Knowledge Representation for the Semantic Web

Winter Quarter 2010

Slides 2 – 01/07/2010

Pascal Hitzler
Kno.e.sis Center
Wright State University, Dayton, OH
http://www.knoesis.org/pascal/

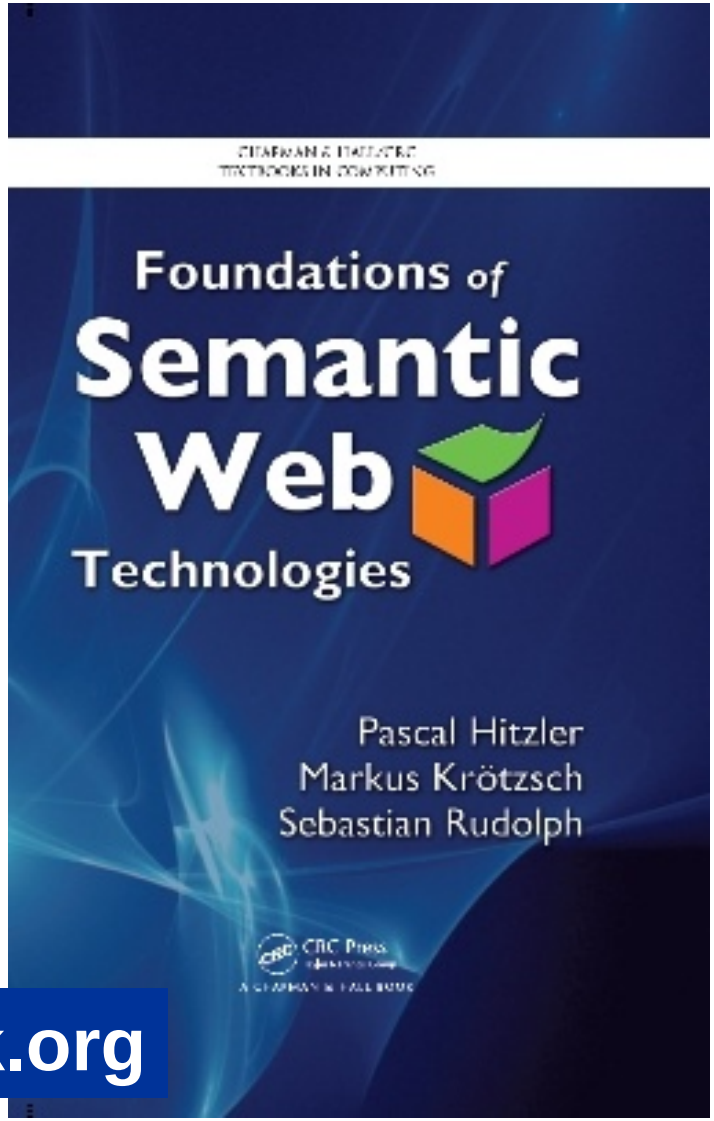
Slides are based on

Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph

Foundations of Semantic Web Technologies

Chapman & Hall/CRC, 2010

Flyer with special offer is available.

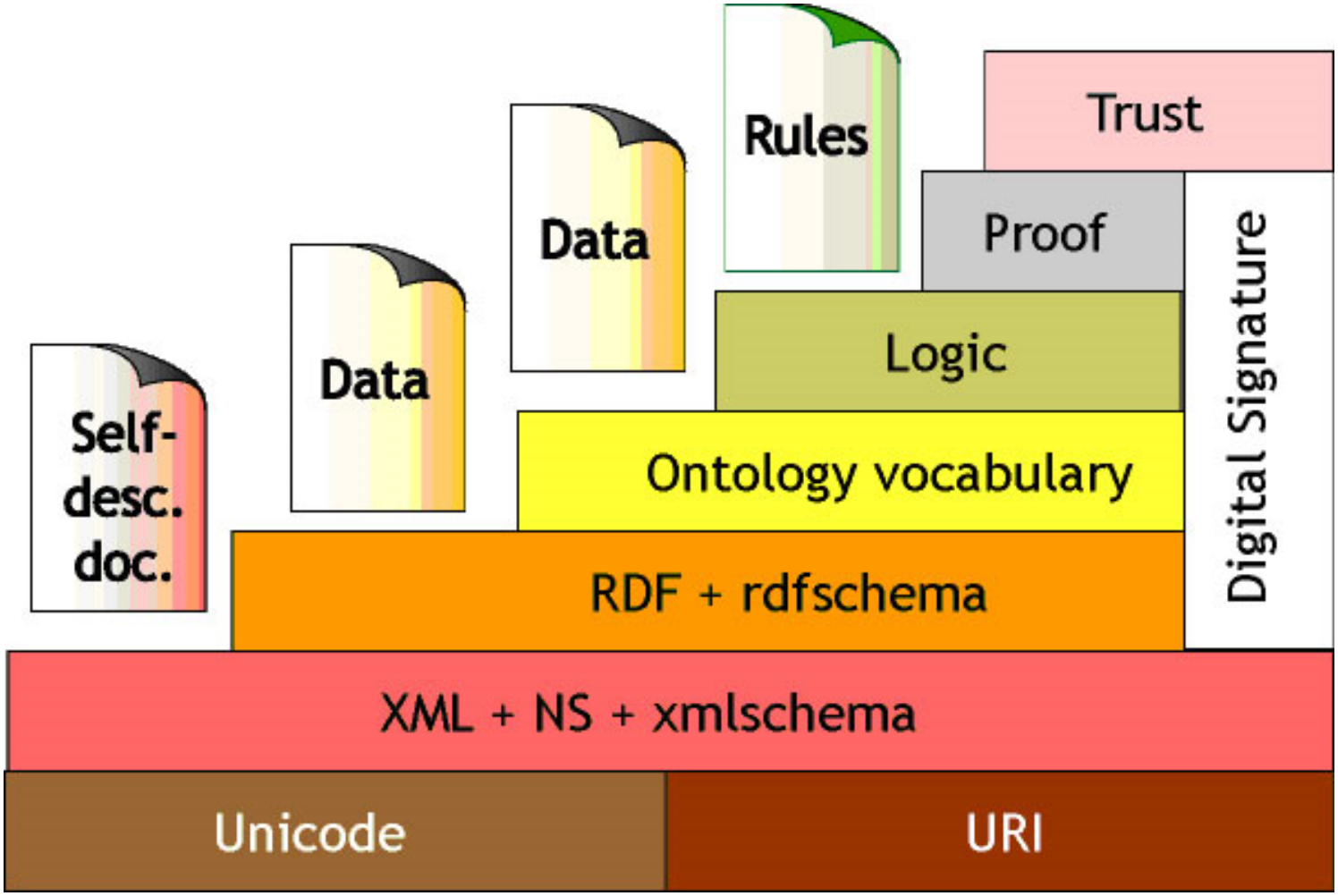


http://www.semantic-web-book.org

Today's Session

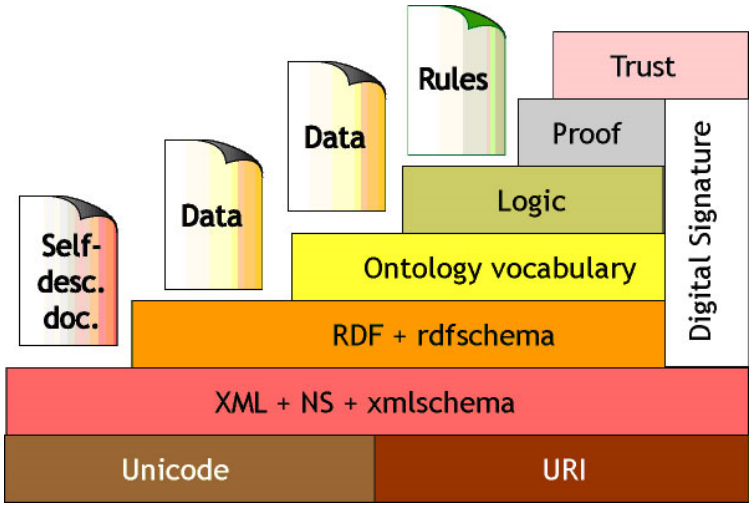
1. **The Semantic Web Layer Cake**
2. **Essentials of the eXtensible Markup Language XML**
3. **Class project – status**
4. **Class presentations – first topics**

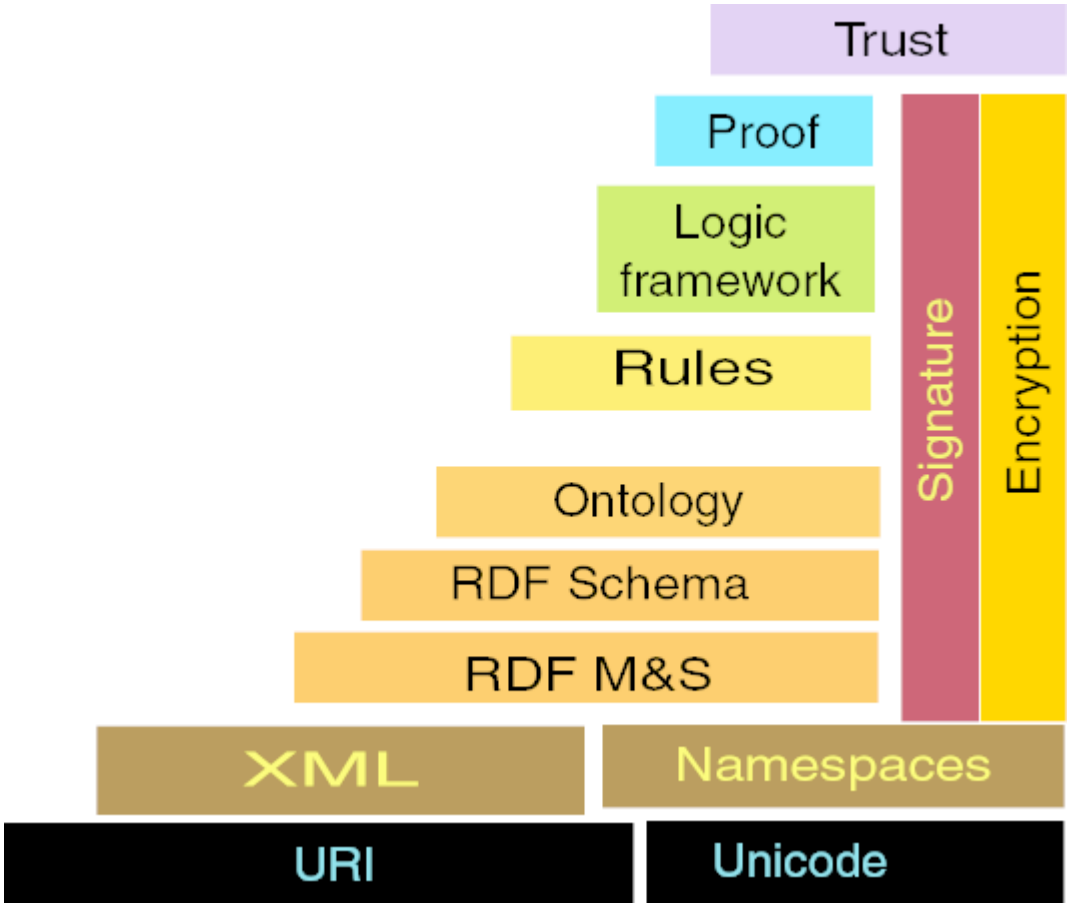
Tim Berners-Lee version, 2000



http://www.w3.org/2000/Talks/1206-xml2k-tbl/Overview.html

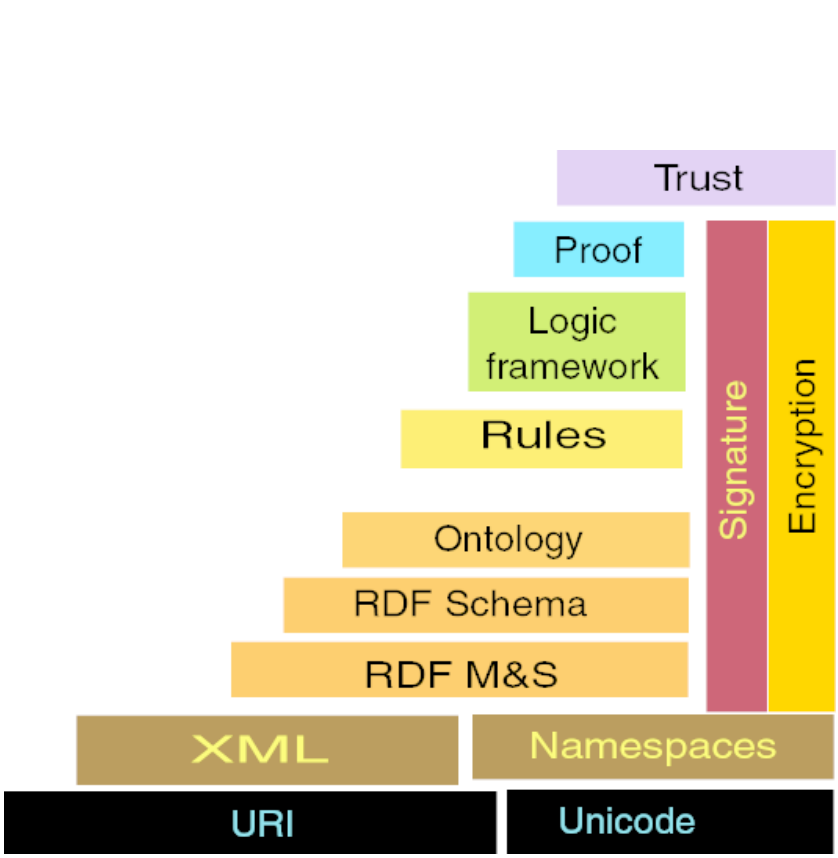
Tim Berners-Lee version, 2003

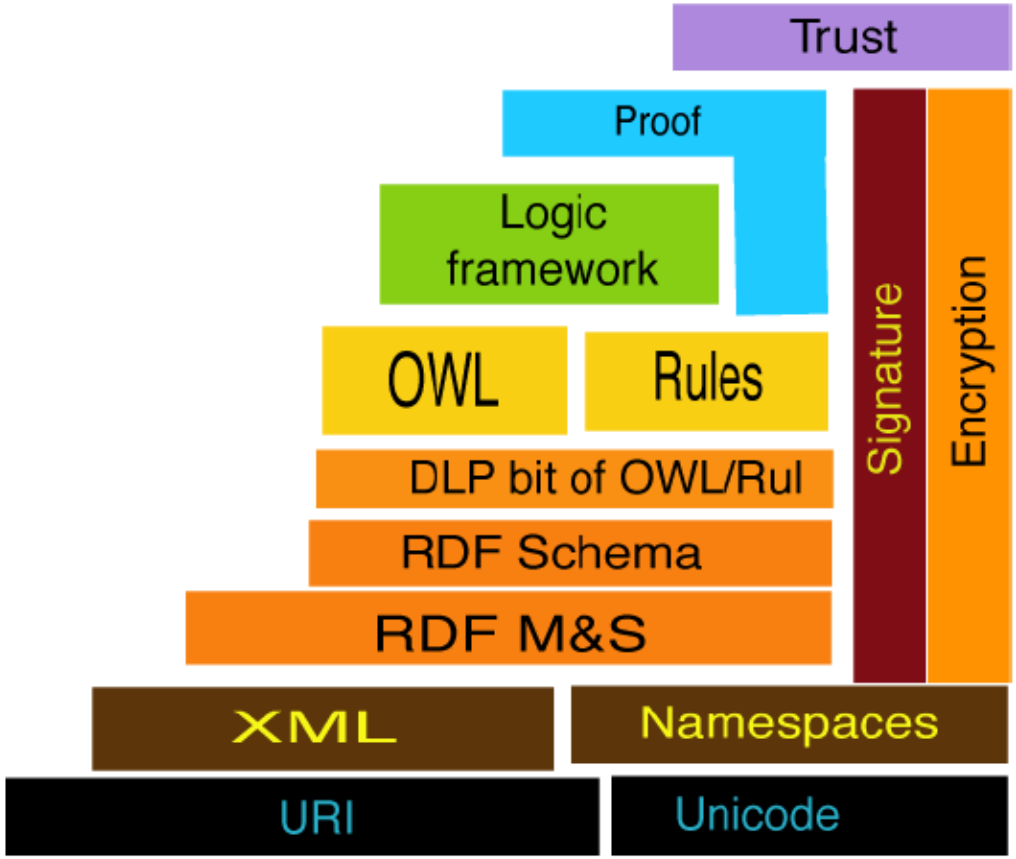




http://www.w3.org/2003/Talks/0922-rsoc-tbl/

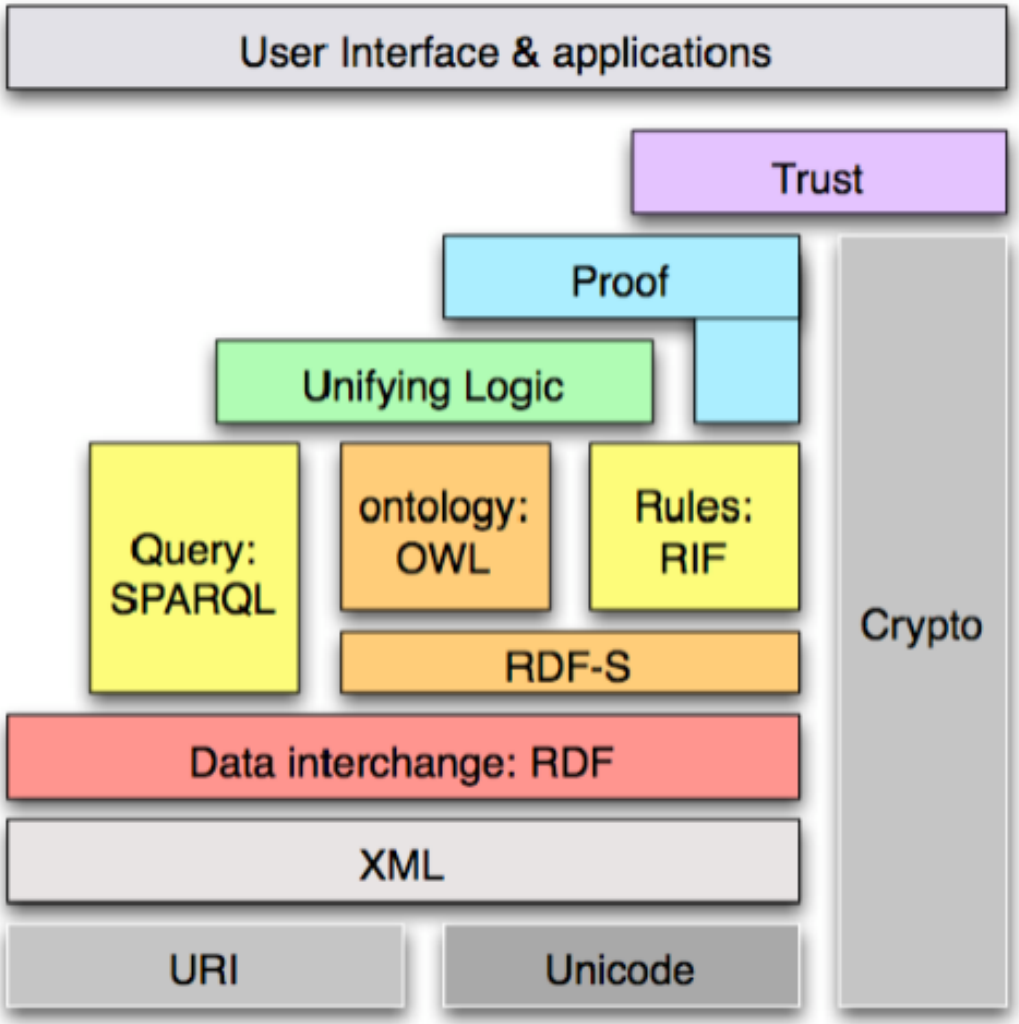
Horrocks et al, 2005





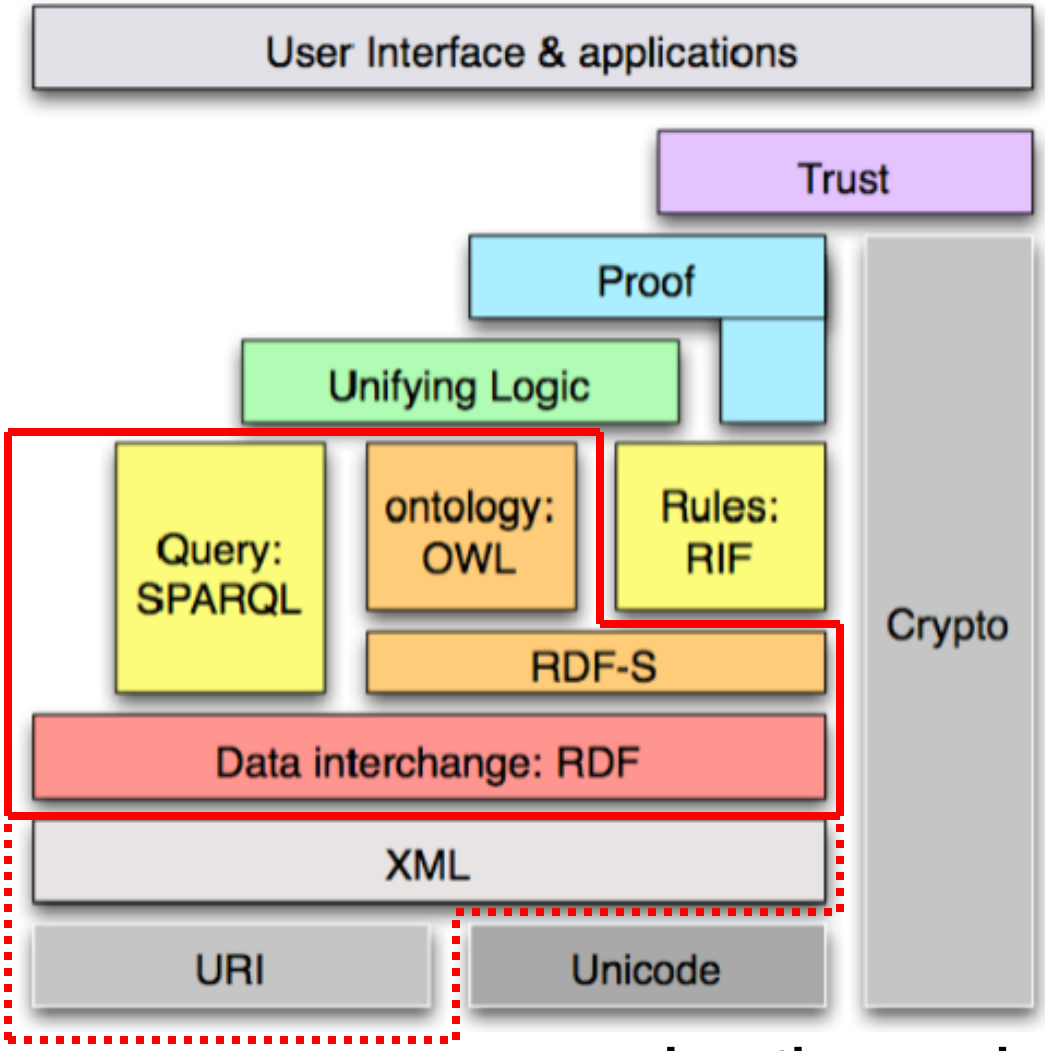
Horrocks, Parsia, Patel-Schneider, Hendler, Semantic Web Architecture: Stack or Two Towers? LNCS 3703, 37-41, 2005.

Tim Berners-Lee version, 2006



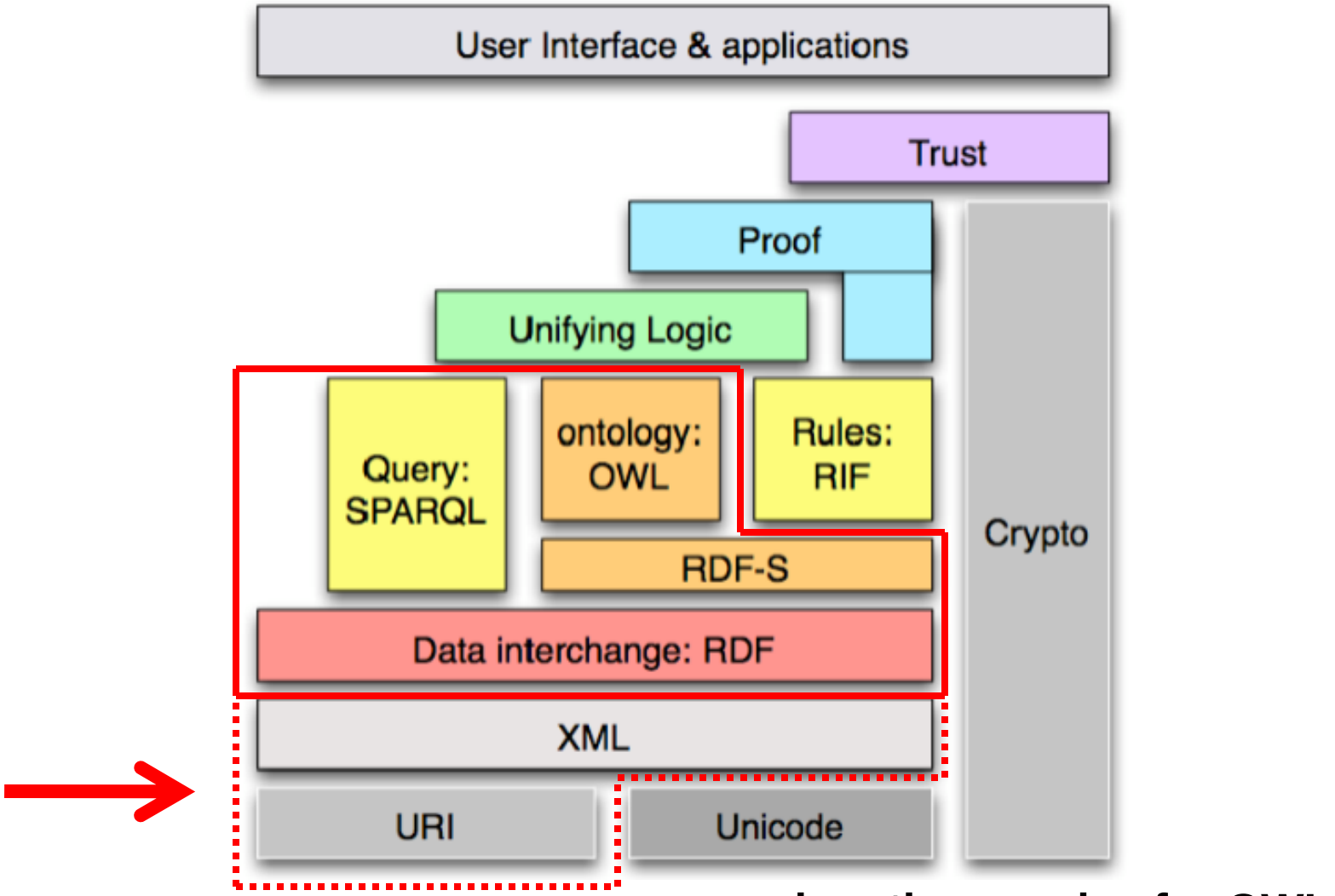
http://www.w3.org/2006/Talks/0718-aaai-tbl/

Planned coverage in this lecture



+ conjunctive queries for OWL

Planned coverage in this lecture



+ conjunctive queries for OWL

Today's Session

1. The Semantic Web Layer Cake

2. Essentials of the eXtensible Markup Language XML

   Appendix A in the textbook, plus some material on namespaces and URIs taken from Chapter 2

3. Class project – status

4. Class presentations – first topics

XML contents

- Motivation
- Syntax
- URIs
- Namespaces
- XML Schema

Markup-languages

- **Basic idea: adding additional information or structure to (unstructured) text**

- **to *annotate* text**
  Webster's: annotation –
  a note added by way of comment or explanation

- **text = data**
  additional info = metadata (data about data)

- **usually done by way of *tags*:**
  <tag-name> ... Text ... </tag-name>
  [opening tag] [closing tag]

Markup-languages

- **Most prominent example: HTML**
  Annotations used for encoding display information

- **This book has the title FOST.**
  Browser shows:

  This book has the title **FOST**.

- **Same idea can be used for content description:**

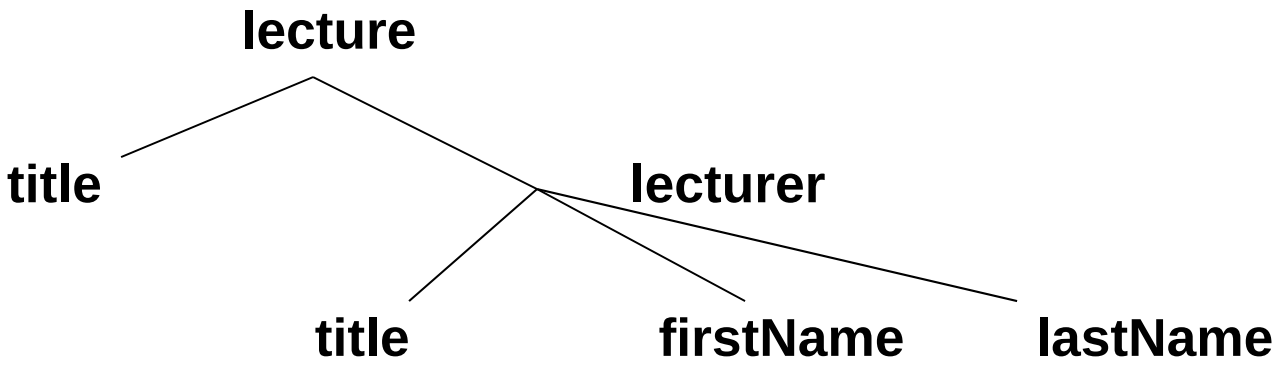
  <book>This book</book> has the title <title>FOST</title>.

Tags may be nested

```
<lecture>
    <title>        KR4SW          </title>
    <lecturer>
        <title>          Prof. Dr.        </title>
        <firstName>      Pascal           </firstName>
        <lastName>       Hitzler          </lastName>
    </lecturer>
</lecture>
```

Tree structure

```
<lecture>
    <title>        KR4SW          </title>
    <lecturer>
        <title>        Prof. Dr.      </title>
        <firstName>    Pascal         </firstName>
        <lastName>     Hitzler        </lastName>
    </lecturer>
</lecture>
```



XML contents

- Motivation
- Syntax
- URIs
- Namespaces
- XML Schema

XML

- eXtensible Markup Language

- origin: structured text
- W3C standard for data exchange
  [see www.w3.org for W3C]
  - input and output data of applications can be described using XML
  - additionally only needed: a standardized description / vocabulary
- complementary to HTML
  - HTML is for display/presentation
  - XML is for describing content
- database view: XML as data model for semi-structured data

XML-Syntax: prolog

- every XML document is a text document
- every XML document begins with a declaration containing
  - the version number of the used standard
  - and optionally, the character encoding.
- example:

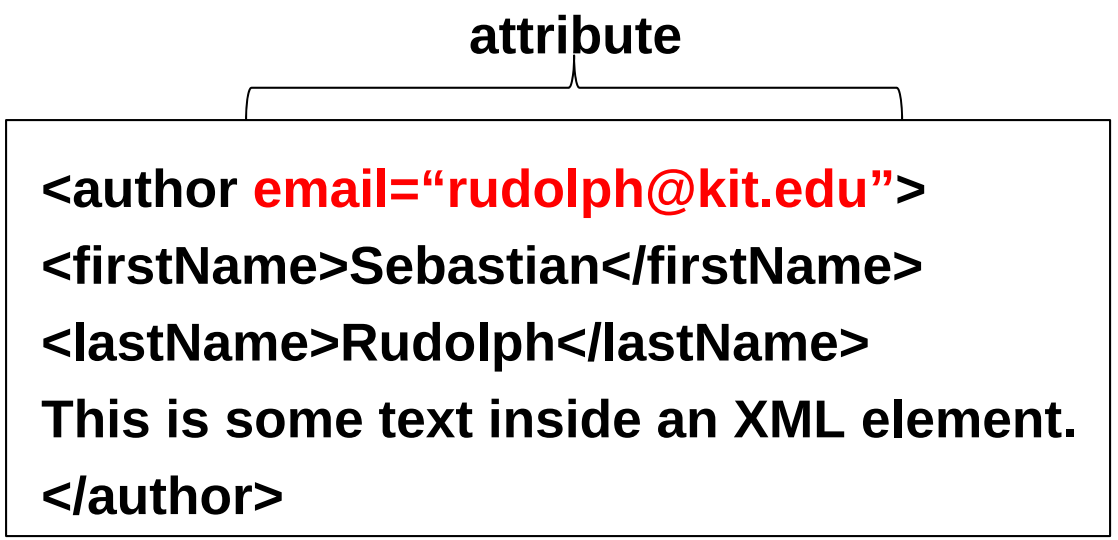
<?xml version="1.0" encoding="utf-8"?>

XML-Syntax: XML Elements

- **XML elements**
  - **describe objects which are enclosed in matching tag-pairs.**
  - **can contain text and/or further XML elements, arbitrarily nested.**
  - **empty elements can be abbreviated, e.g. <year></year> can be written as <year/>.**
  - **the outermost element is called *root* element (there is only one)**

XML-Syntax: XML Attributes

- **XML attributes**
  - **are name-string-pairs in opening tags (or self-closing tags).**
  - **are associated with the corresponding XML element.**
  - **are an alternative means to sub-elements for describing data.**



XML-Syntax, XML vs. HTML

- **XML Documents which are syntactically correct, are said to be *well-formed*.**

- **XML vs HTML:**
  - **HTML uses a fixed vocabulary (set of tags) with a fixed meaning (for display of text)**
  - **XML allows free choice of tag names, whose meaning is not fixed.**

```
<h1> Bib </h1>
  <p>
    <i>FOST</i>
    <b>2010</b>
  <p>
```

```
<Bib id="o1">
        <title>FOST</title>
        <author>...</author>
        <year>2010</year>
</Bib>
```

XML contents

- Motivation
- Syntax
- URIs
- Namespaces
- XML Schema

URIs

- **URI = Uniform Resource Identifier**
  URL = Uniform Resource Locator (has a location on the WWW)
  IRI = Internationalized Resource Identifier (uses Unicode)
  URLs $\subseteq$ URIs $\subseteq$ IRIs

- **used for identifying Web resources**

- **resources can be anything that has an identity in the context of an application (books, locations, humans, abstract concepts, etc.)**

- **analogous to, e.g., ISBN for books**

URIs – format

scheme:[//authority]path[?query][#fragment]

- **scheme: type of URI, e.g. http, ftp, mailto, file, irc**
- **authority: typically a domain name**
- **path: e.g. /etc/passwd/**
- **query: optional; provides non-hierarchical information. Usually for parameters, e.g. for a web service**
- **fragment: optional; often used to address part of a retrieved resource, e.g. section of a HTML file.**

- **not all characters are allowed in URIs.**

URIs

- where do they come from?
- what URIs to use?
- what does a URI stand for?

  http://www.pascal-hitzler.de – is this a URI for a web page or for the person “Pascal Hitzler”?

- What about URIs which do not dereference?

XML contents

- Motivation
- Syntax
- URIs
- Namespaces
- XML Schema

Namespaces

```
<lecture>
    <title>        KR4SW          </title>
    <lecturer>
        <title>          Prof. Dr.      </title>
        <firstName>      Pascal         </firstName>
        <lastName>       Hitzler        </lastName>
    </lecturer>
</lecture>
```

- same tag name – probably better to disambiguate

Namespaces

```
<lecture        xmlns:lec="http://example.org/lecture/"
                xmlns:person="http://example.org/person/>
   <lec:title>  KR4SW          </lec:title>
   <lec:lecturer>
        <person:title>          Prof. Dr.       </person:title>
        <person:firstName>      Pascal          </person:firstName>
        <person:lastName>       Hitzler         </person:lastName>
   </lec:lecturer>
</lec:lecture>
```

- disambiguate using namespaces
- same mechanism can be used for indicating different sources for data

Namespaces – declaration mechanisms

- **Namespace declaration**
  Usage: namespace:name in XML element names
  Declaration: xmlns:namespace="<uri>" in XML opening tags or empty-element tags. Affects XML subtree, multiple declarations possible.
- **Base namespace (only RDF)**
  Usage: non-URI name as value for some RDF/XML elements.
  Declaration: xml:base="<uri>" in XML opening tags or empty-element tags. Affects XML subtree, multiple declarations possible.
- **Entity declaration**
  This is part of so-called *Document Type Definitions*.
  Usage: &entity; in XML attribute values or RDF literal values.
  Declaration: <!ENTITY entity 'text'> in initial DOCTYPE declaration. Affects whole document, only one declaration possible.

Namespaces – entity declarations

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF
    [  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
       <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
       <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
       <!ENTITY otherOnt "http://example.org/otherOntology/" >
    ]>
```

Usage examples follow below.

We will not discuss Document Type Declarations (DTDs) in more detail – they are a weaker mechanism than XML schema. Just use the above as a form of "macro".

XML contents

- Motivation
- Syntax
- URIs
- Namespaces
- XML Schema

XML Schema

- XML allows a lot of freedom in encoding information

<author>Sebastian Rudolph</author>

<author name="Sebastian Rudolph"/>

<author><fullName>Sebastian Rudolph</fullName></author>

<author> <firstName>Sebastian</firstName>
<secondName>Rudolph</secondName> </author>

<author givenName="Sebastian" surname="Rudolph"/>

XML Schema

- These degrees of freedom get in the way when exchanging XML documents between applications!
- It is necessary to come up with agreements about the structure of the information, including the names of tags and attributes, and whether certain subelements are required or not.
- XML Schema is a W3C standard which provides for this.
- XML schemas are themselves written in XML.
- An XML document is said to be *valid* if it adheres to a corresponding XML schema.

XML Schema

- **An XML Schema document is a well-formed XML document which contains *XML schema definitions*.**
- **An XML schema definition begins with an opening tag like**

  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  it then contains *element types, which can contain attribute types*, which themselves refer to predefined or user-defined datatypes.

- **datatypes are, e.g. xsd:integer, xsd:string, xsd:time, xsd:date, xsd:anyURI,**
  xsd:ID (a specific kind of string used as identifier of XML elements)

XML Schema Example

```
<?xml version="1.1" encoding="utf-16"?>
<!DOCTYPE xsd:schema
    [  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    ]>
  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:element name="author" type="&xsd;string"
                   minOccurs="1" maxOccurs="unbounded">
          <xsd:attribute name="email" type="&xsd;string"
                   use="required">
          <xsd:attribute name="homepage"
                   type="&xsd;anyURI" use="optional">
      </xsd:element>
  </xsd:schema>
```

XML Schema Example

```
<xsd:element name="author" type="&xsd;string"
                    minOccurs="1" maxOccurs="unbounded">
          <xsd:attribute name="email" type="&xsd;string"
                    use="required">
          <xsd:attribute name="homepage" type="&xsd;anyURI"
                    use="optional">
    </xsd:element>

<author email="email1@example.org" homepage="http://korrekt.org">
    Markus Kroetzsch
</author>
<author email="email2@example.org" >
    Sebastian Rudolph
</author>
```

XML Schema – user-defined types

Simple types: obtained by restricting other types.

```
<xsd:simpleType name="humanAge">
    <xsd:restriction base="&xsd;integer">
        <xsd:minInclusive value="0"/>
        <xsd:maxInclusive value=200"/>
    </xsd:restriction>
</xsd:simpleType
```

No use of embedded element or attribute types!

XML Schema – user-defined types

```
<xsd:complexType name="bookType“>
    <xsd:sequence>
        <xsd:element name="author" type="&xsd;string“
            minOccurs="1“ maxOccurs="unbounded" />
        <xsd:element name="title" type="&xsd;string”
            minOccurs="1” maxOccurs="1" />
        <xsd:element name="publisher" type="&xsd;string”
            minOccurs="1“ maxOccurs="1" />
        <xsd:element name="year" type="&xsd;gYear”
            minOccurs="1" maxOccurs="1" />
    </xsd:sequence>
    <xsd:attribute name="ISBNnumber" type=“&xsd;nonNegativeInteger“
        use="optional" />
</xsd:complexType>
```

XML Schema – user-defined types

```
<xsd:complexType name="researchBookType">
    <xsd:extension base="bookType">
    <xsd:sequence>
          <xsd:element name="field" type="&xsd;string" />
    </xsd:sequence>
    <xsd:attribute name="price" type="&xsd;nonNegativeInteger“
                  use="optional" />
</xsd:complexType>
```

Today's Session

1. The Semantic Web Layer Cake
2. Essentials of the eXtensible Markup Language XML
3. Class project – status
4. Class presentations – first topics

Class project – status

- Aircrafts
- American Football
- Car
- Computer Science
- Cuisine
- Networks
- Parasite lifecycle
- People at University
- Trust

Today's Session

1. The Semantic Web Layer Cake
2. Essentials of the eXtensible Markup Language XML
3. Class project – status
4. Class presentations – first topics

Class presentations – first topics

Standards:

- **RDFa – embedding RDF in HTML (W3C standard)**
- **SKOS – data model for sharing and linking knowledge organization systems via the Web (W3C standard)**

Tools:

- **Protege – Ontology editing tool**

Research papers:

- **Parallel Materialization of the Finite RDFS Closure for Hundreds of Millions of Triples (Weaver, Hendler, ISWC2009)**
- **Scalable Distributed Reasoning using MapReduce (Urbani, Kotoulas, Oren, van Harmelen, ISWC2009)**

Class Planning

Topic next Tuesday: RDF Part I

Exercise session planned for Tuesday, 26th of January

Estimated (incomplete) breakdown of sessions:

Intro + XML: 2

RDF: 3

OWL and Logic: 5

SPARQL and Querying: 2

Class Presentations: 3

Exercise sessions: 3