XPath

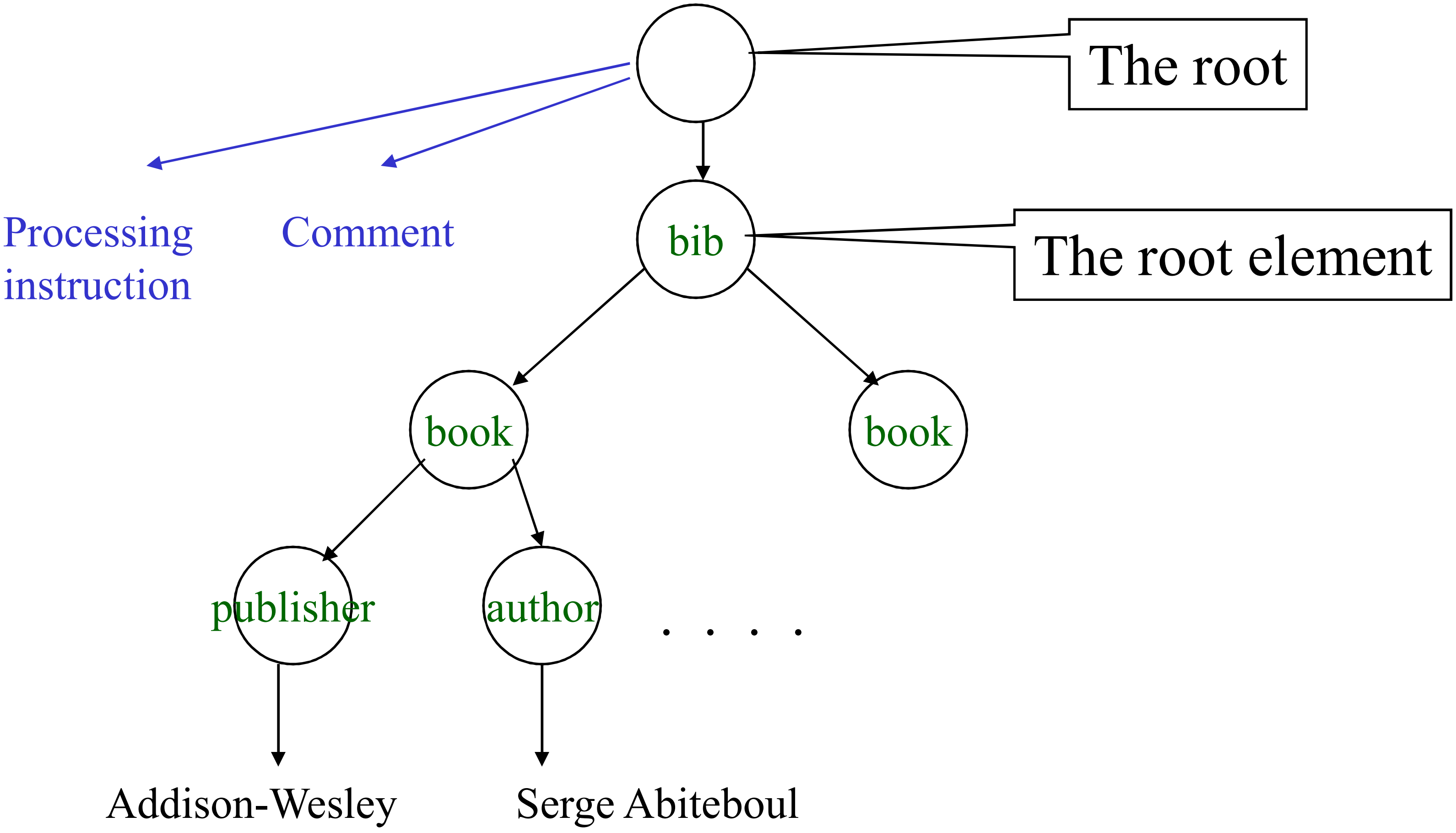
In this lecture

- Review of the XPath specification
  - data model
  - examples
  - syntax

Example for XPath Queries

```
<bib>
    <book>  <publisher> Addison-Wesley </publisher>
             <author> Serge Abiteboul </author>
             <author> <first-name> Rick </first-name>
                        <last-name> Hull </last-name>
             </author>
             <author> Victor Vianu </author>
             <title> Foundations of Databases </title>
             <year> 1995 </year>
    </book>
    <book price="55">
            <publisher> Freeman </publisher>
             <author> Jeffrey D. Ullman </author>
             <title> Principles of Database and Knowledge Base Systems </title>
             <year> 1998 </year>
    </book>
</bib>
```

Data Model for XPath



Much like the Xquery data model

XPath: Simple Expressions

/bib/book/year

Result: <year> 1995 </year>
<year> 1998 </year>

/bib/paper/year

Result: empty (there were no papers)

XPath: Restricted Kleene Closure

```
//author
```

```
Result:<author> Serge Abiteboul </author>
       <author> <first-name> Rick </first-name>
                <last-name> Hull </last-name>
       </author>
       <author> Victor Vianu </author>
       <author> Jeffrey D. Ullman </author>
```

```
/bib//first-name
Result:  <first-name> Rick </first-name>
```

Xpath: Functions

/bib/book/author/text()

Result: Serge Abiteboul
Jeffrey D. Ullman

Rick Hull doesn’t appear because he has firstname, lastname

Functions in XPath:

- text() = matches the text value
- node() = matches any node (= * or @* or text())
- name() = returns the name of the current tag

Xpath: Wildcard

//author/*

Result: <first-name> Rick </first-name>
<last-name> Hull </last-name>

* Matches any element

Xpath: Attribute Nodes

/bib/book/@price

Result: “55”

@price means that price is has to be an attribute

Xpath: Qualifiers

/bib/book/author[firstname]

```
Result: <author> <first-name> Rick </first-name>
                 <last-name> Hull </last-name>
        </author>
```

Xpath: More Qualifiers

/bib/book/author[firstname][address[//zip][city]]/lastname

Result: <lastname> … </lastname>
<lastname> … </lastname>

Xpath: More Qualifiers

/bib/book[@price < “60”]

/bib/book[author/@age < “25”]

/bib/book[author/text()]

Xpath: Summary

bib	matches a bib element
*	matches any element
/	matches the root element
/bib	matches a bib element under root
bib/paper	matches a paper in bib
bib//paper	matches a paper in bib, at any depth
//paper	matches a paper at any depth
paper|book	matches a paper or a book
@price	matches a price attribute
bib/book/@price	matches price attribute in book, in bib
bib/book/[@price<“55”]/author/lastname	matches…

Xpath: More Details

- An Xpath expression, p, establishes a relation between:
  - A *context node*, and
  - A node in the *answer set*
- In other words, p denotes a function:
  - S[p] : Nodes -> {Nodes}
- Examples:
  - author/firstname
  - . = self
  - .. = parent
  - part/*/*/subpart/../name = part/*/*[subpart]/name

The Root and the Root

- <bib> <paper> 1 </paper> <paper> 2 </paper> </bib>
- bib is the "***document element***"
- The "***root***" is above bib

- /bib = returns the document element
- / = returns the root

- Why ? Because we may have comments before and after <bib>; they become siblings of <bib>

- This is advanced xmlogy

Xpath: More Details

- We can navigate along 13 axes:

  ancestor
  ancestor-or-self
  attribute
  child
  descendant
  descendant-or-self
  following
  following-sibling
  namespace
  parent
  preceding
  preceding-sibling
  self

We've only seen these, so far.

Xpath: More Details

- Examples:
  - child::author/child:lastname = author/lastname
  - child::author/descendant::zip = author//zip
  - child::author/parent::* = author/..
  - child::author/attribute::age = author/@age
- What does this mean ?
  - paper/publisher/parent::*/author
  - /bib//address[ancestor::book]
  - /bib//author/ancestor::*//zip

Xpath: Even More Details

- name() = the name of the current node
  - /bib//*[name()=book] same as /bib//book

- What does this mean ?

  /bib//*[ancestor::*[name()!=book]]

  - In a different notation bib.[^book]*._

- Navigation axis gives us strictly more power !