos @id

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

//BBB[@*]

selecciona todos los elementos que tengan atributos

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

//BBB[@name]

Selecciona elementos BBB con atributo name

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

//BBB[@id]

selecciona los elementos BBB con atributo id

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

//BBB[not(@*)]

selecciona elementos BBB que no tienen atributos

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

Localization Paths. Ejes (4)

Descendant

- Se especifica poniendo una doble barra: // (en su forma larga: descendant::).
- Sirve para seleccionar TODOS los nodos que desciendan del conjunto de nodos contexto. Es decir, no solo los hijos de los nodos contexto, sino también los hijos de los hijos, y los hijos de estos, etc.
- Ejemplo: Seleccionar todos los **parrafo** de un **libro**:
  - /libro//parrafo
- Ejemplo: Seleccionar todos los descendientes de **parrafo** que tienen un atributo **href**.
  - //parrafo//*[@href]
- Ejemplo: Mostrar el valor del atributo **href** del caso anterior:.
  - //parrafo//*[@href]/@href

Localization Paths. Ejes (5)

Self

- Se especifica mediante el punto (.).
- Es muy útil pues sirve para seleccionar el nodo contexto.
- Para seleccionar todos los parrafo descendientes del nodo contexto no podemos escribir //parrafo, dado que seleccionaría todos los descendientes del nodo raíz. Por ello, la forma correcta es: .//parrafo

Parent

- Al igual que en los sistemas de ficheros, se utilizan los dos puntos para identificarlo: ..
- El comportamiento de este eje es un poco extraño al principio dado que realiza un paso hacia atrás en el árbol de nodos.
- Ejemplo: Seleccionar todos los nodos que tienen algún hijo de tipo **parrafo**:
  - //parrafo/..
- Ejemplo: Seleccionar todos los nodos **capitulo** que tienen algún hijo de tipo **parrafo**:
  - //parrafo/../../capitulo
  - O bien: //capitulo/parrafo/..

Localization Paths. Ejes (6)

Ancestor

- De todas los ejes que podemos usar, esta es la única que no tiene ninguna forma de abreviación. Siempre aparece como ancestor::
- Ancestor es a parent lo que descendant es a child. Es decir, devuelve todos los elementos de los cuales el nodo contexto es descendiente.
- Ejemplo: Seleccionar todos los elementos que tienen entre sus descendientes algún **parrafo**
  - //parrafo/ancestor::*