Web Data and Declarative Programming

Jesús M. Almendros Jiménez

Universidad de Almería.
jalmen@ual.es

March 2, 2010
Table of contents

1 Introduction

2 Web Data
   - XML
   - RDF
   - OWL

3 Querying and Reasoning with Web Data
   - XQuery
   - SPARQL

4 Contributions
   - XPath
   - XQuery
   - Ontology Reasoning
   - XQuery with Ontology Reasoning

5 Current Research
Introduction

Research Lines

- **Data on the Web**
  - XML
  - Ontology languages:
    - RDF
    - OWL
    - Description Logic

- **Querying and Reasoning with Web Data**

- **Database Programming Languages**
  - XPath and XQuery
  - SPARQL

- **Rule Based Languages**
  - Prolog
  - SWRL
Contributions

- XML and Logic Programming
- XPath and Logic Programming
- XQuery and Logic Programming
- Ontology Reasoning with Prolog
- XQuery and Ontology Reasoning
Semantic Web

Figure: Tim Berners-Lee Semantic Web Stack
Web Data

- Representation of Data: XML, RDF, OWL
- Querying of XML
- Reasoning with RDF and OWL
- Description Logic
- Efficient storing and retrieval
- Complete reasoning
XML

- Tree Based Representation
- Books, People, Papers, ...
- Year, Identifier, Link to Web Page, Impact Factor, ...
- Name, Title, Authors, Price, ...
- More than one author, Missing Price, Mix of Books and Papers
- Non relational database
Example

<books>
  <book year="2003">
    <author>Abiteboul</author>
    <author>Buneman</author>
    <author>Suciu</author>
    <title>Data on the Web</title>
    <review>A <em>fine</em> book.</review>
  </book>
  <book year="2002">
    <author>Buneman</author>
    <title>XML in Scotland</title>
    <review><em>The best ever!</em></review>
  </book>
</books>
Where?

- Exported from RDBMS
- Web resources (DBLP, http://dblp.uni-trier.de/xml/dblp.xml)
- W3C recommendation
RDF

RDF/RDFS

- Graph based Representation
- (Subject, Property, Object)
- Metadata
  - Subclass: “An student is a person”
  - Subproperty: “A girlfriend is a friend”
  - Type: “Jesus is a teacher”
  - Domain and Range: “The range and domain of friend is a person”
RDF

```xml
<rdf:Description rdf:about="http://www.amazon.com/12345">
  <author rdf:resource="Abiteboul"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.amazon.com/12345">
  <author rdf:resource="Suciu"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.amazon.com/12345">
  <author rdf:resource="Buneman"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.amazon.com/12345">
  <title rdf:resource="Data on the Web"/>
</rdf:Description>
```
Introduction
Web Data
Querying and Reasoning with Web Data
Contributions
Current Research

RDF

RDFS

```xml
<rdf:Property rdf:ID="author"/>
<rdf:Property rdf:ID="title"/>
<rdf:Class rdf:ID="paper"/>
<rdf:Class rdf:ID="manuscript"/>
<rdf:Description rdf:about="#paper">
  <rdfs:subClassOf rdf:resource="#manuscript"/>
</rdf:Description>
<rdf:Description rdf:about="Abiteboul">
  <rdf:type rdf:resource="#man"/>
</rdf:Description>
<rdf:Description rdf:about="#girlfriend">
  <rdfs:subPropertyOf rdf:resource="#friend"/>
</rdf:Description>
```

Jesús M. Almendros Jiménez
Web Data and Declarative Programming
**OWL**

- Extension of RDF
- Based on Description Logic
- RDF-based representation: Graph based representation
- Metadata:
  - **Subclass**, **Subproperty**, **Type**, **Domain** and **Range** from RDF(S)
  - **Union**, **intersection**
  - **Symmetric** (Friend), **Inverse** (Husband and Wife) and **Transitive Properties** (Friend, Facebook)
  - Class and property **equivalence** and **Complex subclass relationships**: married are husband or wife of persons
  - More complex relationships, more complex description logic: OWL DL, OWL Lite, OWL Full
OWL

```xml
<owl:Class ID="Husband_or_Wife">
<owl:unionOf rdf:parseType="Collection">
<owl:Restriction>
<owl:onProperty rdf:resource="#husband"/>
<owl:someValuesFrom rdf:resource="#Person"/>
</owl:Restriction>
<owl:Restriction>
<owl:onProperty rdf:resource="#wife"/>
<owl:someValuesFrom rdf:resource="#Person"/>
</owl:Restriction>
</owl:unionOf>
</owl:Class>
<owl:Class rdf:about="#Husband_or_Wife">
<rdfs:subClassOf rdf:resource="#Married"/>
</owl:Class>
<owl:Class rdf:about="#Husband_or_Wife">
<owl:inverseOf rdf:resource="#wife"/>
</owl:inverseOf>
</owl:Class>
```
XQuery

- Typed Functional Language for XML Querying
- For Let Order By Where Return, ‘FLOWR” expressions
- Join of Multiple Documents
- Sublanguage: XPath: Querying the tree structure of XML
XQuery

XPath

- /books/book/author
- /books/book[@year=2002]/author
- /books/book/author/..
- /books/book/*
XQuery

<result>
for $Book in doc('ex.xml')/bib/book return
let $Year := $Book/@year
where $Year < 1995 return
let $Title := $Book/title return
<mybook> { $Year } { $Title } </mybook>
</result>
XQuery

```
<result>
  <mybook year="1994">
    <title>TCP/IP Illustrated</title>
  </mybook>
  <mybook year="1992">
    <title>Advanced Programming in the Unix environment</title>
  </mybook>
</result>
```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX ns: <http://example.org/ns#>
SELECT ?title ?price
WHERE { ?x ns:price ?price .
FILTER (?price ≤ 30.5)
?x dc:title ?title . }
XML and XPath

- Representation of XML documents in Prolog
  - **Facts** for leaves of the XML tree, **Rules** for structure of the XML tree (TPLP’08)
  - A **Prolog term** for XML tree (XSym’09, SWI-Prolog)

- Implementation of XPath in Prolog
  - Facts/Rules: **Program specialization** of Prolog rules (TPLP’08)
  - Prolog term: **Traversal** of the Prolog term (XSym’09)
XPath

Facts and Rules for the XML Tree

**Rules (Schema):**

books(bookstype(Book, []), NBooks,1,Doc) :-
  book(Book, [NBook|NBooks],2,Doc).

book(booktype(Author, Title, Review, [year=Year]),NBook ,2,Doc) :-
  author(Author, [NAu|NBook],3,Doc),
  title(Title, [NTitle|NBook],3,Doc),
  review(Review, [NRe|NBook],3,Doc),
  year(Year, NBook,3,Doc).

review(reviewtype(Un,Em,[]),NReview,3,Doc):-
  unlabeled(Un,[NUn|NReview],4,Doc),
  em(Em,[NEm|NReview],4,Doc).

review(reviewtype(Em,[]),NReview,3,Doc):-
  em(Em,[NEm|NReview],5,Doc).

em(emtype(Unlabeled,Em,[]),NEms,5,Doc) :-
  unlabeled(Unlabeled,[NUn|NEms],6,Doc),
  em(Em, [NEm|NEms],6,Doc).
### XPath

#### Facts and Rules for the XML Tree

<table>
<thead>
<tr>
<th>Facts (Document):</th>
</tr>
</thead>
<tbody>
<tr>
<td>year('2003', [1, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>author('Abiteboul', [1, 1, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>author('Buneman', [2,1, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>author('Suciu', [3,1,1], 3, 'books.xml').</td>
</tr>
<tr>
<td>title('Data on the Web', [4, 1, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>unlabeled('A', [1, 5, 1, 1], 4, 'books.xml').</td>
</tr>
<tr>
<td>em('fine', [2, 5, 1, 1], 4, 'books.xml').</td>
</tr>
<tr>
<td>unlabeled('book.', [3, 5, 1, 1], 4, 'books.xml').</td>
</tr>
<tr>
<td>year('2002', [2, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>author('Buneman', [1, 2, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>title('XML in Scotland', [2, 2, 1], 3, 'books.xml').</td>
</tr>
<tr>
<td>unlabeled('The', [1, 1, 3, 2, 1], 6, 'books.xml').</td>
</tr>
<tr>
<td>em('best', [2, 1, 3, 2, 1], 6, 'books.xml').</td>
</tr>
<tr>
<td>unlabeled('ever!', [3, 1, 3, 2, 1], 6, 'books.xml').</td>
</tr>
</tbody>
</table>
Prolog term of the XML tree

```prolog
[element(bib, []),
[element(book, [year=1994],
[element(title, [], [TCP/IP Illustrated]),
[element(author, [], [element(last, [], [Stevens]),
[element(first, [], [W.])]),
[element(publisher, [], [Addison-Wesley]),
[element(price, [], [65.95]) ]),
[element(book, [year=1992],
[element(title, [], [Advanced Programming in the Unix environment]),
[element(author, [], [element(last, [], [Stevens]),
[element(first, [], [W.])]),
[element(publisher, [], [Addison-Wesley]),
[element(price, [], [65.95]) ]),
[element(book, [year=2000],
... ]])
```
 XPath

Query

/books/book[@year=2002]/author

Specialized Program

book(booktype(Author, [year=Year]),NBook ,2,Doc) :-
author(Author, [NAu|NBook],3,Doc),
year(Year, NBook,3,Doc).

Goal

?- book(booktype(Author, [year=2002]),NBook ,2,"ex.xml").
XPath

XPath Predicate

\[
\text{xpath([Label], [element(Label, Attr, Sublabel)] | Relement),}
\]
\[
\text{element(Label, Attr, Sublabel)] | Relement2)):!},
\]
\[
\text{xpath([Label], Relement, Relement2).}
\]
\[
\text{xpath([Label], [ - Relement], Relement2)):!}, \text{xpath([Label], Relement, Relement2).}
\]
\[
\ldots
\]

Goal

\[
?- \text{xpath([} \text{books, book, author}], [element(books, ...)], Author).
\]
XQuery

- XQuery implementation based on facts and rules representation (WLP’07)
- XQuery implementation based on Prolog term representation (XSym’09)
XQuery

for $book in document ('books.xml')/books/book
return let $year := $book/@year
where $year<2003
return <mybook>{$year, $book/title}</mybook>

Encoding of XQuery

mybook(mybooktype(Title,[Year]),[Node],[Type],[Doc]):- 
join(Title,Year,Node,Type,Doc).

join(Title,Year,[Node],[Type],[Doc]):-
  vbook(Title,Year,Node,Type,Doc),
  constraints(vbook(Title,Year)).

vbook(Title,Year,[Node,Node],[TTTtitle,TTYear],’’books.xml’’):-
  title(Ttitle,[NTtitle|Node],TTtitle,’’books.xml’’),
  year(Year,Node,TTYear,’’books.xml’’).
XQuery

XQuery

```xml
<result>
  for $Book in doc('ex.xml')/bib/book return
  let $Year := $Book/@year
  where $Year < 1995 return
  <mybook> { $Year } { $Book/title } </mybook>
</result>
```

Encoding of XQuery

```prolog
xquery([element(result, [], Result)], 1) :- xquery(Result, 2).
xquery(MyBooks, 2) :- findall(MyBook, xquery([MyBook], 3), MyBooks).
xquery([element(mybook, [], YearTitle)], 3) :- xquery(YearTitle, 6).
xquery(YearTitle, 6) :- findall(Year, xquery([Year], 7), Years),
                  findall(Title, xquery(Title, 8), Titles),
                  combine([Years, Titles], YearsTitles),
                  member(YearTitle, YearsTitles).
...
Ontology Reasoning

**RDF(S) and OWL**

- RDF/RDF(S) Entailment
- OWL Reasoning: Description Logic
- Complete reasoning
- SWI-Prolog: RDF library
- Description Logic Programming
Ontology Reasoning

**RDF(S) and OWL**

- RDF Entailment: Prolog Rules (WWv’07)
- OWL Reasoning: Prolog Rules (WWv’09)
- Inference Calculus and Complete Reasoning (WWv’09)
- **triple** predicate in Prolog
Ontology Reasoning

Description Logic

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C \sqsubset D$</td>
<td>(rdfs:subClassof)</td>
</tr>
<tr>
<td>$E \equiv F$</td>
<td>(owl:equivalentClass)</td>
</tr>
<tr>
<td>$P \sqsubset Q$</td>
<td>(rdfs:subPropertyOf)</td>
</tr>
<tr>
<td>$P \equiv Q$</td>
<td>(owl:equivalentProperty)</td>
</tr>
<tr>
<td>$P \equiv Q^-$</td>
<td>(owl:inverseOf)</td>
</tr>
<tr>
<td>$P^+ \sqsubseteq P$</td>
<td>(owl:SymmetricProperty)</td>
</tr>
<tr>
<td>$T \sqsubseteq \forall P^- . D$</td>
<td>(rdfs:domain)</td>
</tr>
<tr>
<td>$T \sqsubseteq \forall P . D$</td>
<td>(rdfs:range)</td>
</tr>
<tr>
<td>$P(A, B)$</td>
<td>(property filler)</td>
</tr>
<tr>
<td>$C(A)$</td>
<td>(individual assertion)</td>
</tr>
</tbody>
</table>
## Description Logic

### TBox

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>Man ⊑ Person</td>
</tr>
<tr>
<td>(3)</td>
<td>Person ⊓ ∃author_of.Manuscript ⊑ Writer</td>
</tr>
<tr>
<td>(5)</td>
<td>Book ⊓ ∃topic.{ &quot;XML&quot;} ⊑ XMLbook</td>
</tr>
<tr>
<td>(2)</td>
<td>Woman ⊑ Person</td>
</tr>
<tr>
<td>(4)</td>
<td>Paper ⊔ Book ⊑ Manuscript</td>
</tr>
<tr>
<td>(6)</td>
<td>Manuscript ⊓ ∃reviewed_by.Person ⊑ Reviewed</td>
</tr>
<tr>
<td>(7)</td>
<td>Manuscript ⊑ ∀rating.Score</td>
</tr>
<tr>
<td>(9)</td>
<td>author_of ≡ writes</td>
</tr>
<tr>
<td>(11)</td>
<td>authored_by ≡ author_of⁻</td>
</tr>
<tr>
<td>(13)</td>
<td>T ⊑ ∀ author_of⁻. Person</td>
</tr>
<tr>
<td>(15)</td>
<td>T ⊑ ∀ reviewed_by⁻. Manuscript</td>
</tr>
</tbody>
</table>

### ABox

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>Man(“Abiteboul”)</td>
</tr>
<tr>
<td>(2)</td>
<td>Man(“Buneman”)</td>
</tr>
<tr>
<td>(4)</td>
<td>Book(“Data on the Web”)</td>
</tr>
<tr>
<td>(6)</td>
<td>Paper(“Growing XQuery”)</td>
</tr>
<tr>
<td>(8)</td>
<td>author_of(“Abiteboul”, “Data on the Web”)</td>
</tr>
<tr>
<td>(3)</td>
<td>Man(“Suciu”)</td>
</tr>
<tr>
<td>(5)</td>
<td>Book(“XML in Scottland”)</td>
</tr>
<tr>
<td>(7)</td>
<td>Person(“Anonymous”)</td>
</tr>
<tr>
<td>(9)</td>
<td>authored_by(“Data on the Web”, “Buneman”)</td>
</tr>
<tr>
<td>(10)</td>
<td>author_of(“Suciu”, “Data on the Web”)</td>
</tr>
<tr>
<td>(11)</td>
<td>author_of(“Buneman”, “XML in Scottland”)</td>
</tr>
<tr>
<td>(12)</td>
<td>writes(“Simeon”, “Growing XQuery”)</td>
</tr>
<tr>
<td>(13)</td>
<td>reviewed_by(“Data on the Web”, “Anonymous”)</td>
</tr>
</tbody>
</table>
## Inference Calculus

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Inference</th>
<th>$\not\vdash_{O} \ E \equiv \ E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Eq1)</td>
<td>$E \equiv F, F \equiv G$</td>
<td>$\vdash_{O} E \equiv G$</td>
</tr>
<tr>
<td>(Eq2)</td>
<td>$E \equiv F$</td>
<td>$\vdash_{O} F \equiv E$</td>
</tr>
<tr>
<td>(Eq3)</td>
<td>$C \sqsupset D, D \sqsubset C$</td>
<td>$\vdash_{O} C \equiv D$</td>
</tr>
<tr>
<td>(Sub1)</td>
<td>$E \equiv F$</td>
<td>$\vdash_{O} E \sqsubset F$</td>
</tr>
<tr>
<td>(Sub2)</td>
<td>$C \sqsupset D, D \sqsubset E$</td>
<td>$\vdash_{O} C \equiv E$</td>
</tr>
<tr>
<td>(Sub3)</td>
<td>$C \sqcup D \sqsubset E$</td>
<td>$\vdash_{O} C \equiv E$</td>
</tr>
<tr>
<td>(Sub4)</td>
<td>$E \sqsubset C \sqcap D$</td>
<td>$\vdash_{O} C \equiv E$</td>
</tr>
<tr>
<td>(Sub5)</td>
<td>$C_{1} \sqcap C_{2} \sqsubset D, E \sqsubset C_{1}$</td>
<td>$\vdash_{O} E \sqcap C_{2} \sqsubset D$</td>
</tr>
<tr>
<td>(Sub6)</td>
<td>$C_{1} \sqcup C_{2} \sqsubset D, E \sqsubset C_{1}$</td>
<td>$\vdash_{O} E \sqcup C_{2} \sqsubset D$</td>
</tr>
<tr>
<td>(Sub7)</td>
<td>$C \sqsubset D_{1} \sqcap D_{2}, D_{1} \sqsubset E$</td>
<td>$\vdash_{O} C \sqsubset E \sqcap D_{2}$</td>
</tr>
<tr>
<td>(Sub8)</td>
<td>$\exists P.{O} \sqsubset D, Q \sqsubset P$</td>
<td>$\vdash_{O} \exists Q.{O} \sqsubset D$</td>
</tr>
<tr>
<td>(Sub9)</td>
<td>$\exists P.C \sqsubset D, Q \sqsubset P$</td>
<td>$\vdash_{O} \exists Q.C \sqsubset D$</td>
</tr>
<tr>
<td>(Sub10)</td>
<td>$\exists P.C \sqsubset D, E \sqsubset C$</td>
<td>$\vdash_{O} \exists P.E \sqsubset D$</td>
</tr>
<tr>
<td>(Sub11)</td>
<td>$C \sqsubset \exists P.{O}, P \sqsubset Q$</td>
<td>$\vdash_{O} C \sqsubset \exists Q.{O}$</td>
</tr>
<tr>
<td>(Sub12)</td>
<td>$C \sqsubset \forall P.D, Q \sqsubset P$</td>
<td>$\vdash_{O} C \sqsubset \forall Q.D$</td>
</tr>
<tr>
<td>(Sub13)</td>
<td>$C \sqsubset \forall P.D, D \sqsubset E$</td>
<td>$\vdash_{O} C \sqsubset \forall P.E$</td>
</tr>
</tbody>
</table>
### Prolog Rules

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Prolog Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Eq1</strong></td>
<td><code>triple(E, owl:equivalentClass, E) : - class(E).</code></td>
</tr>
<tr>
<td><strong>Eq2</strong></td>
<td><code>triple(E, owl:equivalentClass, G) : - triple(E, owl:equivalentClass, F), triple(F, owl:equivalentClass, G).</code></td>
</tr>
<tr>
<td><strong>Eq3</strong></td>
<td><code>triple(E, owl:equivalentClass, F) : - triple(F, owl:equivalentClass, E).</code></td>
</tr>
<tr>
<td><strong>Eq4</strong></td>
<td><code>triple(E, owl:equivalentClass, F) : - triple(E, rdfs:subClassOf, F), triple(F, rdfs:subClassOf, E).</code></td>
</tr>
<tr>
<td><strong>Sub1</strong></td>
<td><code>triple(E, rdfs:subClassOf, F) : - triple(E, owl:equivalentClass, F), triple(D, rdfs:subClassOf, E).</code></td>
</tr>
<tr>
<td><strong>Sub2</strong></td>
<td><code>triple(C, rdfs:subClassOf, E) : - triple(C, rdfs:subClassOf, D), triple(D, rdfs:subClassOf, E).</code></td>
</tr>
<tr>
<td><strong>Sub3</strong></td>
<td><code>triple(D, rdfs:subClassOf, E) : - triple(union(U), rdfs:subClassOf, E), member(D, U).</code></td>
</tr>
<tr>
<td><strong>Sub4</strong></td>
<td><code>triple(E, rdfs:subClassOf, C) : - triple(E, rdfs:subClassOf, inter(I)), member(C, I).</code></td>
</tr>
<tr>
<td><strong>Sub5</strong></td>
<td><code>triple(inter(I2), rdfs:subClassOf, D) : - triple(inter(I1), rdfs:subClassOf, D), member(C, I1), triple(E, rdfs:subClassOf, C), replace(I1, C, E, I2).</code></td>
</tr>
<tr>
<td><strong>Sub6</strong></td>
<td><code>triple(union(U2), rdfs:subClassOf, D) : - triple(union(U1), rdfs:subClassOf, D), member(C, U1), triple(E, rdfs:subClassOf, C), replace(U1, C, E, U2).</code></td>
</tr>
<tr>
<td><strong>Sub7</strong></td>
<td><code>triple(C, rdfs:subClassOf, inter(I2)) : - triple(C, rdfs:subClassOf, inter(I1)), member(D, I1), triple(D, rdfs:subClassOf, E), replace(I1, D, E, I2).</code></td>
</tr>
<tr>
<td><strong>Sub8</strong></td>
<td><code>triple(hasvalue(Q, O), rdfs:subClassOf, D) : - triple(Q, owl:subPropertyOf, P), triple(hasvalue(P, O), rdfs:subClassOf, D).</code></td>
</tr>
</tbody>
</table>
XQuery with RDF(S) and OWL

- XQuery for Ontology Querying and Reasoning [WWv’07, PLAN-X’09, INAP’09]
- RDF(S) and OWL reasoning
- Built-in boolean functions for Ontology Reasoning
- XML/RDF(S)/OWL as input and as output
- Prolog-based XQuery implementation
XQuery extension

```xml
<list>
  { 
  for ($Writer,$Property,$Manus) in owldocument(‘ex.owl’) return 
    for ($Manus,$Property2,$Type) in owldocument(‘ex.owl’) 
    where rdfs:subPropertyOf($Property,writes) 
    and $Property2=rdf:typeOf and rdfs:subClassOf($Type,reviewed) return 
    <item>
      <manuscript> {$Manus } </manuscript>
      <writer> {$Writer } </writer>
    </item>
  }
</list>
```
Prolog-based XQuery implementation

```
list(listtype(Item,[]),NL,1,Doc):- item(Item,[NItem|N],2,Doc).
item(itemtype(manuscripttype(Manus,[]),
  writertype(Writer,[]),[]),NItem,2,'result.xml'):-
  join(Manus,Writer,NItem).
join(Manuscript,Writer,[NM,[1]]):-
  triple(Writer,Property,Manuscript,NM,'ex.owl'),
  triple(Manuscript,Property2,Type,_,'ex.owl'),
  triple(Property,rdfs:subPropertyOf,writes,_,'ex.owl'),
  Property2=rdf:typeOf,
  triple(Type,rdfs:subClassOf,reviewed,_,'ex.owl').
```
Current Research

- XQuery implementation
- XQuery extension in XQuery
- OWL extensions: More complex description logic
- GML: Geographic Markup Language: XML for Geographic Data
- Ontologies for Geographic Data
- GQuery? GPath?