Querying Linked XML Documents in the Web

Wolfgang May
Institut für Informatik
Universität Freiburg
Germany
may@informatik.uni-freiburg.de

WWW Conference
Honolulu, 8.5.2002

Situation

- focus on databases
- autonomous XML sources on the Web
- sources provide data + external schema
- links between sources
- Scenario: geographical database
  - an XML instance for all countries
  - for each country, an instance with its cities

- are links transparent (vs. external schema)?
- queries against the referencing document must be "forwarded"

Simple Links

XPointer and XLink:

- specify how to *express* inter-document links in XML
- simple links similar to the HTML <A href="..."> construct.

Capital of countries:

```
<country code="B">
   <capital xlink:type="simple"
            xlink:href="file:cities-B.xml#//city[@name='Brussels']"/>
   :
</country>
```

- XLink specification tailored to browsing, not to querying.

Querying along XLinks

There is not yet an official proposal

- how to add link semantics to the actual data model (e.g., the XML Query Data Model)
- how to express/process queries through links
- for evaluation strategies

Data Models for XLinks

- extend the abstract data model with a linking construct
  - additional explicit navigation operator required
- abstract data model makes the links transparent:
- each link can be seen as a view definition
  - integrates external schemata
  - embedded views implicitly become subtrees
  - queried by standard XPath expressions

Transparent Links

- regard link elements to be *transparent*

```
<country code="B">
   <capital xlink:type="simple"
            xlink:href="file:cities-B.xml#//city[@name='Brussels']">
            attributes of Brussels>
      contents of Brussels
   </capital>
   :
</country>
```

query: `//country[@code="B"]/capital/population`

similar for out-of-line extended links

Evaluation of XPath Expressions

Cost Factors and Requirements:

- answering time
- actual costs (contents providers, communication)

- computation expenses
- internet communication
  - volume of transferred data
  - number of queries; time for establishing a connection
- availability vs. up-to-dateness

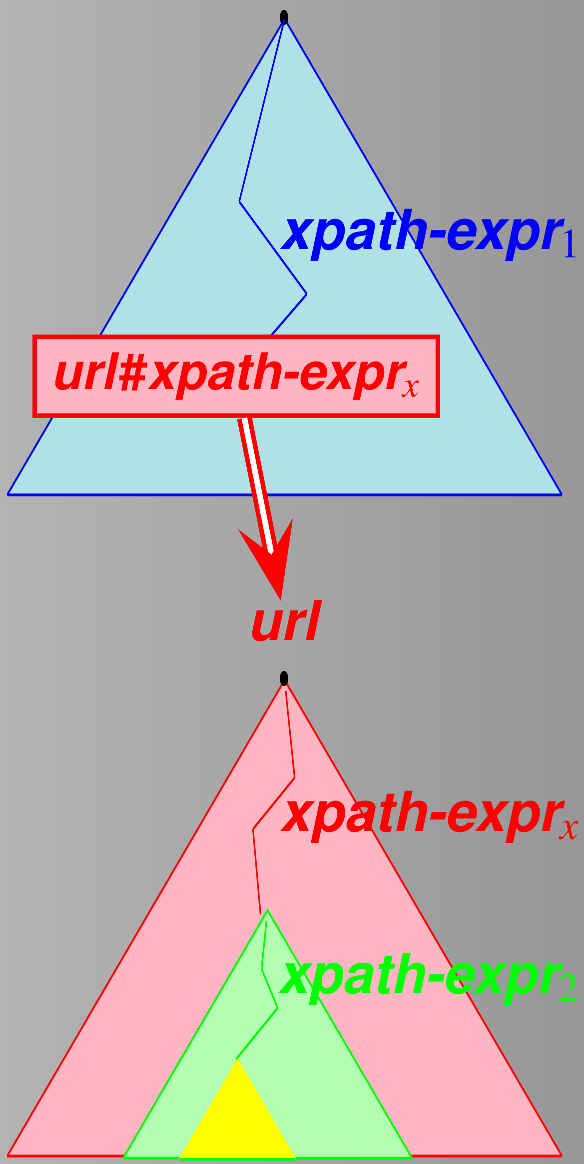
- strategies known from relational views:
  - Prefetching, Caching
  - Materialized Views

Evaluation of XPath Expressions

XPath Query: $\boxed{\textit{xpath-expr}_1/\textit{xpath-expr}_2}$

- result of *xpath-expr*$_1$ contains a link node
  <*linkelement* xlink:type="simple"
  href="*url#xpath-expr*$_x$"/>
- link pointer $\boxed{\textit{url\#xpath-expr}_x}$
  defines a view $\mathcal{V}$
- Then evaluate $\boxed{\textit{xpath-expr}_2}$
  against $\mathcal{V}$.

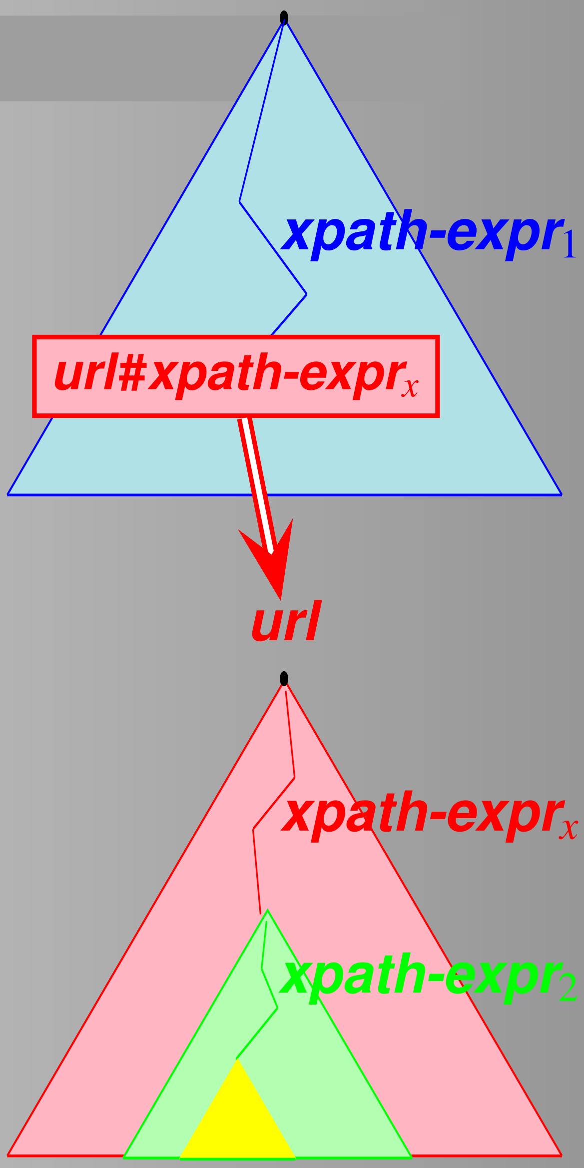
when and *where* to evaluate?



Evaluation of XPath Expressions

Where?

- Server answers XPath queries:
  - evaluate *xpath-expr*$_x$/*xpath-expr*$_2$ at the server, transfer the result
  - evaluate *xpath-expr*$_x$ at the server, transfer the result. Then, evaluate *xpath-expr*$_2$ at the client.
- Server makes XML document accessible:
  - transfer contents of *url* and evaluate *xpath-expr*$_x$/*xpath-expr*$_2$ at the client.
- Server answers only some XPath queries: query rewriting needed.



Evaluation of XPath Expressions

When?

- when parsing an XML instance:
  - access the referenced documents, evaluate locally later
  - materialize the virtual tree views (evaluate: local or remote)

  $\Rightarrow$ no internet access required at query answering time

  $\Rightarrow$ outdated information
- evaluation when the link is firstly used by a query:
  - local vs. remote evaluation
  - caching document or link-view or nothing?
- evaluation at each time when a link is used
  - communication and computation overhead
  - always up-to-date information

Switches

- when is the link evaluated?
  (at parse time or at query answering time),
- where does the computation take place?
  (locally or at the remote server), and
- which (intermediate) results should be stored and reused.

Define dbxlink namespace

Switches

Activating Event: dbxlink:actuate attribute:

auto: XPointer is evaluated when the node containing it is parsed

user: XPointer is evaluated when it is used by a query

noaction: the unresolved link remains – as a kind of view *definition* – in the answer tree
(mainly for information brokers, returning unresolved links to be processed by the client)

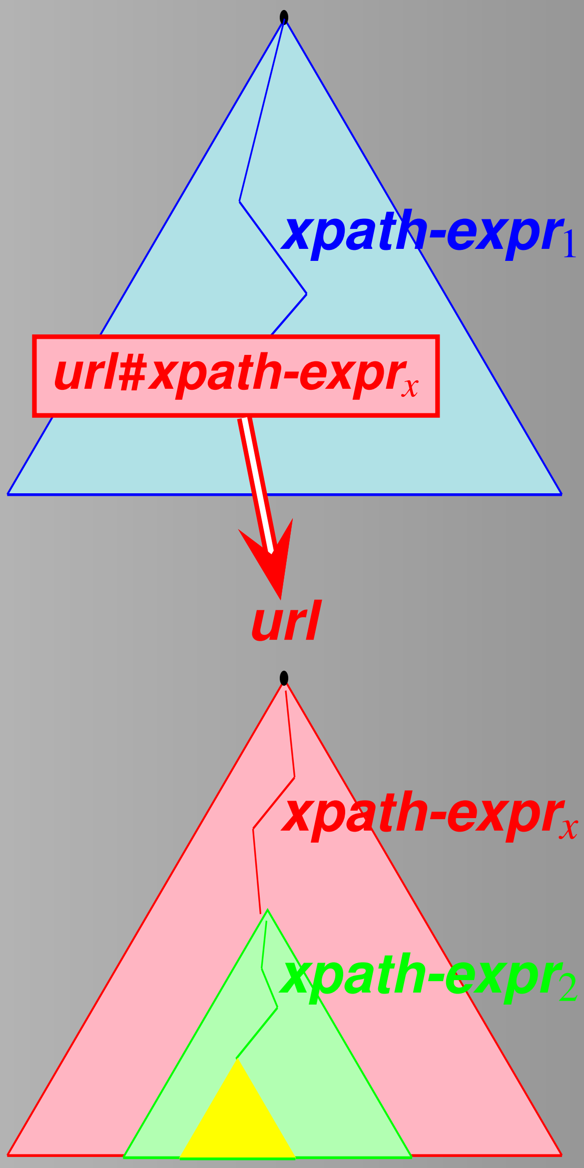
Switches

Evaluation: dbxlink:eval attribute

remote: submit *xpath-expr*$_x$ (parsing) or *xpath-expr*$_x$/*xpath-expr*$_2$ (answering a query), transfer the result.

distributed: submit *xpath-expr*$_x$ and transfer the result. Evaluate *xpath-expr*$_2$ locally.

local: requests target document and evaluate *xpath-expr*$_x$ (parsing) or *xpath-expr*$_x$/*xpath-expr*$_2$ (answering a query) locally



Switches

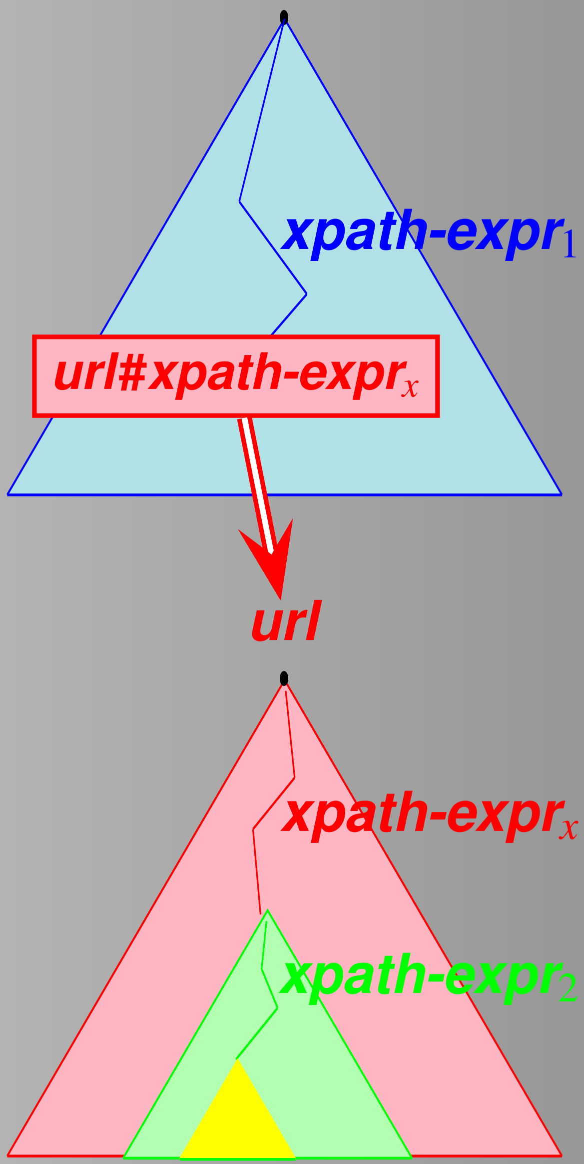
Answer Caching: dbxlink:cache attribute

complete: parses the *whole* target *document* and stores it (centrally) in the local XML database.

transparent: replaces the link by the result set of document(*url*)/$xpath\text{-}expr_x$

answer: stores the result of document(*url*)/$xpath\text{-}expr_x$/$xpath\text{-}expr_2$,

none: default

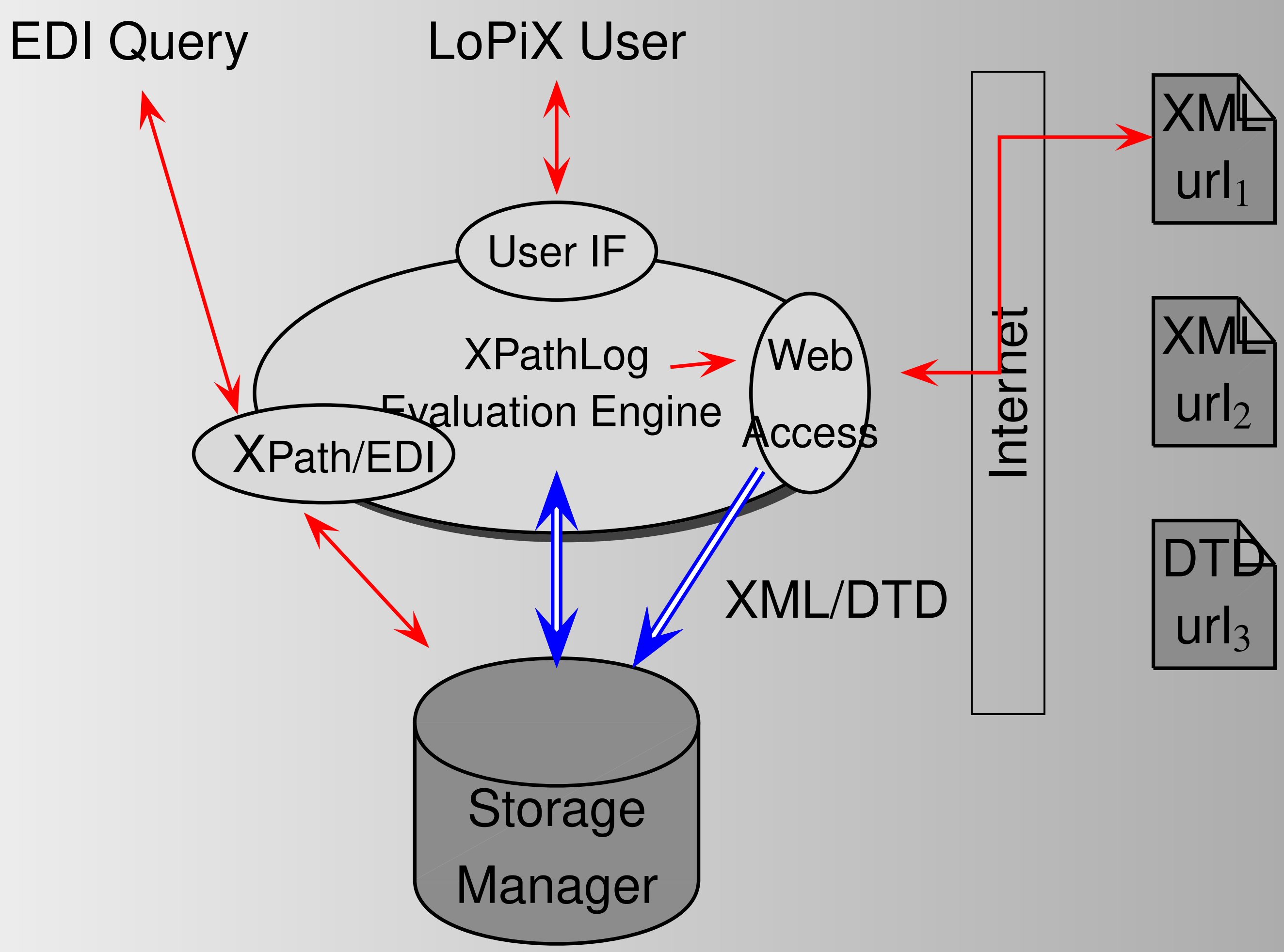


Runtime Behavior

- Overview table:
  For each query: first time, all following times
- Cornerstones
  - best query performance
  - minimized computation efforts; do computations remote
  - independence from server
- Constraints:
  services provides by the remote source

Experimental Base: LoPiX

- XPathLog as a language for XML querying and data integration.



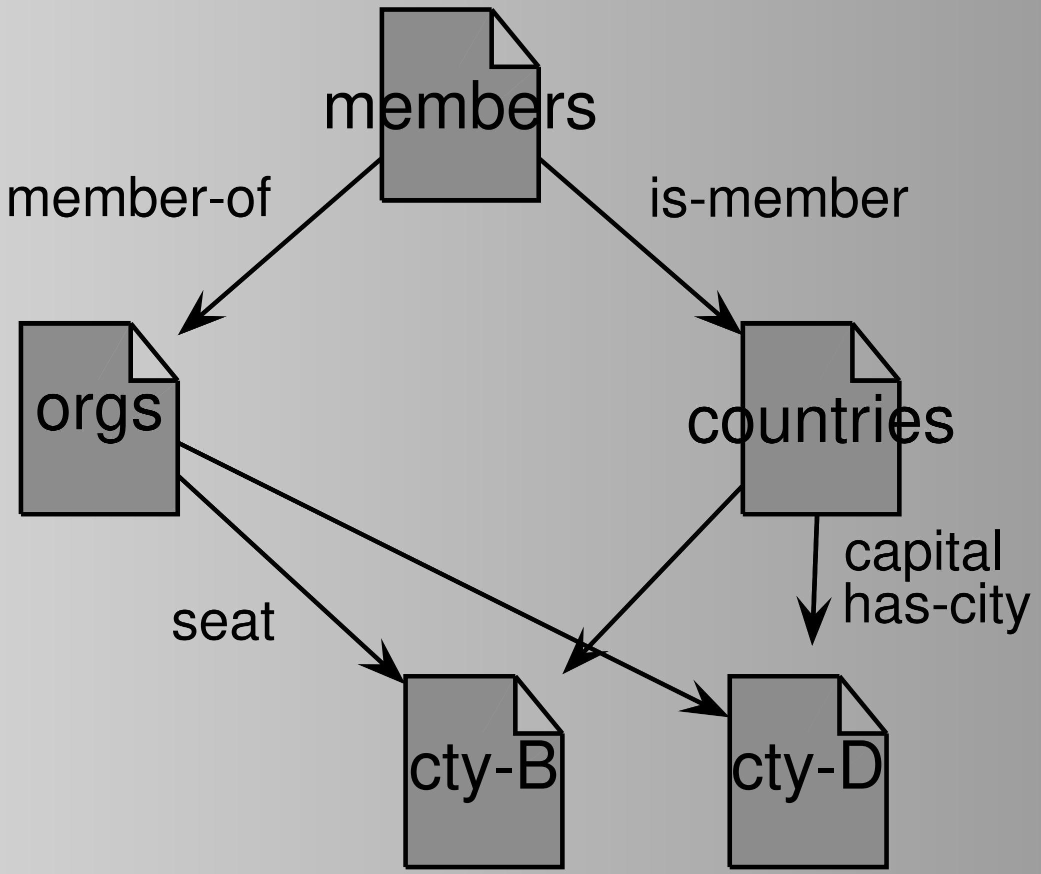
Conclusion

- XLink elements:
  - seamless integration of *view definitions* into the database
  - transparent semantics
- new aspects for XLink coming from query languages
  - add evaluation switches for querying (dbxlink namespace)
- Evaluation
  - prototypically implementable by XSLT rules
  - LoPiX
- Perspectives: XML-aware caches at intermediate network nodes, proxies etc.

Questions ??

Example Scenario: MONDIAL

- mondial-countries.xml
- mondial-cities-XX.xml
- mondial-organizations.xml
- mondial-memberships.xml



Out-of-Line-Links

<memberships>

<country code=“B” xlink:type=“locator”
xlink:href=“.../countries.xml#//country[@car_code=’B’]”/>

<country code=“D” xlink:type=“locator”
xlink:href=“.../countries.xml#//country[@car_code=’D’]”/>

<organization id=“org-EU” xlink:type=“locator”
xlink:href=“.../organizations.xml#//organization[@abbrev=’EU’]”/>

<organization id=“org-UN” xlink:type=“locator”
xlink:href=“.../organizations.xml#//organization[@abbrev=’UN’]”/>

<membership xlink:from=“B” xlink:to=“org-EU” xlink:type=“arc”
membership_type=“member”/>

<membership xlink:from=“B” xlink:to=“org-UN” xlink:type=“arc”
membership_type=“member”/>

</memberships>

Transparent Out-of-Line Links

```
<memberships>
    <!-- country xlink:type="locator"  xlink:href= Belgium -->
    <country id="B"   with attributes and contents of Belgium  />
    :
    <!-- organization xlink:type="locator"  xlink:href= EU -->
    <organization id="org-EU"   with attributes and contents of EU  />
    :
    <membership membership_type="member">
        <!-- xlink:from="B" xlink:to="org-EU"-->
        <country id="B"   with attributes and contents of Belgium />
        <organization id="org-EU"   with attributes and contents of EU  />
    </membership>
    :
</memberships>
```

Query:

//membership[organization/@abbrev="EU"]/country/name

Transparent Links: Query

"Select all names of organizations whose seats is the capital of one of its members."

```
FOR $ms IN document("mondial-memberships.xml")
                    //ms:membership,
      $org IN $ms/ms:organization,
                    <!-- dereferencing the arc element -->
      $abbrev IN $org/@org:abbrev,
      $seatname IN $org/org:seat/@cty:name,
                    <!-- dereferencing the simple link -->
      $capname IN $ms/ms:country/ctry:capital/@cty:name
                    <!-- dereferencing the arc element -->
                    <!-- dereferencing the simple link -->
WHERE $capname = $seatname
RETURN <result org = $abbrev/>
```