Web Data and Declarative Programming

Jesús M. Almendros Jiménez

Universidad de Almería.
jalmen@ual.es

March 2, 2010

Table of contents

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

Introduction

Contributions

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

Semantic Web

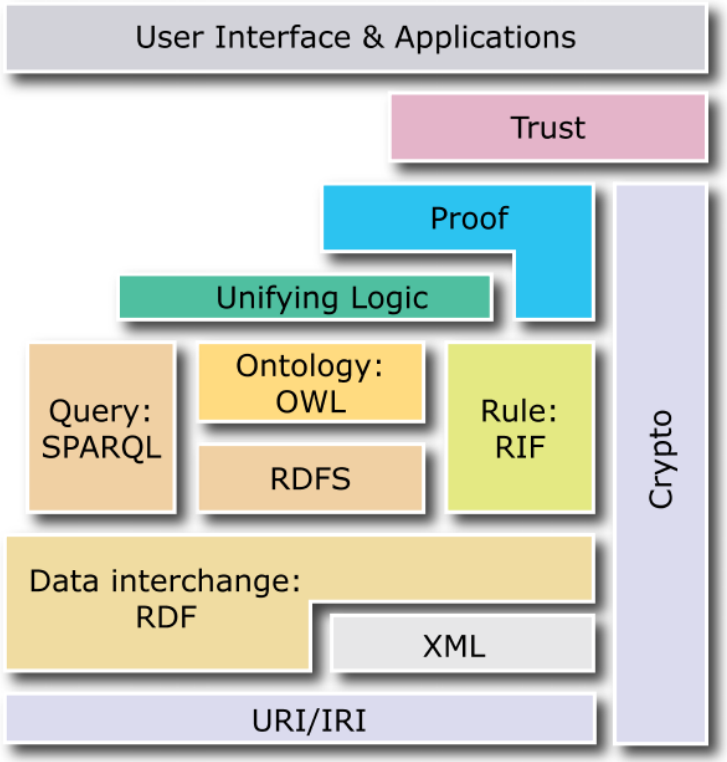

Figure: Tim Berners-Lee Semantic Web Stack

Web Data

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

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

XML

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 Scottland</title>
    <review><em>The <em>best</em> ever!</em></review>
</book>
</books>
```

XML

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

RDF

```
<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>
```

RDF

RDFS

<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>

OWL

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

OWL

```
<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>
<rdf:Property rdf:about="#husband">
<owl:inverseOf rdf:resource="#wife"/>
</rdf:Property>
```

XQuery

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

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

XQuery

```
<result>
<mybook year="1994"><title>TCP/IP Illustrated< /title>< /mybook>
<mybook year="1992"><title>Advanced Programming
        in the Unix environment</title>< /mybook>
< /result>
```

SPARQL

SPARQL

```
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 . }
```

XPath

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

Facts (Document):

```
year('2003', [1, 1], 3,``books.xml").
author('Abiteboul', [1, 1, 1], 3,``books.xml").
author('Buneman', [2,1, 1], 3,``books.xml").
author('Suciu', [3,1,1], 3,``books.xml").
title('Data on the Web', [4, 1, 1], 3,``books.xml").
unlabeled('A', [1, 5, 1, 1], 4,``books.xml").
em('fine', [2, 5, 1, 1], 4,``books.xml").
unlabeled('book.', [3, 5, 1, 1], 4,``books.xml").
year('2002', [2, 1], 3,``books.xml").
author('Buneman', [1, 2, 1], 3,``books.xml").
title('XML in Scottland', [2, 2, 1], 3,``books.xml").
unlabeled('The', [1, 1, 3, 2, 1], 6,``books.xml").
em('best', [2, 1, 3, 2, 1], 6,``books.xml").
unlabeled('ever!', [3, 1, 3, 2, 1], 6,``books.xml").
```

XPath

Prolog term of the XML tree

```
[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

```
xpath([Label],[element(Label,Attr,Sublabel)|Relement],
          [element(Label,Attr,Sublabel)|Relement2]):-!,
          xpath([Label],Relement,Relement2).
xpath([Label],[_|Relement],Relement2):-!,xpath([Label],Relement,Relement2).
...
```

Goal

?- xpath([**books,book,author**],**[element(books,...)]**,Author).

XQuery

XQuery

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

XQuery

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],[TTitle,TYear],''books.xml''):-
       title(Title,[NTitle|Node],TTitle,''books.xml''),
       year(Year,Node,TYear,''books.xml'').
```

XQuery

XQuery

```
<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

```
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

$C \sqsubseteq D$ |	(rdfs:subClassof)
$E \equiv F$ |	(owl:equivalentClass)
$P \sqsubseteq Q$ |	(rdfs:subPropertyOf)
$P \equiv Q$ |	(owl:equivalentProperty)
$P \equiv Q^-$ |	(owl:inverseOf)
$P \equiv P^-$ |	(owl:SymmetricProperty)
$P^+ \sqsubseteq P$ |	(owl:TransitiveProperty)
$\top \sqsubseteq \forall P^-.D$ |	(rdfs:domain)
$\top \sqsubseteq \forall P.D$ |	(rdfs:range)
$P(A, B)$ |	(property filler)
$C(A)$	(individual assertion)

Ontology Reasoning

Description Logic

TBox

(1) $Man \sqsubseteq Person$	(2) $Woman \sqsubseteq Person$
(3) $Person \sqcap \exists author\_of.Manuscript \sqsubseteq Writer$	(4) $Paper \sqcup Book \sqsubseteq Manuscript$
(5) $Book \sqcap \exists topic.\{\text{“XML”}\} \sqsubseteq XMLbook$	(6) $Manuscript \sqcap \exists reviewed\_by.Person \sqsubseteq Reviewed$
(7) $Manuscript \sqsubseteq \forall rating.Score$	(8) $Manuscript \sqsubseteq \forall topic.Topic$
(9) $author\_of \equiv writes$	(10) $average\_rating \sqsubseteq rating$
(11) $authored\_by \equiv author\_of^{-}$	(12) $\top \sqsubseteq \forall\ author\_of.Manuscript$
(13) $\top \sqsubseteq \forall\ author\_of^{-}.Person$	(14) $\top \sqsubseteq \forall\ reviewed\_by.Person$
(15) $\top \sqsubseteq \forall\ reviewed\_by^{-}.Manuscript$	

ABox

(1) $Man(\text{“Abiteboul”})$	
(2) $Man(\text{“Buneman”})$	(3) $Man(\text{“Suciu”})$
(4) $Book(\text{“Data on the Web”})$	(5) $Book(\text{“XML in Scottland”})$
(6) $Paper(\text{“Growing XQuery”})$	(7) $Person(\text{“Anonymous”})$
(8) $author\_of(\text{“Abiteboul”}, \text{“Data on the Web”})$	(9) $authored\_by(\text{“Data on the Web”}, \text{“Buneman”})$
(10) $author\_of(\text{“Suciu”}, \text{“Data on the Web”})$	(11) $author\_of(\text{“Buneman”}, \text{“XML in Scottland”})$
(12) $writes(\text{“Simeon”}, \text{“Growing XQuery”})$	(13) $reviewed\_by(\text{“Data on the Web”}, \text{“Anonymous”})$

Ontology Reasoning

Inference Calculus

Rule Name	Inference	
(Eq1)		$\vdash_{OI} E \equiv E$
(Eq2)	$E \equiv F,\ F \equiv G$	$\vdash_{OI} E \equiv G$
(Eq3)	$E \equiv F$	$\vdash_{OI} F \equiv E$
(Eq4)	$C \sqsubseteq D,\ D \sqsubseteq C$	$\vdash_{OI} C \equiv D$
(Sub1)	$E \equiv F$	$\vdash_{OI} E \sqsubseteq F$
(Sub2)	$C \sqsubseteq D,\ D \sqsubseteq E$	$\vdash_{OI} C \sqsubseteq E$
(Sub3)	$C \sqcup D \sqsubseteq E$	$\vdash_{OI} C \sqsubseteq E$
(Sub4)	$E \sqsubseteq C \sqcap D$	$\vdash_{OI} E \sqsubseteq C$
(Sub5)	$C_1 \sqcap C_2 \sqsubseteq D,\ E \sqsubseteq C_1$	$\vdash_{OI} E \sqcap C_2 \sqsubseteq D$
(Sub6)	$C_1 \sqcup C_2 \sqsubseteq D,\ E \sqsubseteq C_1$	$\vdash_{OI} E \sqcup C_2 \sqsubseteq D$
(Sub7)	$C \sqsubseteq D_1 \sqcap D_2,\ D_1 \sqsubseteq E$	$\vdash_{OI} C \sqsubseteq E \sqcap D_2$
(Sub8)	$\exists P.\{O\} \sqsubseteq D,\ Q \sqsubseteq P$	$\vdash_{OI} \exists Q.\{O\} \sqsubseteq D$
(Sub9)	$\exists P.C \sqsubseteq D,\ Q \sqsubseteq P$	$\vdash_{OI} \exists Q.C \sqsubseteq D$
(Sub10)	$\exists P.C \sqsubseteq D,\ E \sqsubseteq C$	$\vdash_{OI} \exists P.E \sqsubseteq D$
(Sub11)	$C \sqsubseteq \exists P.\{O\},\ P \sqsubseteq Q$	$\vdash_{OI} C \sqsubseteq \exists Q.\{O\}$
(Sub12)	$C \sqsubseteq \forall P.D,\ Q \sqsubseteq P$	$\vdash_{OI} C \sqsubseteq \forall Q.D$
(Sub13)	$C \sqsubseteq \forall P.D,\ D \sqsubseteq E$	$\vdash_{OI} C \sqsubseteq \forall P.E$

Ontology Reasoning

Prolog Rules

Rule Name	Prolog Rule
(Eq1)	$triple(E, owl : equivalentClass, E) : -class(E).$
(Eq2)	$triple(E, owl : equivalentClass, G) : -triple(E, owl : equivalentClass, F),$ $triple(F, owl : equivalentClass, G).$
(Eq3)	$triple(E, owl : equivalentClass, F) : -triple(F, owl : equivalentClass, E).$
(Eq4)	$triple(E, owl : equivalentClass, F) : -triple(E, rdfs : subClassOf, F),$ $triple(F, rdfs : subClassOf, E).$
(Sub1)	$triple(E, rdfs : subClassOf, F) : -triple(E, owl : equivalentClass, F).$
(Sub2)	$triple(C, rdfs : subClassOf, E) : -triple(C, rdfs : subClassOf, D),$ $triple(D, rdfs : subClassOf, E).$
(Sub3)	$triple(D, rdfs : subClassOf, E) : -triple(union(U), rdfs : subClassOf, E), member(D, U).$
(Sub4)	$triple(E, rdfs : subClassOf, C) : -triple(E, rdfs : subClassOf, inter(I)), member(C, I).$
(Sub5)	$triple(inter(I2), rdfs : subClassOf, D) : -triple(inter(I1), rdfs : subClassOf, D),$ $member(C, I1), triple(E, rdfs : subClassOf, C),$ $replace(I1, C, E, I2).$
(Sub6)	$triple(union(U2), rdfs : subClassOf, D) : -triple(union(U1), rdfs : subClassOf, D),$ $member(C, U1), triple(E, rdfs : subClassOf, C),$ $replace(U1, C, E, U2).$
(Sub7)	$triple(C, rdfs : subClassOf, inter(I2)) : -triple(C, rdfs : subClassOf, inter(I1)),$ $member(D, I1), triple(D, rdfs : subClassOf, E),$ $replace(I1, D, E, I2).$
(Sub8)	$triple(hasvalue(Q, O), rdfs : subClassOf, D) : -triple(Q, owl : subPropertyOf, P),$ $triple(hasvalue(P, O), rdfs : subClassOf, D).$

XQuery with Ontology Reasoning

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 with Ontology Reasoning

XQuery extension

```
< 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 >
```

XQuery with Ontology Reasoning

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?