**XML 101 and Some XML Research from Database Perspective**

Dongwon Lee
Penn State / IST
Feb. 17, 2005

---

**Outline**

- Part I: (Quick) XML 101
- Part II: XML (Database) Research
  - XML Access Controls
  - XML Schema Languages
  - Web Services Compositions

---

**Mark-up Language**

- A notation for writing text with markup tags `<tag>`
- Tags indicate the structure of the text
- Tags have names and attributes
- Tags may enclose a part of the text
- Invented around 1970 by Charles F. Goldfarb (SGML)

```xml
<employee id="100">
  <fullname>John Doe</fullname>
</employee>
```

---

**HTML**

- HTML: Hyper-Text Markup Language
  - Invented by Tim Berners-Lee and Robert Caillau at CERN in 1991
  - What is hyper-text?
    - A document that contains links to other documents (and text, sound, images...)
    - Invented around 1945 by Vannevar Bush

---

**XML**

- What does XML stand for (GRE question)?
  1. X-rated Modular 3 Language
  2. eXtensible Standard ML Programming Language
  3. eXtensible UML (Unified Modeling Language) for the Web
  4. eXtensible Meta Language for the Web
  5. eXtensible Markup Language originating from SGML
  6. None of the above
XML: eXtensible Mark-up Language

- XML is a **framework** for defining mark-up languages:
  - XML was designed to **describe data, not format**
  - There is no fixed collection of markup tags
  - Allow **tailor-made markup** for any imaginable application domain
  - XML uses a schema language (e.g., DTD, XML-Schema) to formally describe the data.
  - XML separates **syntax** from **semantics** to provide a common framework for structuring information
- Web browser rendering semantics is separately defined by stylesheets

### HTML vs. XML

- **Need a stylesheet to define browser presentation semantics**
- Data/Format vs. Data/Format
- Browser

### A conceptual view of XML

- An XML document is an **ordered, labeled tree**
- **Character data** leaf nodes contain the actual data (text strings)
- **Elements** nodes are each labeled with
  - a name (often called the element type), and
  - a set of **attributes**, each consisting of a name and a value,
  - and can have child nodes
A concrete view of XML

- An XML document is a text with **mark-up tags** and other meta-information.
- Markup tags denote elements:
  ```
  ...<foo attr="val" ...>bar</foo>...
  ```
  - a matching **element end tag**
  - the contents "bar" of the element
  - an attribute with name attr and value val, enclosed by "or"
  - an **element start tag** with name foo
- An XML document must be **well-formed**:
  - start and end tags must match
  - element tags must be properly nested

Applications of XML

- CML: Chemical Markup Language
  ```
  <molecule id="METHANOL">
    <atomArray>
      <stringArray builtin="elementType">C O H H H H</stringArray>
      <floatArray builtin="x3" units="pm">-0.748 0.558 -1.293 -1.263 -0.699 0.716</floatArray>
    </atomArray>
  </molecule>
  ```
- There are +1000 new markup languages made by XML (eg, www.schema.net)

Database Perspective

- DB must support
  - Capture
  - Storage
  - Retrieval
  - Exchange
- XML originally as the language to **exchange data over Web**
  - Replacing EDI (Electronic Data Interchange)

Element vs. Attribute

- The same information can be captured by either Element or Attribute in XML

Applications of XML

- XML is a **meta-language** to create another languages; the main application of XML is making new languages
- XML is a **wire language** to hook up various building blocks (<tags>) to build new languages

RDBMS vs. XML Example

- **AFV** receives 100+ videos every week
- [IST210 HW#2, 2004] Build a DB to be able to answer the following queries:
  - Who sent which videos?
  - Show me all videos about Cat category
  - How many videos in a database since Jan 1, 2003?
  - Which is the video with the best rating for the 1st week of Jan?
  - Where does the sender James live? Phone? Gender?
  - How many videos does James send so far?
  - Show me all the ghost videos (ones without sender information)
ER Model

<table>
<thead>
<tr>
<th>VID</th>
<th>Category</th>
<th>Date</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Comedy</td>
<td>2005/1/1</td>
<td>S</td>
</tr>
<tr>
<td>200</td>
<td>Action</td>
<td>2005/1/10</td>
<td>4</td>
</tr>
<tr>
<td>300</td>
<td>SF</td>
<td>2004/12/31</td>
<td>S</td>
</tr>
</tbody>
</table>

ER => RDBMS

<table>
<thead>
<tr>
<th>VID</th>
<th>Category</th>
<th>Date</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Comedy</td>
<td>2005/1/1</td>
<td>S</td>
</tr>
<tr>
<td>200</td>
<td>Action</td>
<td>2005/1/10</td>
<td>4</td>
</tr>
<tr>
<td>300</td>
<td>SF</td>
<td>2004/12/31</td>
<td>S</td>
</tr>
</tbody>
</table>

RDBMS

<table>
<thead>
<tr>
<th>VID</th>
<th>Category</th>
<th>Date</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Comedy</td>
<td>2005/1/1</td>
<td>S</td>
</tr>
<tr>
<td>200</td>
<td>Action</td>
<td>2005/1/10</td>
<td>4</td>
</tr>
<tr>
<td>300</td>
<td>SF</td>
<td>2004/12/31</td>
<td>S</td>
</tr>
</tbody>
</table>

Changes #1

- 100+ videos => 1 million videos
- 100+ owners => 1 million owners
- VHS video => VHS, CD, DVD

Changes #2

- Arbitrary name formats for owners
  - Eg. J. Doe vs. Dr. “Jonny” John Jay Doe Jr
- 100+ different ways to capture owners' information
  - “100 E. Foster Ave #200, State College, PA, 16801” vs.
    - adr1=“100 E. Foster #210”, adr2=“State College, PA, 16801”
  - street=“100 E. Foster, #200”, city=“State College”, state=“PA”, zip=“16801”
- 100+ different video formats with varying properties => 1000+ attributes for Videos
**RDBMS: Finest Granularity**

<table>
<thead>
<tr>
<th>VID</th>
<th>Category</th>
<th>Date</th>
<th>Rating</th>
<th>Art</th>
<th>Att</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Comedy</td>
<td>2005/1/1</td>
<td>5</td>
<td>10</td>
<td>T1</td>
</tr>
<tr>
<td>200</td>
<td>Action</td>
<td>2005/11/10</td>
<td>4</td>
<td>20</td>
<td>T1</td>
</tr>
<tr>
<td>300</td>
<td>SF</td>
<td>2004/12/31</td>
<td>5</td>
<td>20</td>
<td>S20</td>
</tr>
</tbody>
</table>

**Gender**

<table>
<thead>
<tr>
<th>Name</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jonny John Doe Jr.</td>
<td>564-3456</td>
</tr>
</tbody>
</table>

**Address**

<table>
<thead>
<tr>
<th>Street</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 S. Beaver</td>
<td>State College, PA</td>
<td>16801</td>
<td></td>
</tr>
</tbody>
</table>

**Phone**

<table>
<thead>
<tr>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>564-3456</td>
</tr>
</tbody>
</table>

**Violation of 1NF**

**XML**

```xml
<VideoTable>
  <Video vid="100" category="comedy" date="2005/1/1" rating="5" att2="10" att1000="T1"/>
  <Video vid="200" category="action" date="2005/1/10" rating="4" att1="20"/>
  <Video vid="300" category="SF" date="2004/12/31" rating="5" att1000="520"/>
</VideoTable>
```

**RDBMS: Coarsest Granularity**

<table>
<thead>
<tr>
<th>VID</th>
<th>Category</th>
<th>Date</th>
<th>Rating</th>
<th>Art</th>
<th>Att</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Comedy</td>
<td>2005/1/1</td>
<td>5</td>
<td>10</td>
<td>T1</td>
</tr>
<tr>
<td>200</td>
<td>Action</td>
<td>2005/11/10</td>
<td>4</td>
<td>20</td>
<td>T1</td>
</tr>
<tr>
<td>300</td>
<td>SF</td>
<td>2004/12/31</td>
<td>5</td>
<td>20</td>
<td>S20</td>
</tr>
</tbody>
</table>

**Gender**

<table>
<thead>
<tr>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jenny</td>
</tr>
</tbody>
</table>

**Address**

<table>
<thead>
<tr>
<th>Street</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 S. Beaver</td>
<td>State College, PA</td>
<td>16801</td>
<td></td>
</tr>
</tbody>
</table>

**Phone**

<table>
<thead>
<tr>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>564-3456</td>
</tr>
</tbody>
</table>

**XML**

```xml
<OwnerTable>
  <Owner phone="564-3456" gender="F">
    <Name FN="Jenny"/>
    <Address>
      <Street>310 N. Atherton</Street>
      <City>State College</City>
      <State>PA</State>
      <Zip>16801</Zip>
    </Address>
  </Owner>
  <Owner phone="123-4567" gender="M">
    <Name Prefix="Dr." FN="Jonny" MN="Jay" LN="Doe" Suffix="Jr."/>
    <Address>
      <Str1>10 S. Beaver</Str1>
      <Str2>
        <State>PA</State>
      </Str2>
    </Address>
  </Owner>
</OwnerTable>
```

**RDBMS vs. XML**

- **Structured model**
- **Large scale data**
- **Limited semantics**
- Focus: how to handle large size data efficiently?

- **Unstructured or Semi-structured model**
- **Small to Medium scale data**
- **Flexible and rich semantics**
- Focus: how can one handle large number of small size data with various formats efficiently?
**Why is XML NOT important?**

- Some people believe that XML is NOT important
- The best answer that I have gotten so far is (from SIGMOD 2001 Panel Talk): XML is NOT important because ...
  - XML is a wire language
  - But, world is going for wireless
- What do you think?

**Why is XML important? (IMHO)**

- Technically, ... Not much; Just old simple tree model...
- Non-technically, ...
  - Hot ($$$)
  - The standard for representation of Web information
  - The real force of XML is generic languages and tools!
  - By building on XML, you get a massive (standard) infrastructure for free

**XML Usages...**

- [www.w3c.org](http://www.w3c.org)
- [www.oasis-open.org](http://www.oasis-open.org)
Outline

- Part I: (Quick) XML 101
- Part II: XML (Database) Research
  - XML Access Controls
  - XML Schema Languages
  - Web Services Compositions

Credits

- Joint work with Padmapriya Ayagari
- Wang-Chien Lee
- Cathy Li
- Peng Liu
- Robert Luo
- Prasenjit Mitra

- Of Ear, Fire, Fault and Flow: XML Access Control via XPath-based Views

- Bo Lu, Dongwon Lee, Wang-Chien Lee, Peng Liu, In 13th ACM Conf. On Information and Knowledge Management (CIKM), Washington DC, USA, November 2004

- A Flexible Framework for Architecting XML Access Control Enforcement

- Bo Lu, Dongwon Lee, Wang-Chien Lee, Peng Liu, In VLDB Workshop on Secure Data Management in a Connected World (SDM), Sydney, Australia, August 2004

- Supporting XML Security Models using Relational Databases: A Viable

- Dongwon Lee, Wang-Chien Lee, Peng Liu, In XML Database Symposium (XSym), Berlin, Germany, September 2003

Introduction

Relational Access Control ensures only authorized users can access only authorized portion of relational data

- Role-based security model (user << role)
- GRANT / REVOKE
- Can control table-level or column-level access

Eg,

- GRANT SELECT ON Foo TO dlee
- GRANT INSERT ON Bar(A, B) TO admin

Motivation

- Popular solution to enforce XML Access Control is to use “Materialized Views” (eg, [Dimiani, 2002; Bertino, 2002; Yu, 2002])
- Construct a view per role/user
- Once view is constructed, no more security check
- Space cost / Maintenance issue
- Others rely on the support of security feature of XML database [Cho; 2002]
- No XML databases have such features yet

Often, XML Access Control is represented as 5-tuple ACR

- (subject, object, action, +/-, RC/LC)
- Object is expressed by XPath

Eg, “Manager can read employee’s project-related information, but not their salaries”

(manger, //employee/proj, read, +, RC)
(manger, //employee/@salary, read, -, LC)

ACR (Access Control Rule)
Motivation

- Neither approach is fully satisfactory
- Our Approach is
  - Framework-based: devise and compare various approaches
  - Practical solution: can work with off-the-shelf XML database engine (ie, Vanilla XDBMS)

Framework

- XML data (document).
- Stored in XML database
- Describes the information that users want in XPath
  - Query has the same security role as the user who issues it
- Each ACR describes the access control policy of a role.
  - Objects are specified in XPath

Scenarios

- Conventional
- View-based
- Pre-processing
- Post-processing
- Miscellaneous

Post-Processing

- Intermediate answers are computed as usual
- Then, ACR prunes out unsafe answers
- Suitable for role-based data delivery model, where the same data is delivered to different roles.
- Can be implemented by XML data filtering package (eg, YFilter [Diao, 2003])
Pre-Processing: Primitive

- View Query and ACR as two constraints to satisfy
  - Q and + ACR: Q ∩ ACR ⊆ Q'
  - Q and – ACR: Q – ACR ⊆ Q'
- Then, Q' is passed to regular XML engine that can handle XPath with set operator
  - Easy to implement
  - Performance is highly dependent on the capability of underlying XML engine (how it handles set operators, etc)

Pre-Processing: QFilter

- Primitive Pre-Processing satisfies our two goals
  - Non-view based
  - Independent on underlying XML engine
- But, the re-written query Q' is not the most efficient form

QFilter Example

- R1: /site/categories/*
- R2: /site/regions/*/item/location
- R3: /site/regions/*/item/quantity
- R4: /site/regions/*/item/name
- R5: /site/regions/*/item/description
- R6: /site/people/person/name
- R7: /site/people/person/address/*
- R8: /site/people/person/emailaddress

Pre-Processing: QFilter

- Idea of QFilter: Improve Q’ further for better performance
  - Contained Case (Q ⊆ ACR ⊆ Q)
    - Q ∩ ACR ⊆ Q
  - Disjoint Case (Q ∩ ACR ⊆ {})
    - Q ∩ ACR ⊆ {}
  - Overlapping Case (Q ∩ ACR ⊆ Q’)
    - Q ∩ ACR ⊆ Q’

QFilter captures ACR as NFA (Non-deterministic Finite Automata)

Given Q, quickly determine if it is contained, disjoint or overlapping by traversing NFA
When it’s overlapping case, Q’ is generated

Cannot handle general case of XPath

XPath containment is:
- /, //, [, ], *, P [Wood, 2001] ⊆ QFilter supports this
- NOT, <: undecidable [Neven, 2003]
State Transition Map

Q: /site/categories/NW/item

Q: /site/top/item

Q: /site/*/person/name

QFilter with Predicate Handling

R9:/site/regions/*/item[description]/name
Q: /site/regions/item[quantity]/name

Q'=/site/regions/item[quantity][description]/name

**QFilter Discussion**

Theorem: Q' generated by QFilter never returns unauthorized answers to unauthorized users for XPath with /, //, *, []

- QFilter construction: O(|ACR|)
- QFilter execution
  - No *: O(|Q|)
  - *: O(|NFA|)
  - //: O(|Q| * i child for i-th //)
- Worst case only occurs for a query "'/…//*//*…//*'"

**Evaluation Plan**

- No Access Control
- Pre-Processing Scenario
  - Primitive
  - QFilter
  - Static Analysis [Murata; 2003]
- Post-Processing Scenario
  - Figure 1: Ways to support XML access control without using security features of DBMS

**Set-Up**

- XMark / Galax / YFilter
- XML data (1.5 MB)
- ACR
  - 550 Synthetic rules
  - 10 User-defined rules
- Q
  - 7 categories based on /, //, *, []
  - 100 Synthetic queries

**QFilter Performance**

- Query Accepted
- Query Denied
- Query Rewritten

- Number of access control rules

- QFilter execution time (ms)
Comparison among Scenarios

- Accepted
- Denied
- Rewritten

QFilter > No Access Control ??

Two Pre-Processing Methods

- QFilter Approach
- Static Analysis Initialization

Two Pre-Processing Methods

- Security check time (ms)

View-based XML Access Controls

- High maintenance cost when update occurs
- For an organization, frequent updates to ACR are possible
  - Rules are being added/removed
  - For each rule change, new view must be materialized
- Metric
  - Amount of data from DB to view
  - Time/Storage

QFilter Applications

- QFilter is in essence a black box that
  - Consists of Constraints C
  - Takes input Q
  - Outputs filtered result Q'

Scenarios Again

<table>
<thead>
<tr>
<th>Conventional</th>
<th>Query</th>
<th>ACR</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>View-based</td>
<td>Query</td>
<td>ACR</td>
<td>Data</td>
</tr>
<tr>
<td>Pre-processing</td>
<td>Query</td>
<td>ACR</td>
<td>Data</td>
</tr>
<tr>
<td>Post-processing</td>
<td>Query</td>
<td>Data</td>
<td>ACR</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>Query</td>
<td>Data</td>
<td>ACR</td>
</tr>
</tbody>
</table>
**View-based XML Access Controls**

- T1: +R1, +R2, +R3
- T2: +R3 is removed

**Secure P2P Overlay Network**

- XML documents are stored on peer nodes (with possible duplication)
- Different peer nodes have different access controls
- When a node $N$ gets a query $Q$, if $N$ cannot handle $Q$, it has to forward $Q$ to proper peers who have legitimate access controls.

**Secure P2P Overlay Network**

- XML documents are stored on peer nodes (with possible duplication)
- Different peer nodes have different access controls
- When a node $N$ gets a query $Q$, if $N$ cannot handle $Q$, it has to forward $Q$ to proper peers who have legitimate access controls.

**Outline**

- Part I: (Quick) XML 101
- Part II: XML (Database) Research
  - XML Access Controls
  - XML Schema Languages
  - Web Services Compositions

**XML Schema Languages**

- A **schema** is a formal description of structures and constraints
  - Relational schema consists of table, column, constraints, etc.
- A **schema language** is a formal language for expressing schemas.
- **Schema processing**: Given an XML document and a schema, a schema processor checks for validity, i.e. that the document conforms to the schema requirements

**Schema Validation**

- XML Schema
- XML Document
- Valid
- Invalid
- Debugging
Which Schema Language?
- Many proposals competing for acceptance
- W3C Proposals: DTD, XML Data, DCD, DDML, SOX, XML-Schema, ...
- Non-W3C Proposals: Assertion Grammars, Schematron, DSD, TRELX, RELAX, XDuce, RELAX-NG, ...
- Different applications have different needs, and can use different schema languages

Research Question
- Q1: Is an XML schema language A more powerful than B?
- Q2: Can an XML schema language A express a constraint F?
- Q3: Can the union/intersection/difference of two XML schemas be captured by an XML schema language A?

Expressive Power (content model)
- DTD
- XML-Schema
- XDuce, RELAX-NG

Closure (content model)
- DTD
- XML-Schema
- XDuce, RELAX-NG

Outline
- Part I: (Quick) XML 101
- Part II: XML (Database) Research
  - XML Access Controls
  - XML Schema Languages
  - Web Services Compositions

Web Services
- Web Services = RPC + XML + More
- New profit opportunities for vendors
  - Can turn existing functions into distributed, heterogeneous “services”
  - Global-scale virtual market
  - Agent-based autonomous discovery, composition, optimization, negotiation, auction, etc
Research Question

- Q1: Find an efficient way to determine if a web service A can invoke a web service B
- Q2: Given a large number of web services (100,000+) and initial/goal states, find the optimal way to compose N web services to satisfy the goal state

- Solution: BF*: Web Services Discovery and Composition as Graph Search Problem. Seog-Chan Oh, Byung-Won On, Eric J. Larson, Dongwon Lee, in IEEE Int'l Conf. on e-Technology, e-Commerce and e-Service (EEE), Hong Kong, China, March 2005

Outline

- Part I: (Quick) XML 101
- Part II: XML (Database) Research
  - XML Access Controls
  - XML Schema Languages
  - Web Services Compositions

Conclusion

- XML has many interesting and challenging research problems
- As your thesis/dissertation topic
  - Step 1: pick your favorite subject
  - Step 2: throw XML into the picture
  - Step 3: work out the rest...
- Hope to see more students get interested in XML research

Advisement: NIKE

Nittany Information Knowledge wEb Research Group

http://nike.psu.edu

Slide is available at
- http://nike.psu.edu => presentations section

Thank You