

Laboratorio di Informatica II
Riccardo Ortale

XBase, XPath e XPointer

Lezione 14

Introduzione

- ◆ Qui esaminiamo:
 - XBase, XPath e XPointer

XML Base, XPath, XPointer e XLink

- ◆ XBase, XPath, XPointer e XLink sono quattro documenti di W3C per la specifica di link ipertestuali sui documenti XML.
- ◆ Erano un'unica proposta chiamata XLL (da cui la terna XML, XLL e XSL), poi divisa in quattro per semplicità.
 - **XML Base** specifica un meccanismo per esprimere URI di base per parti di documenti XML. E' simile all'elemento BASE di HTML.
 - **XPath** specifica i meccanismi per indicare percorsi all'interno di un documento XML. E' usato anche da XSLT. E' una raccomandazione W3C del 16/11/99
 - **XPointer** specifica i meccanismi per riferirsi a parti del documento XML (SGML permette di riferirsi solo ad elementi con l'attributo "ID", HTML solo ad elementi con l'attributo "NAME"). E' una candidate recommendation
 - **XLink** usa i meccanismi di indirizzamento di XPointer per descrivere link anche sofisticati tra documenti XML. E' una Recommendation

XML Base

- ◆ In molti casi è interessante specificare URI relativi all'interno di documenti. Alcune risorse XML hanno naturalmente un URI associato, che possa fare da base per gli URI relativi posti al suo interno.
 - ◆ Altre volte non li hanno (es.: documenti generati programmaticamente), a volte li hanno indesiderabili (documenti soggetti a trasformazioni intermedie).
 - ◆ In tutti i casi, può essere necessario suggerire una base esplicita per gli URI relativi presenti. XML Base è una raccomandazione W3C (27 giugno 2001) a questo scopo.
 - ◆ Sostanzialmente prevede di specificare la base usando da qualche parte (prima dell'uso) l'attributo riservato `xml:base` e come valore l'URI assoluto di base
- ◆ `<doc xml:base="http://www.site.com/dir/"> ... </doc>`

XPath

- ◆ Gli XPath sono una sintassi comune per XSL e XPointer per esprimere locazioni all'interno di documenti XML.
- ◆ XPath opera sulla struttura logica del documento, non su quella sintattica, usando una sintassi non XML accettabile all'interno di URI e attributi.
- ◆ Un XPath è un'espressione che restituisce un oggetto di uno di questi quattro tipi:
 - Un booleano
 - Una stringa
 - Un numero
 - Un insieme di nodi (nodi elemento, nodi attributi, nodi testo)

XPointer

- ◆ Gli XPointer sono indirizzi di locazioni interne a documenti XML. Possono essere usati per indicare link da o a specifiche parti di documenti XML.
- ◆ Gli XPointer sono una elaborazione dell'identificativo di frammento in un URL:
 - ◆ `http://www.site.com/dir/file.html#nome`
- ◆ Gli XPointer sono dunque usati in un locatore, tipicamente un URI o URL, per indicare un frammento di quella risorsa.
- ◆ Gli XPointer sono un'estensione degli XPath, di cui estendono leggermente la sintassi.

Location Path

- ◆ Il tipo più importante di XPath è il Location Path. Questo può essere o assoluto o relativo. Un Location Path assoluto inizia con ‘/’.
- ◆ Un Location Path è composto di una sequenza di passi di locazione (Location Steps) separati da ‘/’, e letti da sinistra a destra. Ogni termine individua più precisamente un frammento della risorsa individuata in precedenza.
- ◆ Es.: `/child::doc/child::chapter/descendant::para` identifica gli elementi “para” discendenti da un elemento “chapter”, a sua volta figlio diretto della radice “doc” del documento XML.

Location Step

- ◆ Un location step ha tre parti:
 - Un **asse**, che individua la direzione di specifica del location step nell'albero e rispetto al contesto.
 - Un **test di nodo**, che individua il tipo e il nome completo del nodo identificato dal location step
 - Zero o più **predicati** che raffinano ulteriormente l'insieme di nodi selezionati dal location step
- ◆ La sintassi è:

`axis::test [pred1] [pred2]... [pred N]`



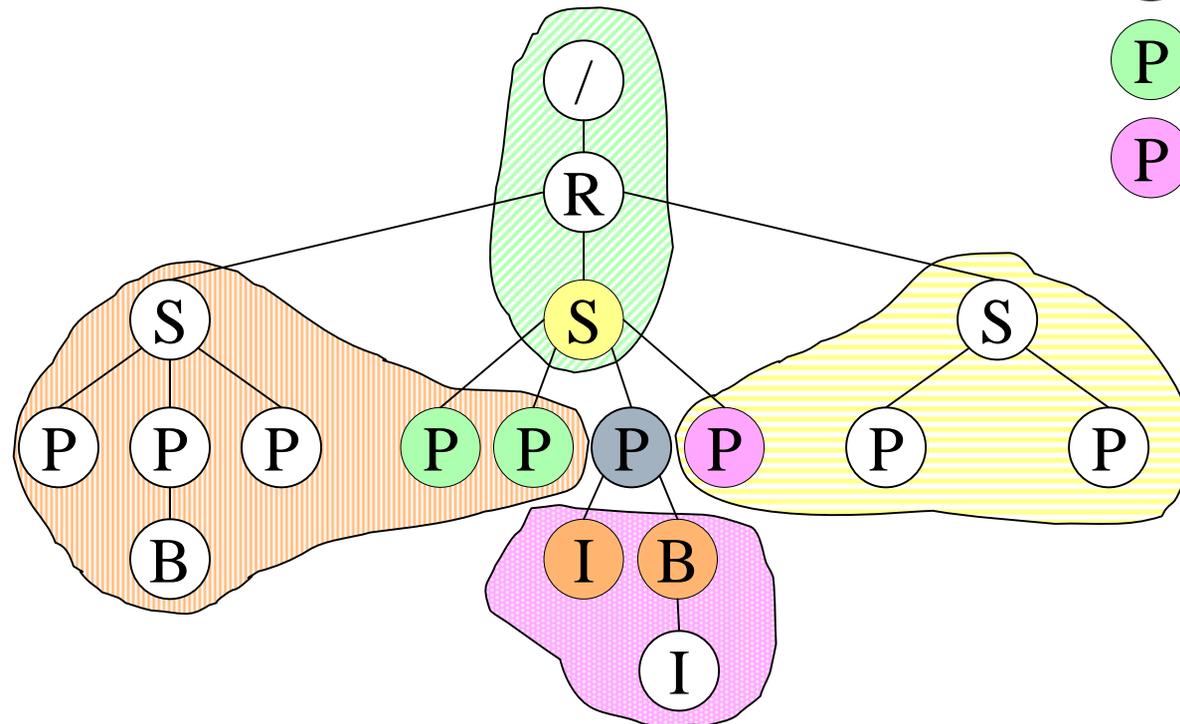
Si considera l'elemento sull'asse, soltanto se questo soddisfa il test sul nome.

Assi

- ◆ Gli assi identificano la direzione rispetto alla struttura del documento in cui andare a cercare l'oggetto da restituire rispetto al nodo contesto (NC). Tra gli assi possibili troviamo:
- **child, descendant**: figlio diretto e a qualunque livello del NC
 - **parent, ancestor**: il genitore immediato / qualunque livello del NC
 - **self, namespace**: il NC; il nodo namespace del NC
 - **attribute**: gli attributi del NC
 - **preceding-sibling, following-sibling**: i nodi allo stesso livello ma precedenti o seguenti il NC.
 - **preceding, following**: i nodi a qualunque livello (ma fuori al NC) che precedono o seguono il NC.
 - **descendant-or-self, ancestor-or-self**: come descendant e ancestor, ma considerando anche il NC.

Un esempio di alcuni assi

- P** Asse self::
- S** Asse parent::
- I** Asse child::
- P** Asse preceding-sibling::
- P** Asse following-sibling::
- Asse ancestor::**
- Asse descendant::**
- Asse preceding::**
- Asse following::**



Test di nodo (node test)

◆ Il test di un nodo identifica attraverso il nome o il tipo l'oggetto da restituire. Se un asse contiene un nodo, questo può essere verificato attraverso un test sul nome (se è un elemento) o attraverso altri meccanismi:

- Se l'asse è `attribute`, il nodo è un attributo
- Se l'asse è `namespace`, il nodo è il namespace
- Altrimenti, il nodo è un elemento.

◆ Il test può essere:

- `Un nome`: vero se il nodo (elemento o attributo) ha quel nome
- `text()`, `processing-instruction()`, `comment()`: vero se il nodo è di tipo testo, processing instruction o commento.
- `node()`: vero sempre
- `*`: vero per tutti i nodi del tipo definito dall'asse

Predicati

- ◆ Un predicato filtra l'insieme dei nodi rispetto alla direzione indicata dall'asse per produrre un nuovo insieme di nodi.
- ◆ Il filtro può essere attuato sulla posizione, o valutando un'espressione booleana. In questo secondo caso il risultato è quel sottoinsieme di nodi, tra quelli individuati finora, per cui l'espressione booleana è vera.
- ◆ L'espressione booleana può essere (ovviamente complicata da operatori booleani come or, and e not), un'espressione relazionale ($a=b$, $a!=b$, $a<b$, ecc.), e/o un Location Path, e/o una funzione pre-definita.
 - **Predicato:** [a] con a EspressioneBooleana
 - **EspressioneBooleana:** a or b, a and b, c, con a e b espressioni booleane e c espressione relazionale
 - **EspressioneRelazionale:** $a = b$, $a != b$, $a < b$, $a > b$, $a <= b$, $a >= b$, c, dove c espressione aritmetica
 - **EspressioniAritmetiche:** $a + b$, $a - b$, $a * b$, $a \text{ div } b$, $a \text{ mod } b$, c, -c, con c espressione di disgiunzione
 - **EspressioniDisgiunzione:** $a | b$, c, con c espressione di path
 - **Espressione di Path:** LocationPath o Espressione primaria
 - **Espressione Primaria:** numero, lettera o chiamata funzione

Funzioni predefinite

◆ XPath non definisce un elenco completo di funzioni, ma un elenco fondamentale.

- Funzioni sull'insieme di nodi: `last()`, `position()`, `count()`, `id()`, `local-name()`, `namespace-uri()`, `name()`
 - `child::para[position()=3]` individua il terzo nodo di nome “para” dentro al NC.
 - `child::para[last()]` individua l'ultimo nodo “para” nel NC.
 - N.B.: `child::para[3]` è equivalente a `child::para[position()=3]`
- Funzioni stringa: `string()`, `concat()`, `starts-with()`, `contains()`, `substring-before()`, `substring-after()`, `substring()`, `string-length()`, `normalize-space()`, `translate()`,
- Funzioni booleane e numeriche

Sintassi abbreviata

◆ In alcuni casi esistono delle forme abbreviate usabili invece della sintassi completa:

- `Child::x` si può abbreviare con `x`
- `Attribute::x` si può abbreviare con `@x`
- `Descendant` si può abbreviare con `'//'`, `self` con `'.'`, `parent` con `'..'`

◆ Esempi:

- `/doc/chapter[5]/section[2]`: la seconda sezione del quinto capitolo del documento.
- `chapter//para`: tutti i para discendenti a qualunque livello del nodo `chapter` figlio del NC
- `//para`: tutti i para discendenti a qualunque livello della radice del documento.

Altri esempi di XPath

- ◆ `para[@type="warning"]`: tutti i para figli del NC che abbiano l'attributo "warning".
- ◆ `para[@type="warning"] [5]`: il quinto para figlio di NC ad avere l'attributo type uguale a "warning".
- ◆ `para[5] [@type="warning"]`: il quinto para figlio di NC, ma solo se ha l'attributo type uguale a "warning".
- ◆ `chapter[title]`: il "chapter" figlio del NC che contenga uno o più elementi "title"

Punti sottili (1)

◆ Root node e document node

- Gli XPath assoluti iniziano sempre con /root (dove root è il nome dell'elemento radice), a parte /.
- Sono la stessa cosa? NO! Il nodo radice è più generale, e contiene l'elemento radice (corrispondente al tag radice), ma non coincide con lui.
- Il nodo radice contiene la dichiarazione XML, la dichiarazione di tipo di documento, eventuali altre processing instruction (es. per indicare il foglio di stile XSLT), **oltre al** nodo dell'elemento radice, chiamato, per evitare confusioni, *document element*.

◆ Assi in ordine rovesciato

- La maggior parte degli assi identifica la posizione seguendo l'ordine degli elementi nel documento. Gli assi che indicano elementi precedenti al nodo (e.g., ancestor, preceding e preceding-sibling) vanno in ordine rovesciato, dunque
 - preceding[1] viene **dopo** di preceding[2]
 - ancestor[1] è il padre, non la radice

Punti sottili (2)

◆ Caratteri proibiti

- XPath viene usato in contesti con sintassi particolari (es. attributi XML o URI). In questi casi i caratteri leciti in XPath ma proibiti nel contesto vanno adeguatamente *escapati*.
 - Es. `<xsl:template match="doc[position() < 3]">...`
- Da notare che - è sia un carattere lecito nei nomi XML sia un operatore matematico in XPath. Quindi è necessario precedere l'operatore matematico con uno spazio
 - `/doc/para[@foo-bar]` è diverso da `/doc/para[@foo - bar]`

◆ ID in XPath

- La funzione `id(foobar)` richiede di identificare quell'elemento che abbia un attributo di **tipo** ID il cui valore sia `foobar`.
- E' necessario avere il DTD per riconoscere che un attributo è di tipo ID. Poiché i DTD non sono necessari nei documenti XML, questa funzione può essere verificata solo da un parser validante, e quindi non è universale.

XPath e Explorer 5.0

◆ Microsoft ha messo a disposizione tre versioni (ad oggi) di librerie XML, leggermente diverse le une dalle altre:

- MSXML 2.0: supporto pieno di XML, parziale di Namespace, versione draft, non definitiva, di XSL e XPath, niente Schema. Internet Explorer 5.0
- MSXML 3.0: supporto pieno di XML, pieno di Namespace, versione definitiva di XSL e XPath, versione draft, non definitiva, di Schema. Internet Explorer 5.5
- MSXML 4.0: supporto pieno di XML, Namespace, XSLT, XPath, versione **quasi** definitiva di Schema. Internet Explorer 6.0

◆ In particolare, MSXML 2.0, per quel che riguarda XPath:

- Solo sintassi abbreviata, pochi assi
- Mancano completamente gli operatori aritmetici, e molti altri sono carenti
- Alcune funzioni hanno nomi diversi: `end()` invece di `last()`, `index()` invece di `position()`. Altre funzioni non esistono.

XPointer

- ◆ XPointer è una delle attività più intricate e disgraziate del W3C (anche se HTTP-NG lo supera)
- ◆ Dopo lunghe e intricate vicissitudini, una versione limitata e parziale è diventata Recommendation W3C nel marzo 2003. Il documento iniziale è stato frammentato in 4, tre dei quali sono stati approvati, ed il quarto ancora no.
- ◆ Gli XPointer sono meccanismi per indicare all'interno di un URI la parte fragment, ovvero la specifica sezione del documento a cui faccio riferimento.
- ◆ Questa parte può essere specificata o sulla base di un attributo id esistente (analogo a NAME di HTML), oppure indicando il nodo sulla base di un percorso di qualche tipo.

XPointer (2)

- ◆ **XPointer framework** (W3C recommendation) stabilisce l'organizzazione generale degli XPointer e la forma degli schemi da usare per indicare i frammenti interessanti.
- ◆ **XPointer xmlns()** (W3C recommendation) stabilisce come specificare nomi qualificati all'interno degli schemi XPointer
- ◆ **XPointer element()** (W3C recommendation) identifica due specifici schemi di indicazione di elementi all'interno di un XPointer: bare names (attributi di tipo id) e child sequences (sequenze di elementi figlio in un albero).
- ◆ **XPointer xpointer()** (W3C working draft) individua uno schema di XPointer che permette l'uso di XPath, e introduce due nuovi concetti, il punto ed il range, per indicare frammenti che non sono sottoalberi di un documento XML.

XPointer (3)

◆ XPointer element()

- un *bare name* identifica l'elemento il cui ID è il nome dato
 - `http://www.sito.com/file.xml#intro` diventa
`http://www.sito.com/file.xml#element(intro)`
- Una *child sequence* è una sequenza di attraversamento di un albero nei suoi figli
 - `http://www.sito.com/file.xml#element(1/2)`
- Le cose possono mescolarsi: una sequenza di attraversamento può partire da un *bare name*
 - `http://www.sito.com/file.xml#xpointer(intro/1/2)`

◆ XPointer xpointer()

- Uno o più forme generali di frammento (General Fragment part). Può essere o un XPath, o un XPath con punti e range.
 - `http://www.sito.com/file.xml#xpointer(a/b)`

Estensioni a XPath (1)

◆ XPointer `xpointer()` estende il concetto di node in quello di locazione. Una locazione è un nodo, o un punto, o un range di un documento XML. Quindi XPointer definisce due nuovi location types:

- Point: è definito da un nodo ed un indice, e rappresenta una posizione descritto dall'indice all'interno del nodo. Se il nodo non ha nodi figli, allora l'indice si riferisce alla stringa contenuta nel nodo.
- Range: è definito come due punti, il primo precedente al secondo.

Estensioni a XPath (2)

- ◆ Inoltre XPointer definisce alcune funzioni aggiuntive, tra cui:
 - `range-to(nodo)`: dall'inizio di NC al nodo parametro
 - `xpointer(id("chap1")/range-to(id("chap2")))`
 - `string-range()`: una sottostringa del NC
 - `string-range(//title,"Thomas Pynchon") [17]`
 - `range(node-set)`: tutti i range espressi dai parametri
 - `start-point()`, `end-point()`: l'inizio e la fine del NC
 - `here()`, `origin()`: il punto contesto, e l'origine del link.

Conclusioni

- ◆ Qui abbiamo parlato di
 - Xbase, XPath e XPointer

Riferimenti

- ◆ J. Clark, S. DeRose, XML Path Language (XPath), Version 1.0, W3C Recommendation 16 November 1999, <http://www.w3.org/TR/xpath>
- ◆ Paul Grosso, Eve Maler, Jonathan Marsh, and Norman Walsh, eds.. XPointer Framework, W3C Recommendation 25 march 2003, <http://www.w3.org/TR/xptr-framework/>
- ◆ Paul Grosso, Eve Maler, Jonathan Marsh, and Norman Walsh, eds.. XPointer element() Scheme, W3C Recommendation 25 march 2003, <http://www.w3.org/TR/xptr-element/>
- ◆ P. Grosso, E. Maler, J. Marsh, and N. Walsh, editors. XPointer xmlns() Scheme. World Wide Web Consortium, W3C Recommendation 25 march 2003, <http://www.w3.org/TR/xptr-xmlns/>.
- ◆ S. DeRose, E. Maler, and R. Daniel Jr., editors. XPointer xpointer() Scheme. World Wide Web Consortium, W3C Working Draft 19 December 2002, <http://www.w3.org/TR/xptr-xpointer/>
- ◆ Un utile tutorial su Xpath: <http://it.wikipedia.org/wiki/XPath>