Information Systems

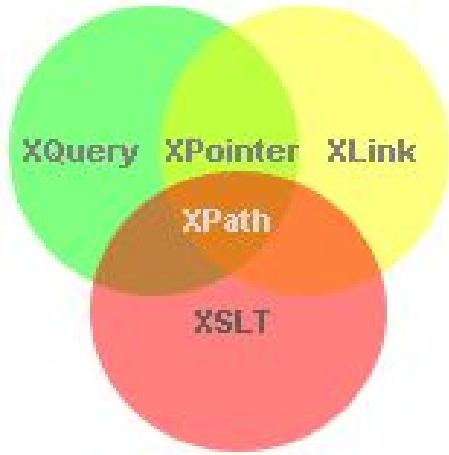
XPath

Nikolaj Popov

Research Institute for Symbolic Computation
Johannes Kepler University of Linz, Austria
popov@risc.uni-linz.ac.at

Outline

XPath



What is XPath?

- XPath is a language whose primary purpose is to provide common syntax and functionality to address parts of XML documents.
- XPath uses path expressions to navigate in XML documents.
- XPath contains a library of standard functions.

XPath

- XPath operates on the logical structure of an XML document and uses a syntax that resembles to the path constructions in URIs.
- XPath models an XML document as a tree of nodes (e.g. elements, attributes, namespaces, etc.)
- XPath expressions can compute strings, numbers, sets of nodes from the data of XML documents.

Location Paths

- Location paths are special expressions for selecting a set of nodes.
- A location path consists of location steps composed together from left to right and separated by ’/’.
- An absolute location path is one that starts with a ’/’.
- Relative location paths are defined always with respect to the context node.

Example

The node selection is analogous to the file selection in a Unix-like file system.

```
../reports/*/summary
```

Location Paths

Example

Path Expression	Result
`/bookstore`	Selects the root element `bookstore` Note: If the path starts with a slash (`/`) it always represents an absolute path to an element!
`bookstore/book`	Selects all `book` elements that are children of `bookstore`.

Location Paths

Example

Path Expression	Result
`//book`	Selects all `book` elements no matter where they are in the document.
`bookstore//book`	Selects all `book` elements that are descendant of the `bookstore` element, no matter where they are under the `bookstore` element.
`//@lang`	Selects all attributes that are named `lang`.

Predicates

- Predicates are used to find a specific node or a node that contains a specific value.
- Predicates are always embedded in square brackets.

Example

Path Expression	Result
`/bookstore/book[1]`	Selects the first `book` element that is the child of the `bookstore` element.
`/bookstore/book[last()-1]`	Selects the last but one `book` element that is the child of the `bookstore` element.
`/bookstore/book[position()<3]`	Selects the first two `book` elements that are children of the `bookstore` element.

Predicates

Example

Path Expression	Result
`//title[@lang='eng']`	Selects all the title elements that have an attribute named lang with a value of `'eng'`.
`/bookstore/book[price>35.00]`	Selects all the `book` elements of the `bookstore` element that have a `price` element with a value greater than 35.00.

Selecting Unknown Nodes

XPath wildcards can be used to select unknown XML elements.

Example

Wildcard	Result
`/bookstore/*`	Selects all the child nodes of the bookstore element.
`//*`	Selects all elements in the document.
`//title[@*]`	Selects all title elements which have any attribute.

Selecting Several Paths

By using the | operator in an XPath expression you can select several paths.

Example

Path Expression	Result
`//title | //price`	Selects all the `title` AND `price` elements in the document.

Location Steps

Location steps have the following parts:

- **axis**. It specifies the (in-tree) relationship between the context node and the nodes selected by the location step:

  Available axes: child, descendant, parent, ancestor, self, descendant-or-self, ancestor-or-self. (Used explicitly in "long notation")
- **node test**. Specifies the node type for the nodes selected by the location step (separated by :: from the axis).
- **predicate**. It specifies further expressions with boolean value, to refine the selected node set (enclosed in [ ], described on before).

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<folder>
  <email date='15 Jan 2018'>
    <from>robert@uni.com</from>
    <to>oliver@uni.com</to>
    <subject>Meeting</subject>
    Could we meet this week to discuss the
    preparation for the IS (do not confuse
    with ISIS please!)  examination?  -Rob
  </email>
</folder>
```

- This location path (using the “long notation”) selects all attributes of all the email elements.
  `/child::folder/des-or-self::email/attribute::*`
- This selects only the attribute of the first email element in the XML document.
  `/child::folder/des-or-self::*[position()=1]/@*`

XPath Expressions

- Simple expressions: numerical and string literals, variable references, function calls.
- The value of a variable `x` can be retrieved by `$x`.
- Basic arithmetic operations are available for numbers.
- More complex expressions are location paths and boolean expressions (e.g. using $<$, $>$, $!=$, $=$ and logical connectives `and`, `or`).

Summary

- XPath provides a language for addressing parts of XML documents.
- XPath uses path expressions to navigate in XML documents.