



XML Databases

II. XML Query languages

II.2 XPath and XPointer

Weeks 4 - 6

Outline

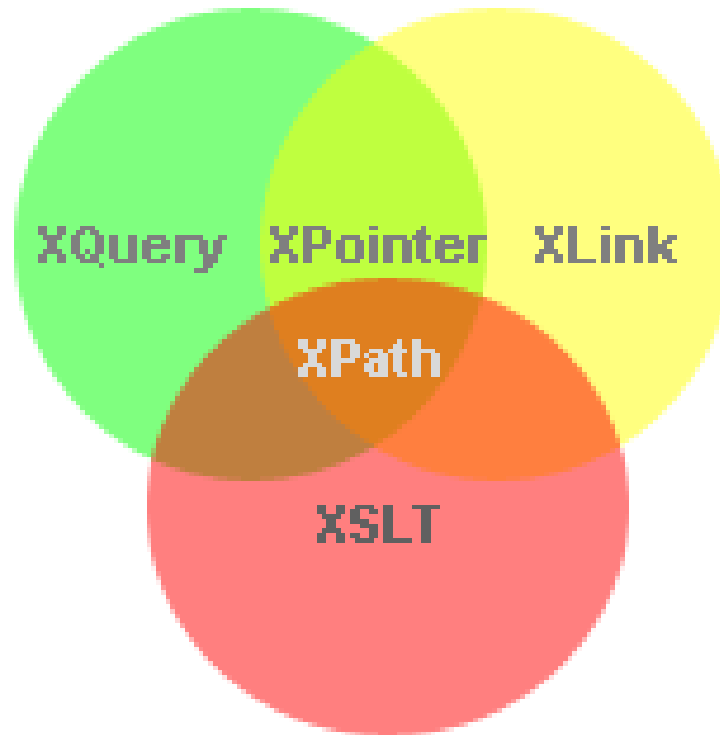
II.1 Introduction

II.2 Selecting

A. XPath

B. XPointer

II.3 Transforming



(from w3schools)

A. XQuery 1.0

- The primary purpose of XPath is to address parts of an XML
- An XPath is made of steps
- An XPath suppose (like every query language) a **data model** (common with XSL)

Based on the W3C specification (November 16, 1999)

A.1. XPath Data Model

“The XML Information Set”

- Seven node types
 - ☐ Root nodes
 - ☐ Element nodes
 - ☐ Text nodes
 - ☐ Attribute nodes
 - ☐ Namespace nodes
 - ☐ Processing instruction nodes
 - ☐ Comment nodes

A.1. XPath data model (cont.)

- Each node has
 - a string value
 - expanded-name (namespace URI + local part)
- There is an ordering among nodes (also called pre-order)
- Node properties are derived from XML (order among children, tree structure)

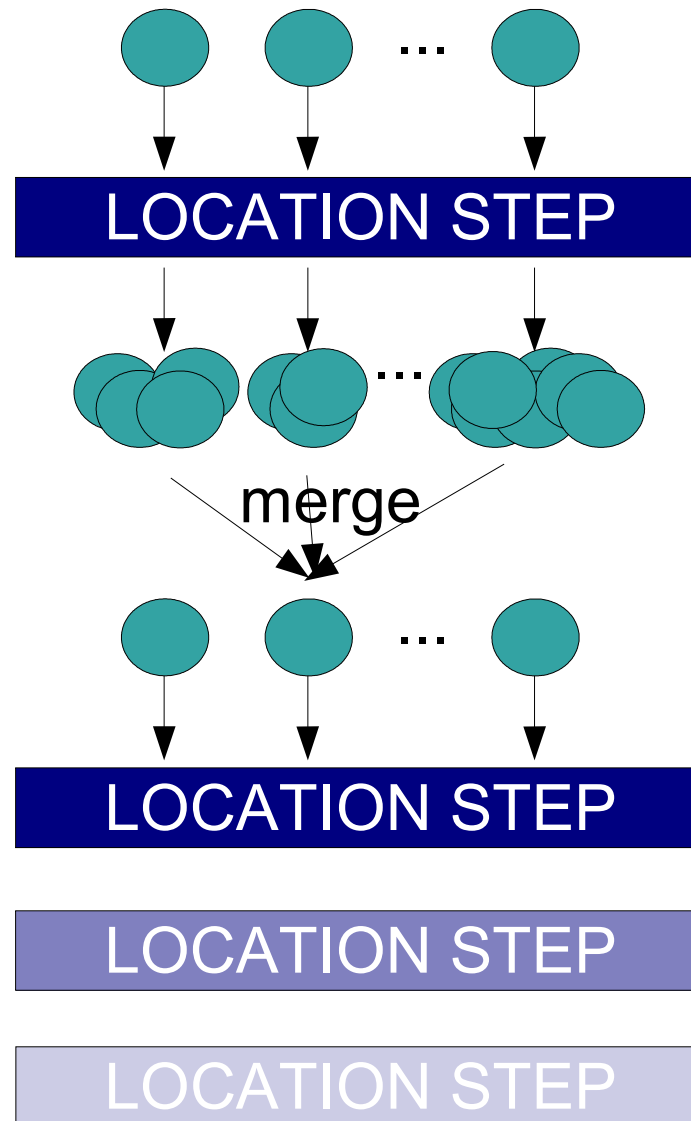
A.2. XPath basic types

- When evaluated, an XPath expression yields one of this four basic types:
 - node-set (an unordered collection of nodes without duplicates)
 - boolean (true or false)
 - number (a floating-point number)
 - string (a sequence of UCS characters)

A.3. XPath context

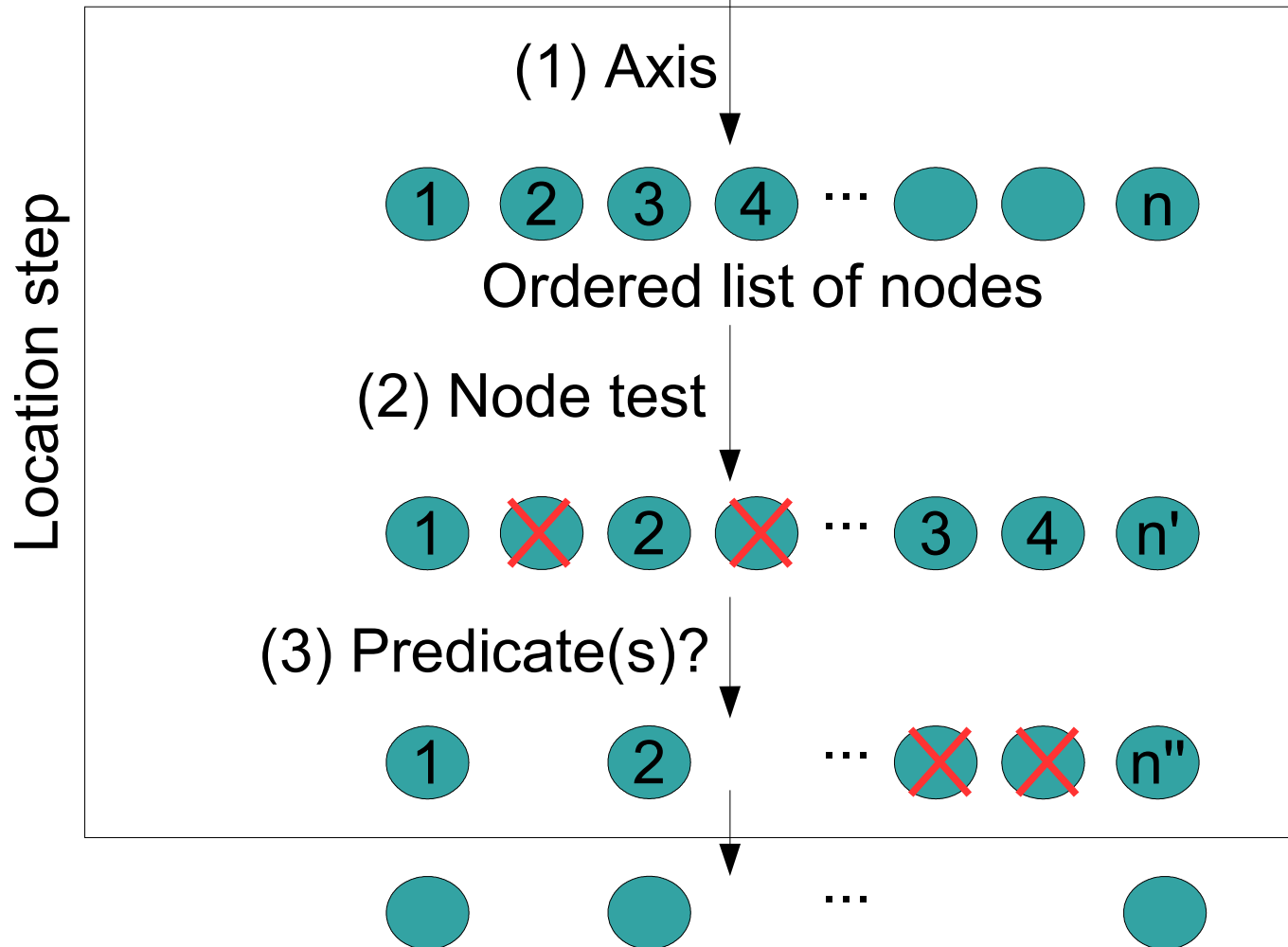
- A node (the context node)
- The context position and the context size (two integers)
- A set of variable bindings
- A function library
- A the set of namespace declarations in scope for the expression

A.4. XPath processing

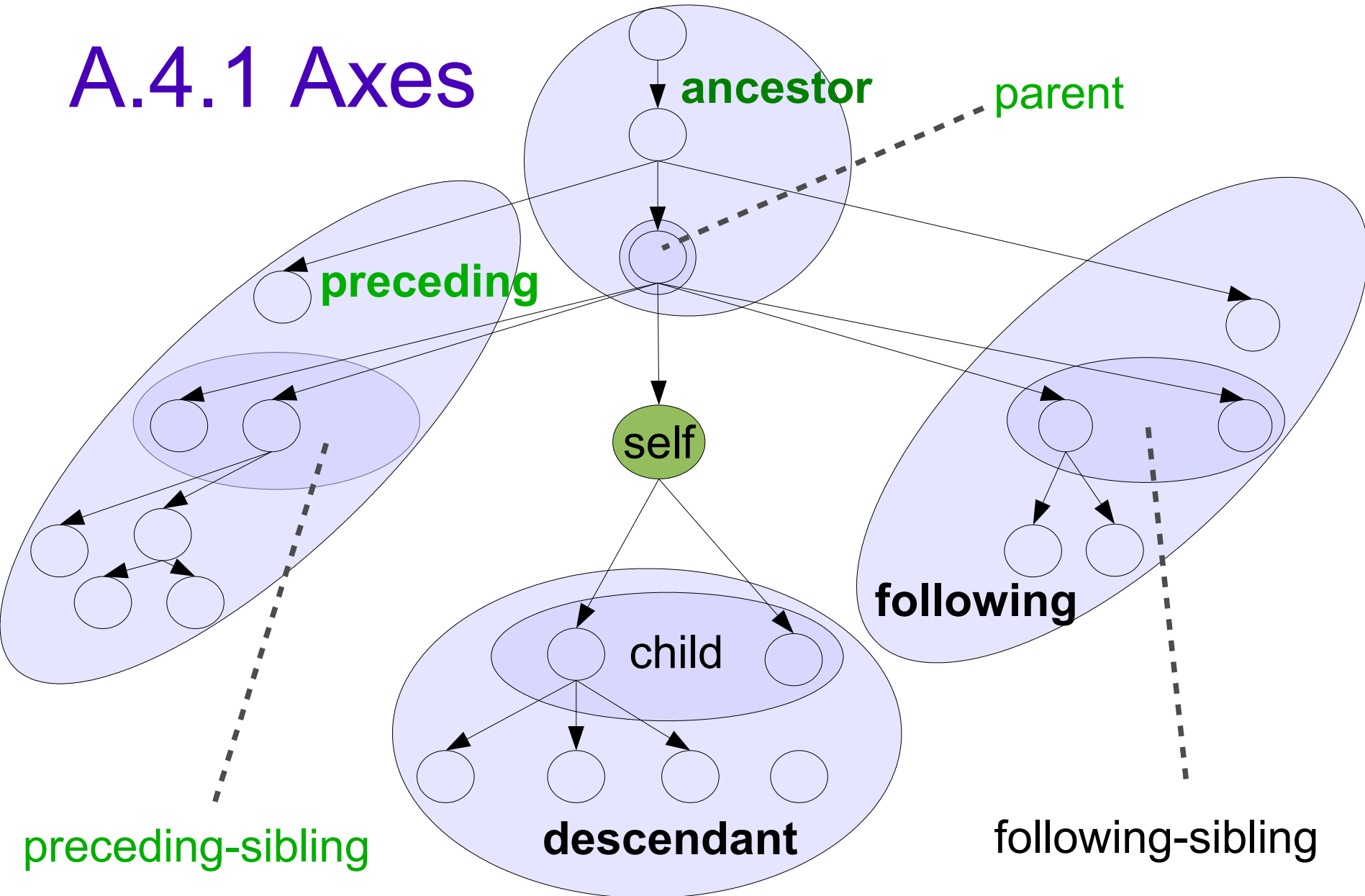


A.4. XPath processing (cont)

Initial context node 



A.4.1 Axes



A.4.1 Axes (cont)

- Other axes
 - ☐ ancestor-or-self, descendant-or-self
 - ☐ namespace
 - ☐ attribute
- Axes imply:
 - ☐ An order among nodes
 - ☐ A principal node type

A.4.2 Node test

- Any node: * (within a namespace: **NS:***)
- Node name: **NS:NN** or **NN**
- Node type:
 - ☐ comment()
 - ☐ text()
 - ☐ processing-instruction(optional argument)
 - ☐ node()

A.4.3. Predicates

- A predicate is an **expression** (the main language construct)
- For each node of the initial set, the expression is evaluated and returns a value which is converted to a boolean value (True => the node is kept, False => the node is removed):
 - Integer value => true only if the context position is equal to this integer
 - Node set, string => true if not empty
 - Real => not 0 nor NaN
 - Other => ?

A.4.4. Putting all together (A step)

axe::node-test

axe::node-test

[expression]
axe::node-test[expression]

[expression]
axe::node-test[expression][expression]

[expression]
axe::node-test[expression][expression][expression]
[expression]

A.4.4. Putting all together (Steps)

step

step/step



Relative

/ste

p/step/ste



Absolute

A.4.4. Putting all together (Abbr)

■

■ ■

node-test

@QName

QName//QName

A.4.5. An expression

■ Operators (lowest priority first): or • and • = != •
 <= < >= > • + - • div mod * • (unary) - • | •
 path expression

■ A Path expression

- A series of steps
- A filter expression (followed by an optional series of steps or predicates):
 variable reference, an expression
 between parenthesis, literal, number or
 a function call

A.4.6 Some core functions

- **Node set:** last(), position(), count(), id(), local-name(*node-set*), namespace-uri(*node-set*), name(*node-set*)
- **String:** string, concat, starts-with, substring-before, substring-after, substring, string-length, normalize-space, translate
- **Boolean:** boolean, not, true, false, lang
- **Number:** number, sum, floor, ceiling, round

A.4.7. Examples

- `child::para, child::*, child::text(), child::node()`
- `attribute::name, attribute::*`
- `descendant::para, ancestor::div, ancestor-or-self::div, descendant-or-self::para, self::para`
- `child::chapter/descendant::para, child::*/child::para, /`
- `/descendant::para, /descendant::olist/child::item,`
- `child::para[position()=1], /child::doc/child::chapter[position()=5]/child::section[position()=2]`
- `child::para[attribute::type="warning"], child::para[attribute::type='warning'][position()=5]`
- `child::*[self::chapter or self::appendix], child::*[self::chapter or self::appendix][position()=last()]`

A.4.8. Pause (for me)

```
<nutrition>
<daily-values>
<total-fat units="g">65</total-fat>
<saturated-fat units="g">20</saturated-fat>
<cholesterol units="mg">300</cholesterol>
<sodium units="mg">2400</sodium>
<carb units="g">300</carb>
<fiber units="g">25</fiber>
<protein units="g">50</protein>
</daily-values>
```

```
<food>
  <name>Truffles, Dark Chocolate</name>
  <mfr>Lyndon's</mfr>
  <serving units="g">39</serving>
  <calories total="220" fat="170"/>
  <total-fat>19</total-fat>
```

```
    <saturated-fat>14</saturated-fat>
    <cholesterol>25</cholesterol>
    <sodium>10</sodium>
    <carb>16</carb>
    <fiber>1</fiber>
  <protein>1</protein>
  <vitamins>
    <a>0</a>
    <c>0</c>
  </vitamins>
  <minerals>
    <ca>0</ca>
    <fe>0</fe>
  </minerals>
</food>
<food>...</food>
</nutrition>
```

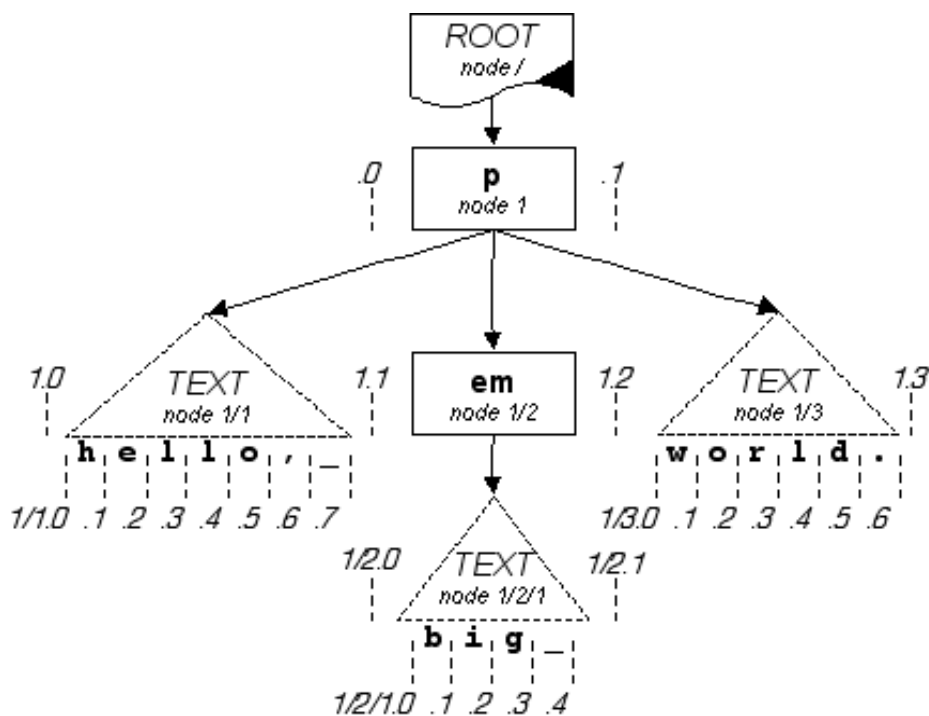
B. XPointer

- Addressing **portions** of XML documents
- Generalisation of XPath:
 - Manipulated: Location \supseteq Node
 - Result: Location-set \supseteq Node-set
 - Document Order
 - Consequence: an XPath is an XPointer
- XPointer is a **framework**, composed of the framework basis, the xmlns, element and xpointer **schemes**

B.1. Point and Range

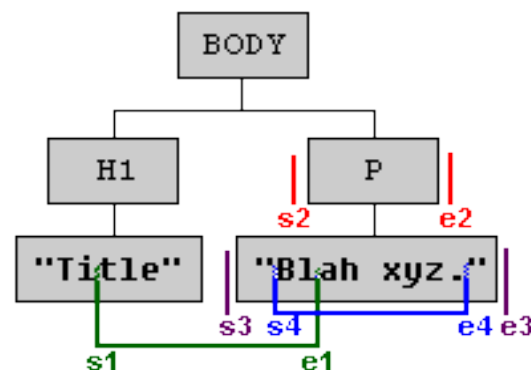
- **Point:** A location (position) within an XML file with no content or children
 - Defined by a container node + an offset
- **Range:** The content between two points
- **Location:** The 7 nodes from XPath data model + point and range

B.1. Point and range (contd)



XML snippet: `<BODY><H1>Title</H1><P>Blah xyz.</P></BODY>`

Range markers (s1, e1, s2, e2, s3, e3, s4, e4) are shown above the snippet, indicating the start and end of various ranges.



Range	s		e	
	Node	Offset	Node	Offset
s1 — e1	Text1	2	Text2	2
s2 — e2	BODY	1	BODY	2
s3 — e3	P	0	P	1
s4 — e4	Text2	0	Text2	9

(source: w3c)

B.2. XPath extension

Point and Range

- For point/range
 - ☐ expanded-name is null
 - ☐ string value is empty
- Axes
 - ☐ Not empty: self, descendant-or-self, parent, ancestor, ancestor-or-self
 - ☐ Empty: following, following-sibling, preceding, preceding-sibling, child, descendant
- Extension of node types

B.2. XPath extension

Document order

- How to compare two ranges?
 - ☐ covering range
 - ☐ start point then end point
- How to compare two points?

B.2. XPath extension

Step and string

- How to specify a range?

- **range-to** step

- p/range-to(it)

- descendant::REVST/range-to(following:REVEN[1])

- **string-range** function

- string-range(//title,"Thomas Pynchon")[17]

- string-range(//P,"Thomas Pynchon",8,0)[3]

B.2. XPath extension

Other functions

- `covering-range(location-set)`
- `range-inside(location-set)`
- `start-point(location-set)`
- `end-point(location-set)`

- `here()` [only within a single document]
- `origin()` [only when traversing a link]