Unidad de Trabajo 5:
XPATH

XPath

- Dadas las limitaciones de CSS, para trabajar en XML se recomienda XSL (Extensible Stylesheet Language – lenguaje extensible de hojas de estilo).
- XSL es una familia de lenguajes basados en XML que permiten definir la transformación y presentación de documentos XML.
- Está formada por tres lenguajes:
- **XPath**
  Lenguaje que permite construir expresiones que recorren y procesan un documento XML.
  Versión 1.0: http://www.w3.org/TR/xpath/
  Versión 2.0: http://www.w3.org/TR/xpath20/
  Versión 3.0: http://www.w3.org/TR/xpath-3/
- **XSLT** http://www.w3.org/TR/xslt
  Lenguaje que permite transformar documentos XML en otros.
  Versión 1.0: http://www.w3.org/TR/xslt
  Versión 2.0: http://www.w3.org/TR/xslt20/
- **XSL-FO** http://www.w3.org/TR/xsl11/
  Lenguaje que permite especificar cómo se van a formatear los datos para presentarlos en pantalla, en papel o en otros medios.

XPath

- XPath es un lenguaje, distinto de XML, para identificar determinadas partes de los documentos XML.
- XPath señala los nodos por posición, posición relativa, tipo, contenido y por otros criterios.
- XSLT usa expresiones XPath para comparar y seleccionar determinados elementos en el documento de entrada y copiarlos en el documento de salida o para un procesamiento posterior.
- XPath considera cada documento XML como un árbol de nodos.
- XPath es un lenguaje para elegir nodos y conjuntos de nodos de este árbol.
- Desde el punto de vista de XPath, existen siete tipos de nodos:
  - Nodo raíz (root node).
  - Nodo de elemento (element node).
  - Nodo de atributo (attribute node).
  - Nodo de texto (text node).
  - Nodo de espacio de nombre (namespace node).
  - Nodo de Instrucción de procesamiento (processing instruction node).
  - Nodo de comentario (comment node).

Xpath: Tipos de nodos

Raíz

- Cada documento tiene exactamente un nodo raíz (root), que es la raíz del árbol.
- Este nodo contiene un *hijo del tipo nodo de comentario* por cada comentario que se encuentre fuera del elemento raíz del documento.
- Contiene un hijo del *tipo nodo de instrucción de procesamiento* para cada instrucción de procesamiento externa al elemento raíz.
- Contiene un *hijo del tipo nodo de elemento* para el elemento raíz del documento.
- El nodo raíz no tiene nodo padre.
- El valor del nodo raíz es el valor del elemento raíz.

Elemento

- *Un nodo de elemento (element)* tiene un nombre, un URI de espacio de nombre, un nodo padre y una lista de nodos hijos que puede incluir otros nodos de elemento, nodos de comentario, nodos de instrucción de procesamiento y nodos de texto.
- Un nodo de elemento también tiene una colección de atributos y una colección de espacios de nombre. No se consideran hijos.
- El valor de un nodo de elemento es el texto incluido entre las etiquetas de inicio y de cierre que permanezca después de haberse eliminado todas las etiquetas, comentarios e instrucciones de procesamiento.

Xpath: tipos de nodos

Atributo

- Un nodo de *atributo (attribute*) tiene un nombre, un URI de espacio de nombre, un valor y un elemento padre.
- Los elementos son padres del atributo pero los atributos no son hijos de sus elementos padres.
- Un nodo de atributo tiene un nodo padre pero no tiene hijos.
- Los atributos **xmlns** no se representan como nodos de atributo.
- El valor de un nodo de atributo es el valor del atributo.

Texto

- Cada nodo de *texto (text)* representa el texto entre etiquetas, instrucciones de procesamiento y comentarios.
- Un nodo de texto tiene un nodo padre pero no tiene hijos.
- El valor de un nodo de texto es el texto del nodo.

Espacio de nombre

- Un nodo de *espacio de nombre (namespace)* representa un espacio de nombre en el ámbito de un elemento.
- Cada declaración de espacio de nombre mediante un atributo **xmlns** produce un nodo de espacio de nombre en dicho elemento y en todos sus descendientes.
- Cada nodo de espacio de nombre tiene un elemento padre, pero no es hijo.
- El nombre de un nodo de espacio de nombre es el prefijo.

Xpath: tipos de nodos

Instrucción de procesamiento

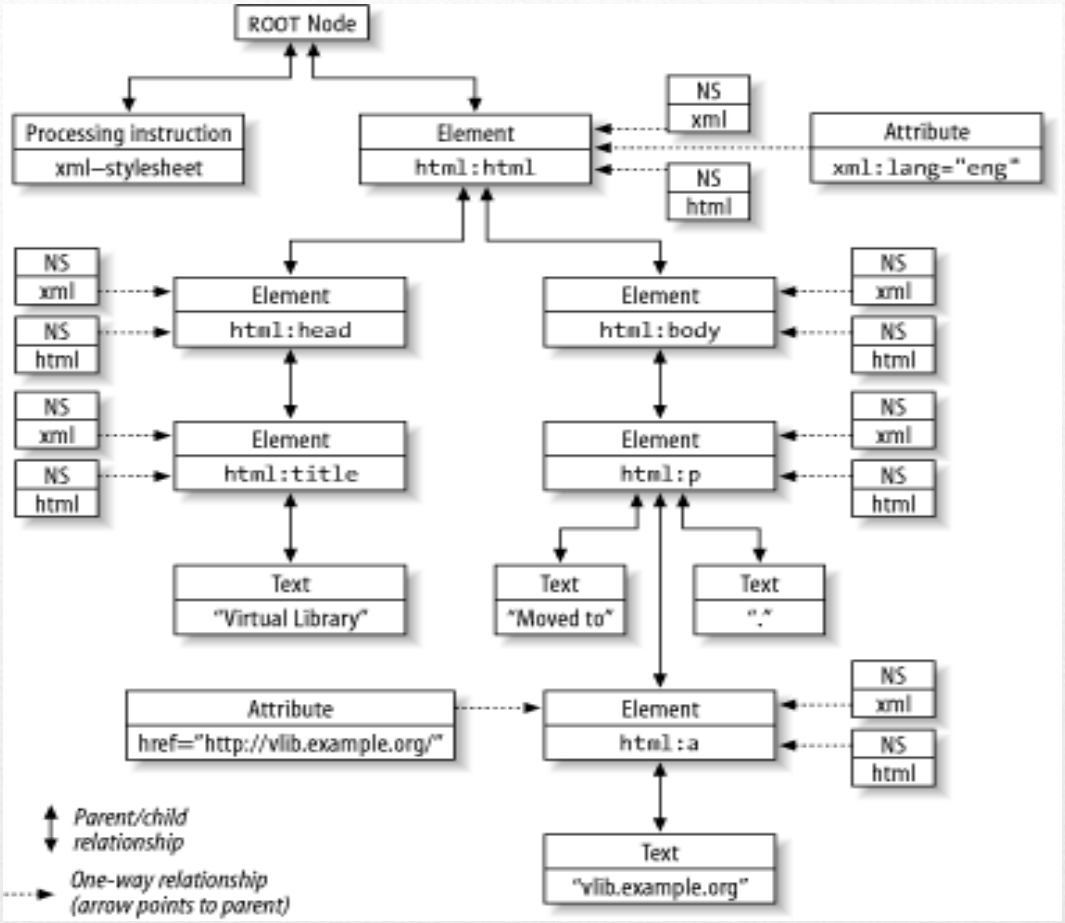
- Un nodo de instrucción de procesamiento (processing-instruction) tiene un destino, datos, un nodo padre y ningún hijo.
- El nombre del nodo de instrucción de procesamiento es su destino.
- El valor del nodo de instrucción de procesamiento son los datos de la instrucción de procesamiento.

Comentario

- Un nodo de comentario (comment) representa un comentario.
- Tiene un nodo padre y no tiene hijos.
- El valor de un comentario es el contenido del mismo, sin incluir <!-- y -->.

La declaración XML y la declaración de tipo de documento no se incluyen en la vista XPath de un documento XML.

Todas las referencias de entidad, referencias de carácter y secciones CDATA se resuelven antes de que se haya creado el árbol XPath.




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

<html:html
xmlns:html='http://www.w3.org/1999/xhtml'
xml:lang="eng">

  <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>
```

Expresiones XPath

- **Las expresiones XPath se construyen a través de palabras claves, símbolos y operadores.**
- **Distingue entre mayúsculas y minúsculas.**
- **Existen los siguientes tipos de expresiones:** literales, llamadas a funciones, node-sets, booleanas, numéricas y cadenas.
- **Las expresiones son evaluadas y pueden devolver 4 tipos distintos de resultados: booleanos, numéricos, cadenas o un conjunto de nodos.**
- **Las expresiones se evaluaran respecto a un contexto, es decir, la misma sentencia XPath ejecutada en contextos distintos devolverá resultados distintos.**
- **El contexto viene definido principalmente por: el nodo de contexto, el número de elementos hermanos del nodo de contexto y el número que indica la posición del nodo de contexto dentro de sus hermanos.**
- **El tipo de expresión más importante en XPath son las rutas de localización (location paths), que podrán ser absolutas o relativas.**

Valores que devuelve una expresión Xpath

XPath Expression Evaluation

/universidad/carreras/carrera/@id="c01"

Valores booleanos

Type	Name	Value
Bool		true

XPath Expression Evaluation

count(//carreras/carrera)

Número

Type	Name	Value
Num		6.000000

XPath Expression Evaluation

/universidad/carreras/carrera

Conjunto de nodos

Type	Name	Value
Node	carrera	id="c01"
Node	carrera	id="c02"
Node	carrera	id="c03"
Node	carrera	id="c04"
Node	carrera	id="c05"
Node	carrera	id="c06"

XPath Expression Evaluation

/universidad/carreras/carrera/nombre/text()

text

Type	Name	Value
Text		I.T. InformÃ¡tica
Text		Dipl. Empresariales
Text		Dipl. Relaciones Laborales
Text		Lic. Quimica
Text		Lic. BiologÃ-a
Text		Lic. Humanidades

Rutas de localización (path)

- Las expresiones XPath se construyen a través de palabras claves, símbolos y operadores.
- Distingue entre mayúsculas y minúsculas.
- Existen los siguientes tipos de expresiones: literales, llamadas a funciones, node-sets, booleanas, numéricas y cadenas.
- Las expresiones son evaluadas y pueden devolver 4 tipos distintos de resultados: booleanos, numéricos, cadenas o un conjunto de nodos.
- Las expresiones se evaluaran respecto a un contexto, es decir, la misma sentencia XPath ejecutada en contextos distintos devolverá resultados distintos.
- El contexto viene definido principalmente por: el nodo de contexto, el número de elementos hermanos del nodo de contexto y el número que indica la posición del nodo de contexto dentro de sus hermanos.
- El tipo de expresión más importante en XPath son las rutas de localización (location paths), que podrán ser absolutas o relativas.
- Formato de un paso de localización: **nombreEje::nodoComprobación[predicado(s)]**

Rutas de localización (path)

- Los ejes son los elementos encargados de especificar en el paso de localización la relación existente dentro del árbol XPath entre el nodo contexto de donde se parte y los distintos nodos que se quieren seleccionar.
- Los nodos de comprobación o búsqueda (node test) son los encargados de identificar un nodo o nodos concretos dentro de un eje; su función dentro del paso de localización es que cada eje pueda determinar un tipo de nodo (llamado principal) a la hora de efectuar la selección.
- Los predicados son elementos que en un paso de localización permiten restringir, dentro del conjunto de nodos seleccionados por un eje, a aquellos nodos que cumplen una cierta condición debidamente especificada. Permite pues filtrar un conjunto de datos. Estos predicados se declaran entre corchetes “[]”.

Rutas de localización

- Ruta de localización de la raíz
  - La ruta de localización más simple es la que selecciona el nodo raíz del documento.
  - Se trata de la barra diagonal (/).

- Pasos de localización de elementos hijos.
  - La segunda ruta de localización más simple es un solo nombre de elemento.
  - Esta ruta selecciona todos los elementos hijos del nodo de contexto con el nombre especificado.

- Pasos de localización de atributos.
  - Para seleccionar un determinado atributo de un elemento, usamos el signo @ seguido por el nombre del atributo deseado.

- Pasos de localización de comment(), text() y processing-instruction().

- **Comodines**
  - Los comodines comparan distintos elementos y tipos de nodo simultáneamente.
  - Los caracteres comodín son tres: *, node() y @*

Rutas de localización

Comodines

- El carácter comodín * compara cualquier nodo de elemento, independientemente de su nombre.
- El carácter comodín **node()** compara no sólo todos los tipos de elementos sino también el nodo raíz, los nodos de texto, los nodos de instrucción de procesamiento, los nodos de espacios de nombre, los nodos de atributo y los nodos de comentarios.
- El carácter comodín @* compara todos los nodos de atributo.

- **Mú**ltiples coincidencias con |
  - Para indicar que deseamos comparar cualquiera de los elementos indicados, podemos combinar rutas y pasos de localización con la barra vertical (|).

- Rutas de localización compuestas
  - Podemos combinar los pasos de localización individuales con la barra diagonal (/) para movernos por la jerarquía desde el nodo coincidente a otros nodos.
  - Podemos usar un punto (.) para hacer referencia al nodo de contexto.
  - Podemos usar un doble punto (..) para hacer referencia al nodo padre.
  - Podemos usar una barra diagonal doble (//) para hacer referencia a los descendientes del nodo de contexto.

Rutas de localización

- **Crear rutas de localización compuestas a partir de los pasos de localización con /**
  - Los pasos de localización se pueden combinar con una barra diagonal para crear una ruta de localización compuesta.
  - Cada paso de la ruta es relativo al que le precede.
  - Si la ruta empieza con /, el primer paso será relativo al nodo raíz.
  - En caso contrario, será relativo al nodo de contexto**.**
- **Seleccionar a partir de los descendientes con //**
  - Una doble barra diagonal selecciona a partir de todos los descendientes del nodo de contexto, así como el propio nodo de contexto.
  - Al inicio de una expresión XPath, selecciona a partir de todos los nodos en el documento.
- **Seleccionar el elemento padre con ..**
  - Un doble punto indica el padre del nodo actual.
- **Seleccionar el nodo de contexto con .**
  - Un solo punto indica el nodo de contexto.

Ejes

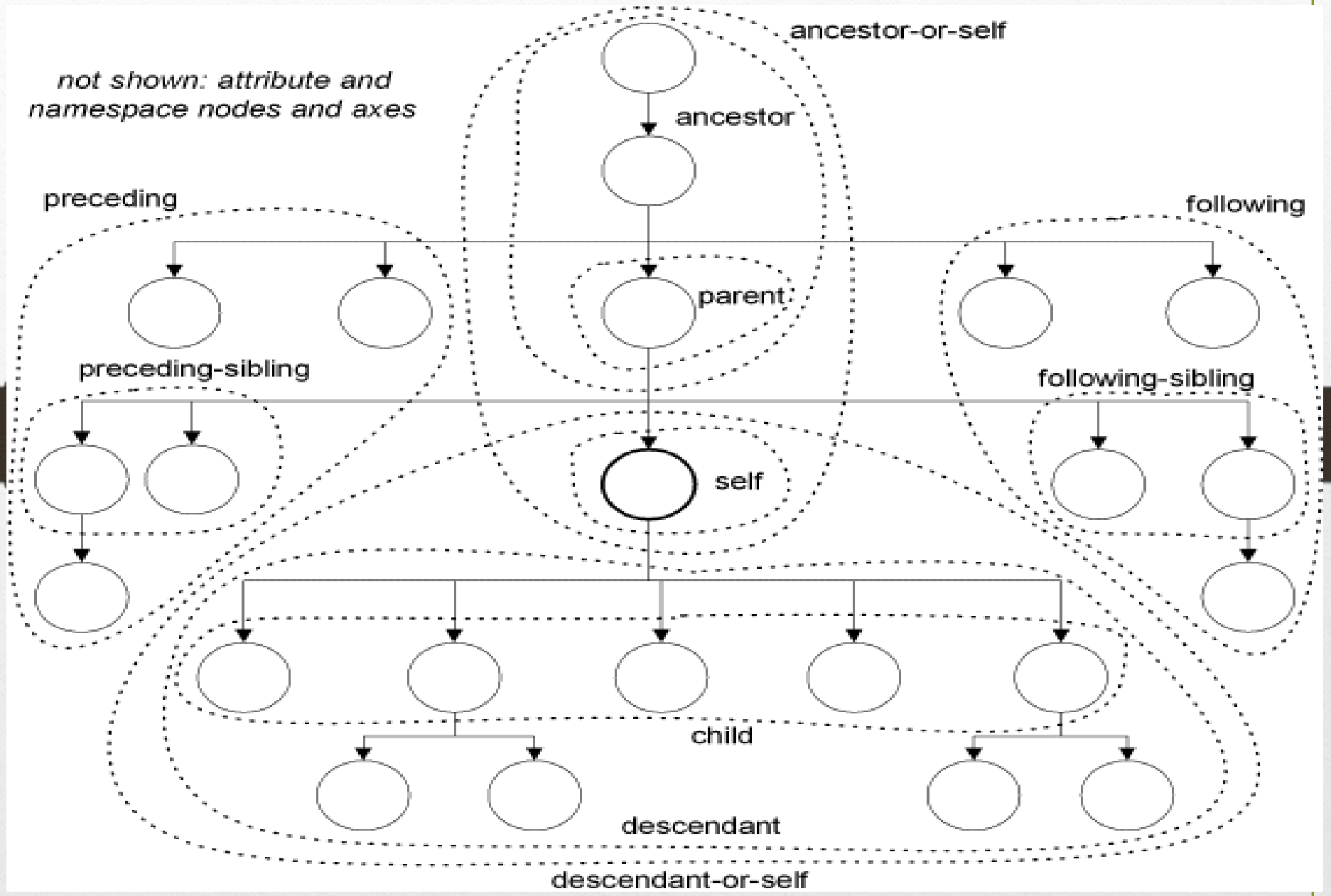
- **Cada paso de localización de desplaza por un eje desde el nodo de contexto.**
- **Hay 13 ejes, divididos en dos grupos, que podemos usar en un paso de localización XPath.**
- **Ejes con dirección hacia adelante (forward axis):**
- **child**
  - Todos los hijos del nodo de contexto.
  - Sólo los nodos de raíz y de elemento tienen hijos.
  - Los nodos de atributo y de espacio de nombre no son hijos de ningún nodo, aunque tienen nodos padres.
- **descendant**
  - Todos los nodos contenidos dentro del nodo de contexto.
  - Sólo los nodos de raíz y de elemento tienen descendientes.
- **attribute**
  - Todos los atributos del nodo de contexto.
  - Abreviatura de ::nombre_atributo @nombre_atributo

Ejes

- **Ejes con dirección hacia adelante (forward axis):**
- **self**
  - El propio nodo de contexto.
  - Abreviatura de self::node() .
- **descendant-or-self**
  - Cualquier descendiente del nodo de contexto o el nodo de contexto.
  - Abreviatura de descendant-or-self::node()/ //
- **following-sibling**
  - Todos los nodos que son hijos del nodo padre y que siguen en orden al nodo de contexto.
- **following**
  - **T**odos los nodos descendientes del nodo raíz que se encuentran después de la posición del nodo de contexto, excepto nodos de atributo y de espacio de nombre, y no son descendientes del mismo.
- **namespace**
  - **T**odos los espacios de nombre en el ámbito del nodo de contexto.

Ejes

- **Ejes con dirección hacia atrás (reverse axis):**
- **parent**
  - El elemento o nodo de raíz que inmediatamente contiene al nodo de contexto.
  - Sólo el nodo de raíx no tiene un nodo padre.
  - Abreviatura de **parent::node()** **..**
- **ancestor**
  - El nodo de raíz y todos los nodos de elemento que contienen el nodo de contexto.
- **preceding-sibling**
  - Todos los nodos que son hijos del nodo padre y que se encuentran antes del nodo de contexto.
  - Los nodos de atributo y de espacio de nombre no tienen hermanos.
- **preceding**
  - Todos los nodos descendientes del nodo raíz que se encuentran antes de la posición del nodo de contexto, excepto nodos de atributo y de espacio de nombre, y no son antepasados del mismo.
- **ancestor-or-self**
  - Todos los antepasados del nodo de contexto y el propio nodo.


not shown: attribute and namespace nodes and axes
ancestor-or-self
ancestor
preceding
following
parent
preceding-sibling
following-sibling
self
child
descendant
descendant-or-self

Nodos de Comprobación

- **Existen siete tipos de nodos de comprobación:**
- **name**
  - Un nombre XML que compara todos los elementos con el mismo nombre.
  - A lo largo del eje attribute compara todos los atributos con el mismo nombre.
  - A lo largo del eje namespace comparara todos los espacios de nombre con dicho prefijo.

- **prefix:***
  - Compara todos los nodos de elemento cuyos URI de espacio de nombre son iguales al URI del espacio de nombre al que está asignado el prefijo.

- **comment()**
  - Compara todos los nodos de comentario.

- **text()**
  - Compara todos los nodos de texto.

Nodos de Comprobación

- **node()**
  - Selecciona todos los nodos, independientemente del tipo.
  - Selecciona todos los nodos de elemento independientemente de su nombre.
  - A lo largo del eje attribute selecciona todos los nodos de atributo.
  - A lo largo del eje namespace selecciona todos los nodos de espacio de nombre.

- **processing-instruction()**
  - Selecciona todas las instrucciones de procesamiento.

Tipos de datos

Cada expresión XPath evalúa uno de estos cuatro tipos:

- **Booleano**
  - **E**l valor booleano (boolean) es un valor binario que puede ser verdadero o falso.
  - Para obtenerlos se utilizan los operadores relacionales (=, !=, <, >, <=, >=).
  - Se pueden combinar varias condiciones usando los operadores and y or.
  - No hay literales booleanos.
  - Para sustituirlos tenemos las funciones true() y false().
  - Se utilizan normalmente en los predicados de las rutas de localización.
- **Número**
  - **T**odos los números se ajustan a punto flotante de 64 bits.
  - Pueden ser positivos o negativos.
  - Los números incluyen los valores especiales Inf (infinito), -Inf (infinito negativo) y NaN (no es un número).
  - Operadores: + (suma), - (resta), * (multiplicación), div (división), mod (resto)
- **Cadena**
  - Secuencia de cero o más caracteres.
  - Los literales de la cadena se entrecomillan con comillas simples o dobles.
  - Para concatenar se utiliza la función concat().

Tipos de datos

- **Conjunto de nodos**
  - Un conjunto de nodos es una colección de cero o más nodos de un documento XML.
  - Las rutas de localización producen la mayoría de los conjuntos de nodos.
  - Un solo conjunto de nodos puede contener múltiples tipos de nodos.

Predicados

- Un predicado es una expresión XPath entre corchetes que sigue a la prueba de nodo en el paso de localización.
- Esta expresión devuelve normalmente un valor booleano.
- El predicado se evalúa frente a cada nodo que se encuentra en la lista de nodos de contexto.
- Si la expresión devuelve verdadero, dicho nodo se conserva en la lista.
- Si la expresión devuelve falso, el nodo se elimina de la lista.
- Si la expresión devuelve un número, ese nodo que se está evaluando se queda en la lista sólo si el número es igual que la posición de dicho nodo en la lista de nodos de contexto.
- Si la expresión devuelve un valor no booleano o un tipo que no es un número, el valor devuelto se convierte en un valor booleano usando la función boolean() para determinar si tiene que retener el nodo en el conjunto.

Funciones Xpath

Se definen 27 funciones integradas.

- **Formato**

  tipo_retorno nombre_función (tipo parámetro, tipo parámetro, ...)

- Algunas funciones aceptan un número variable de argumentos.

- Toda función devuelve uno de estos cuatro tipos: booleano, número, cadena o conjunto de nodos.

.Funciones booleanas

boolean(arg)	Convierte el argumento a boolean, siguiendo estas reglas • Si el argumento es un boolean devuelve su valor. • Un node-set devuelve false si está vacío, en caso contrario devuelve true. • Un string devuelve true si su longitud es distinta de cero. • Un numérico devolverá true si el argumento es positivo ó negativo, false si es cero ó NaN. • Si no es ninguno de estos tipos dependerá del tipo en particular.
not(arg)	Devuelve true si el argumento es false, false si el argumento es true.
true()	Devuelve true.
false()	Devuelve false.
lang()	Devuelve true si en el nodo de contexto está especificado el atributo xml:lang (<para xml:lang="en"/>) y el valor de este coincide con el valor del string que se le da como parámetro, false en caso contrario. Si el nodo de contexto es el especificado como ejemplo, lang("en") devuelve true.

.Funciones de conjuntos de nodos

number count(node-set)	Devuelve el número de elementos del node-set.
id(arg)	El objeto que se pasa como parámetro se convierte a string, y se devuelven todos los elementos con in parámetro ID igual que el argumento
number last()	Devuelve un número que contiene el tamaño del contexto.
string local-name(node-set)	Devuelve la parte local del nombre expandido del primer elemento del node-set.
string name(node-set)	Devuelve el QName del primer elemento del node-set.
string namespace-uri(node-set)	Devuelve el URI del nombre expandido, si no existe, si es nulo o el node-set está lacio retorna una cadena vacía
number position()	Devuelve la posición del elemento de contexto dentro del contexto.

.Funciones numéricas

number(arg)	Convierte el argumento a numérico • Si es una cadena con un número válido devuelve el valor, en caso contrario devuelve NaN • Si es un boolean devuelve 1 si es true, 0 si es false • Si es un node set este es convertido a string y después a numérico igual que si el parámetro fuese una cadena • Si no es ninguno de estos tipos dependerá del tipo en particular.
sum(node-set)	Devuelve el valor de la suma de todos los elementos del node-set convertido a numérico.
floor(number)	Devuelve el entero menor más cercano al parámetro
ceiling(number)	Devuelve el entero mayor más cercano al parámetro
round(number)	Devuelve el entero más cercano al parámetro

.Funciones de cadenas

string string(object)	Convierte el objeto pasado como parámetro a cadena de caracteres. • Un node-set devuelve el valor de convertir a string el primer elemento. • Un numérico o NaN devuelve NaN o Valor cero devuelve 0 o El valor infinito es convertido a (-)Infinity o Si es un valor entero devuelve (-)valor si número decimales o Cualquier otro caso devuelve (-)valor con separador decimal y al menos un decimal • Booleano devuelve 'true' ó 'false'. • Si no es ninguno de estos tipos dependerá del tipo en particular.
string concat(string, string, string*)	Devuelve la concatenación de todos sus argumentos
boolean starts-with(string, string)	Devuelve true si la primera cadena comienza con la segunda cadena, false en caso contrario.
boolean contains(string, string)	Devuelve true si la primera cadena contiene la segunda cadena, false en caso contrario.
string substring-before(string, string)	Devuelve una subcadena formada por los caracteres de la cadena de parámetro hasta que se encuentra la primera ocurrencia de la segunda cadena.
string substring-after(string, string)	Devuelve una subcadena formada por los caracteres de la cadena de parámetro desde que se encuentra la primera ocurrencia de la segunda cadena hasta el final.

.Funciones de cadenas

string substring(string, number, number?)	Devuelve una cadena compuesta por los caracteres del primer argumento, comenzando en la posición que indique el Segundo argumento hasta el final, si hay tercer argumento indica la longitud de la cadena a devolver
number string-length(string?)	Devuelve la longitud en caracteres del argumento. Si no hay parámetro se convierte a cadena el nodo de contexto y se aplica la función.
string normalize-space(string?)	Devuelve la cadena del argumento normalizada, eliminando espacios en blanco iniciales y finales y reduciendo secuencias de espacios a uno solo. Si no hay parámetro se convierte a cadena el nodo de contexto y se aplica la función.
string translate(string, string, string)	Devuelve una cadena que es el resultado de sustituir caracteres en el primer parámetro. Esta sustitución viene indicada por los parámetros 2 y 3, de modo que si en la cadena 1 encontramos un carácter de la cadena 2 se sustituye por el mismo carácter de la cadena 3 situado en la misma posición, esto se ve mejor con un ejemplo, translate("bar","abc","ABC") devuelve "BAr"

Xpath 2

- Diferencias con XPath 1.0.
  - Soporte para los tipos primitivos de datos XML Schema.
  - Se reemplaza el concepto de conjunto de nodos por el de secuencia.
  - Posibilidad de funciones lógicas en expresiones.
  - Se define una función para el tratamiento de secuencias: for.
  - Se define una función para el tratamiento condicionales: if .
  - Se añaden los cuantificadores existenciales y universales (some y every).
  - Operadores de combinación de secuencias: intersect, union, except.
  - Posibilidad de incluir comentarios.

- Soporte para los tipos primitivos de datos XML Schema.
- Se soportan los 19 tipos de datos básicos definidos en la especificación XML Schema (http://www.w3.org/TR/xmlschema-2/#built-in-primitive-datatypes) y se le añaden 5 tipos más: xs:untyped, xs:untypeAtomic, xs:anyAtomicType, xs:dayTimeDuration, xs:yearMonthDuration.

- Se reemplaza el concepto de conjunto de nodos por el de secuencia.
- Una secuencia es un conjunto ordenado de elementos en la que pueden aparecer elementos repetidos.
- En XPath 2.0 todo son secuencias, no se puede decir una expresión devuelve un valor numérico, sino una secuencia que contiene un valor numérico.

Xpath 2

- Se reemplaza el concepto de conjunto de nodos por el de secuencia.
- Se permite la construcción de secuencias; la secuencia está delimitada por los caracteres ( ) y cada uno de los valores están separados por el carácter , Por ejemplo.: (1, 2, 3, 4, 5)
- Podemos especificar secuencia de enteros ascendente usando el operador to. Por ejemplo.: (1 to 5)