Einführung in die Grundprinzipien des Semantic Webs

Cristina Vertan

Inhalt

- Das Semantic-Web-Konzept
- Architektur des Semantic -Webs
- Ressourcen-Annotation für Semantic Web
- Ontologien und Semantic Web
- Visualisieung des Semantic Webs

Grenzen des aktuelles WWW -1-

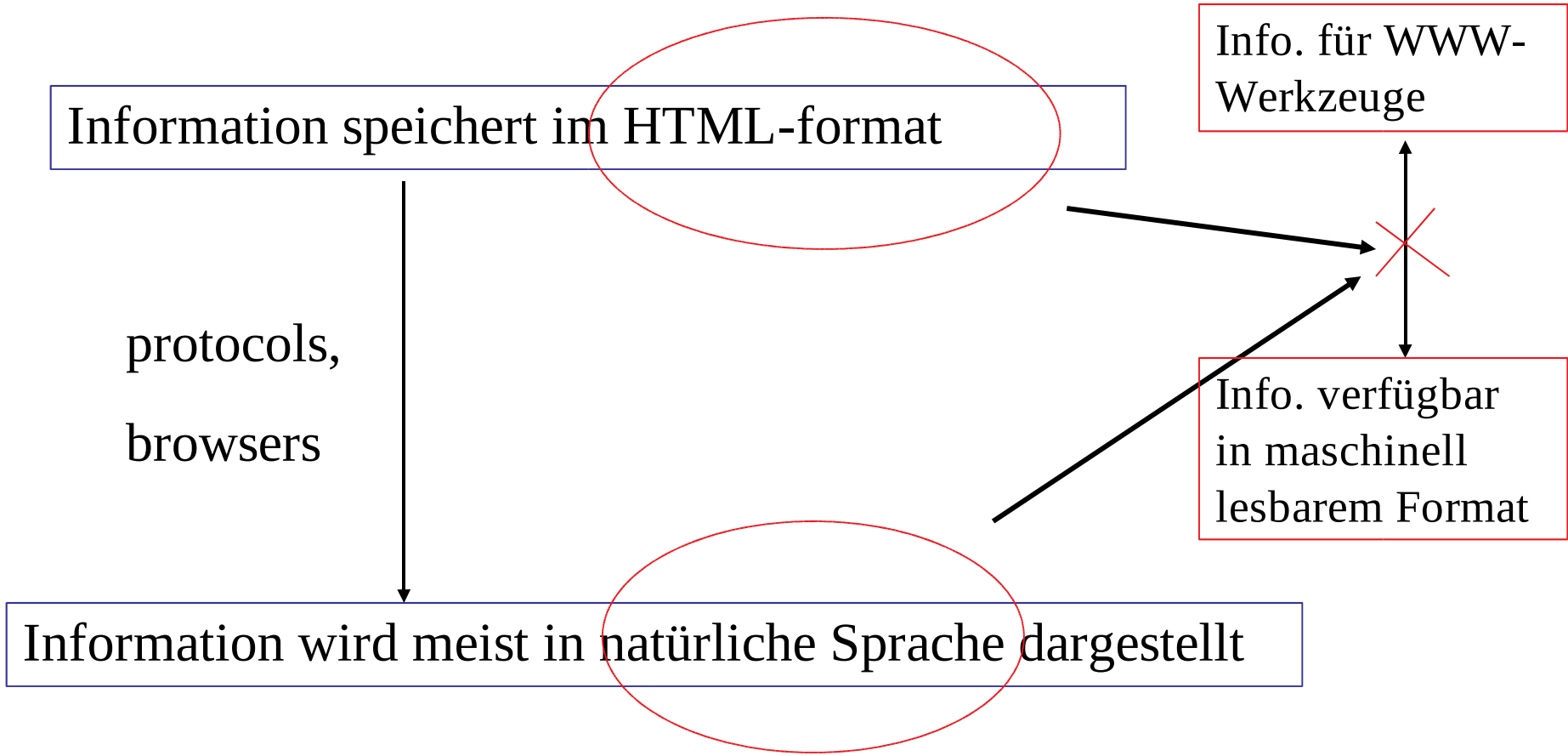
- approx 3 Milliarden statische Dokumente in WWW
- approx. 200 Mi.lionen Benutzer
- diese Ziffern steigen kontinuerlich

WWW heute

- Suchfunktionnen
- Zugänglichkeit
- Darstellungsmöglichkeiten
- Information up-to-date

schwierig

Grenzen des aktuellen WWW -2-



Suche in WWW

- meistens stellen die aktuellen Suchmaschinen viel zu viel irrelevante Information zur Verfügung.
- Die Suchmaschinen geben nur pointers zu Dokumenten. Um die gewümschte Information zu kriegen, muss man die Dokumente lesen
- Wichtige Iinformation für den gesuchte Begriff wird nicht gefunden weil er mit anderen als den Eingegeben Wörtern repräsentiert ist
- Die Situation ist schlechter wenn man die Suche auch auf Multimediainformation erweitert

Informationsdarstellung im WW

- mit der aktuellen Web-Wekzeuge est ist sehr schwer nicht-redundante und konsistente Information zu pflegen
- Die Webmasters sind sehr oft überfördert
- Viele Webseiten enhalten wiedersprüchliche Information

E-commerce und WWW

- Einkaufsagenten benutzen Wrappers und Heuristiken um Produktinformation von schlecht strukturierter textueller Information zu extrahieren
- Entwicklungs- und Pflegekosten sind sehr hoch und das Dienstangebot ist begrenzt

Was bringt das Semantic Web ?

Der Informationinhalt wird für die Maschinen verfügbar

- verbesserte zielorientierte Suche
- wiedersprüchliche Information wird automatisch entdeckt
- Informationsextraktion (z.B. für E-commerce) erleichtert und erweitert.

Was ist das Semantic Web ?

- unterschiedliche Definitionen entsprechend den unterschiedlichen Beiträgen (e-commerce, Netzwerk, KI, Wissensmanagement, Sprachverarbeitung)
- Semantic Web Agreement Group (SWAG) (2000)

SWAG 2001

Das Semantic Web ist ein Web fürs Dokumente und Dokumentteile, die explizit die Beziehungen zwischen Objekte beschreiben. Es enthält die semantische Information, die für maschinelle Verarbeitung benötigt wird.

WWW Veränderung im Semantic Web

Web

2te Generation

Semantic Webseiten:

XML, RDF -Technologien die Bedeutung mit der Data Verbinden

Web = XML-basierte Datenbank+software agenten

Web

1ste Generation

Dynamische Webseiten:

DHTML, Javascript, Java,

Server-side Technologien

Statische Webseiten:

HTML, Hyperlinks, GIFs, Datenbanken+SQL

Was bringt XML-like Annotierung ?

- Beispiel : wir suchen etwas über „*Java programming language*“
- Google wirft alle Dokumente aus die eins von den eingegebenen Wörtern enthalten. Deswegen kriegen wir auch Seiten über die Insel Java...
- mit XML-like Annotierung kann man unterscheiden zwischen z.B.:

  <programming> Java </programming> und

  <geographisch>Java </geographisch>

- aber ....

Was kann ein XML nicht lösen

- z.B. ein tag:

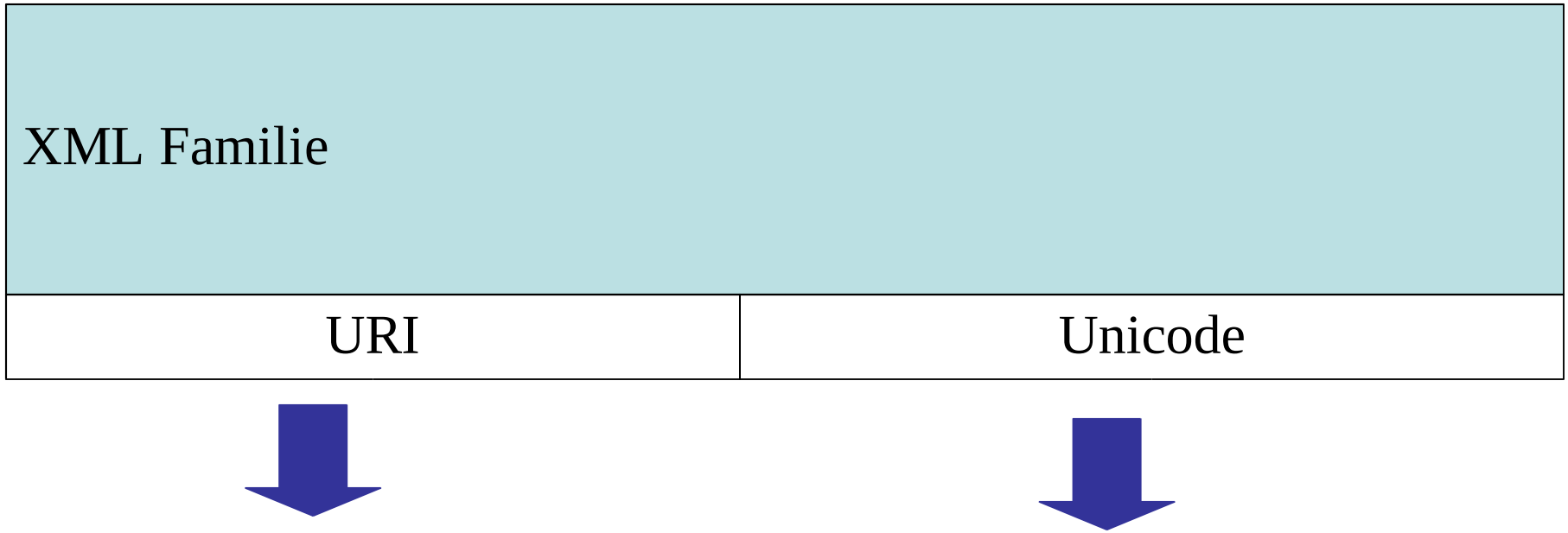
<programmierung>Java</programmierung>

- wird auch ignoriert weil die Maschine nicht verstehet dass die Bedeutung von programming und programmierung dieselbe ist

Inhalt

- Das Semantic-Web-Konzept
- Architektur des Semantic -Webs
- Ressourcen-Annotation für Semantic Web
- Ontologien und Semantic Web
- Visualisierung des Semantic Webs

„Layer-cake“ Architektur (nach Tim Berns-Lee)-1-



Uniform Ressource Identifier (URI)

- ein URI ist ein Ressourcename
- ein URI gibt nicht unbedingt die Ressourcestelle an
- ein URL ist ein URI -Beispiel
- URI‘s dienen der Identifizierung der XML-tags:
- - erstens wird eine Webseite erzeugt wo die benötigten Tags beschrieben werden (Namespace)
- - das URL dieser Webseite ist das URI für das Namespace

URI-Beispiel

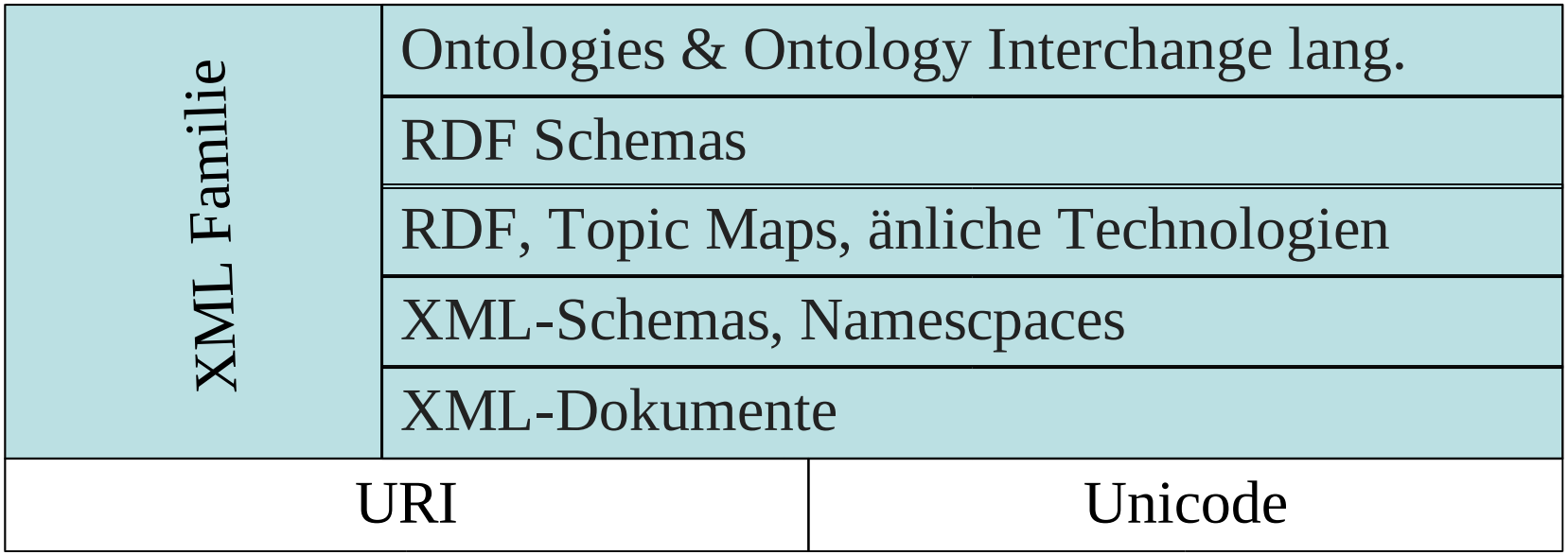
URI für meine eigenen tags

```
<sentence
 xmnls=„http://nats-wiki.informatik.uni-hamburg.de/~cri/“
xmnls:c=„http://tags.example.net/xmnls/“
>
<c:objekt> Semantic Web</c:objekt> wurde von <c:person>Tim
    Bern-Lee</c:person> entwickelt.
</sentence>
```

URI für alle anderen tags,

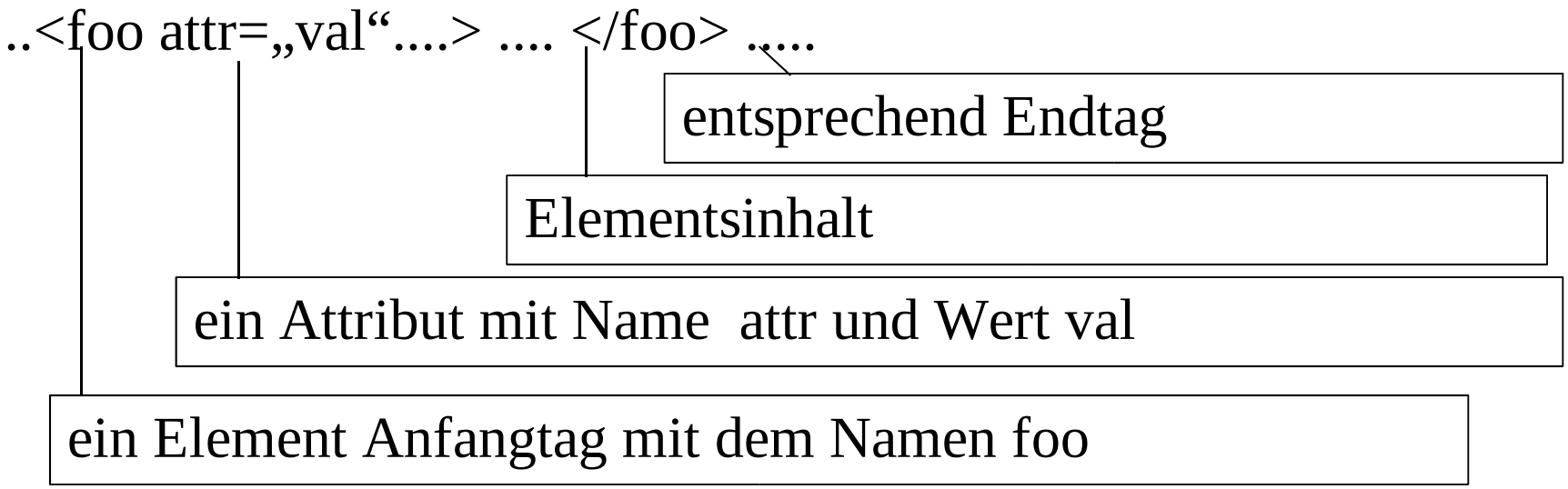
die c enthalten

„Layer-cake“ Architektur (nach Tim Berns-Lee)-2-



XML-Dokumente -1-

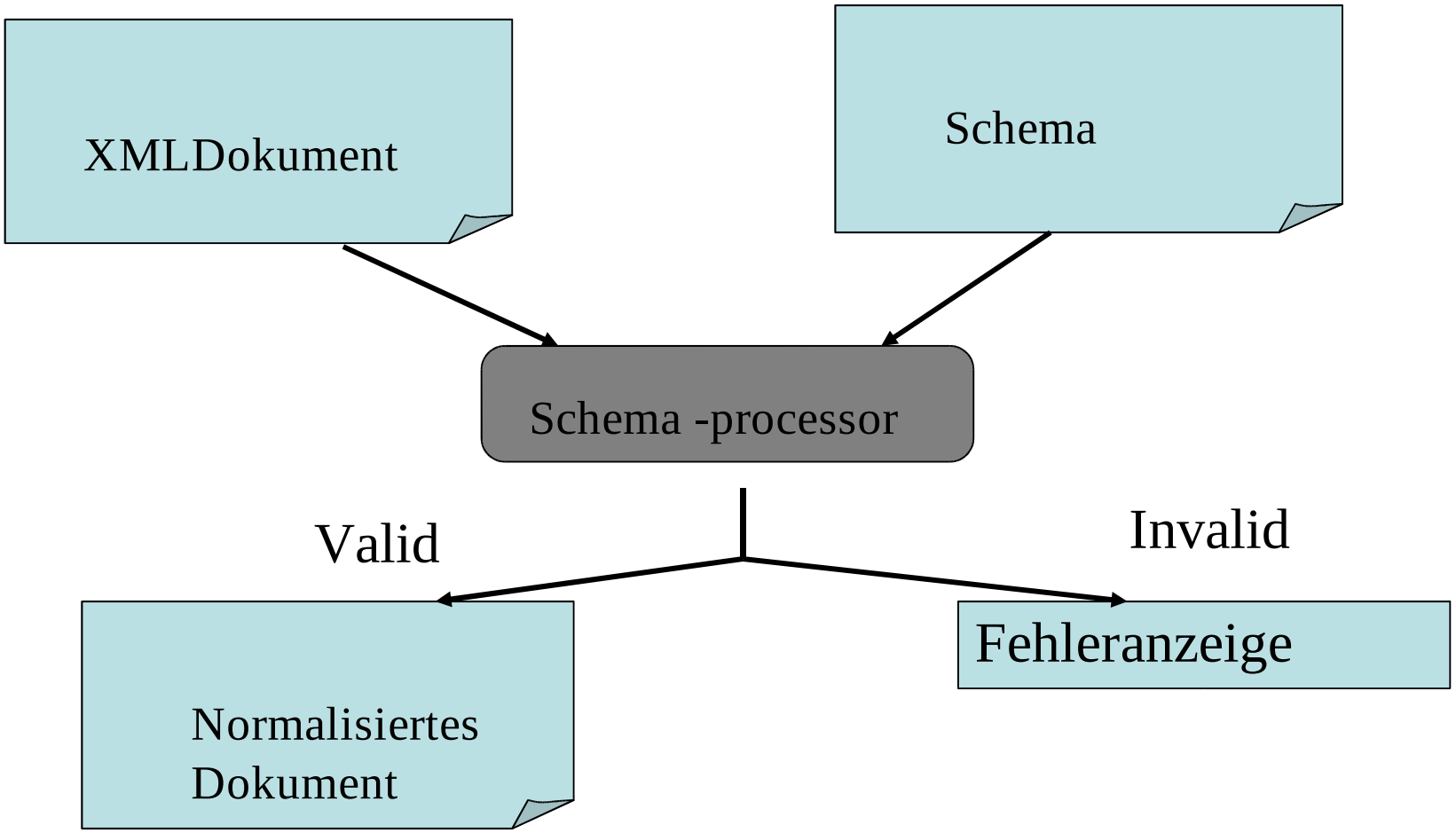
- größte Anzahl an Dokumenten im aktuellen Semantic Web
- ein XML Dokument ist ein (Unicode) Text der Markuptags und andere Meta-Informationen enthält



XML-Schemas -1-

- Ein Schema definiert die Syntax der eigenen „XML-Sprache“
- Eine Schemasprache ist eine formelle Sprache die Schemen ausdrückt
- Geben:
  - Ein XML Dokument und
  - Ein Schema
- eine Schemenvearabeitungs -Prozeß:
  - untersucht die Korrektheit des Dokuments
  - liefert eine normalisierte Version (z.B. default Attribute werden eingefügt)
  - erzeugt ein Instanz-Dokument (Anwendungsdokument)

XML-Schemas -2-



XML-Schemen -3-

- DTD (Dokument Type Definition)
- .....
- XML-Schema
- Hauptprobleme mit DTD:
  - benutzt keine XML-Syntax (sondern SGML)
  - keine Bedigungen für Character-Data
  - keine Unterstutzung für Namespaces
  - sehr weinge Unterstutzung für Modularität und Wiederbenutzbarkeit

XML-Schema -4-
Beispiel

- XML-based language für business cards
- Ein Beispiel-Dokument:

```
<card xmlns=„http://businesscard.org“>
    <name>John Doe </name>
     <title>CEO, Widget Inc </title>
      <email>john.doe@widget.com</email>
       <phone>(202) 466-414 </phone>
      <logo url=widget.gif“/>
</card>
```

business_card.**xml**

XML-Schema -5- Beispiel

```
<schema xmnls=„http://www.w3.org/2001/XMLSchema“
          xmnls:b=„http://buinesscard.org“
          targetNamespace=„http://businesscard.org“>
<element name=„card“ type=„b:card_type“/>
<element name=„name“ type=„string“/>
<element name=„title“ type=„string“/>
<element name=„email“ type=„string“/>
<element name=„phone“ type=„string“/>
<element name=„logo“ type=„b:logo_type“/>
<complexType name=„card_type“>
  <sequence>
     <element ref=„b:name“/>
      <element ref=„b:title“/>
     <element ref=„b:email“/>
      <element ref=„b:phone“ minOccurs=„0“/>
     <element ref=„b:logo“ minOccurs=„0“/>
 </sequence>
</complexType>
<complexType name=„logo_type“>
   <attribute name=„url“ type=„anyURI“/>
</complexType>
</schema>
```

business_card.**xsd**

XML-Schema -6- Beispiel

- Referenz zu einem XML-Schema in einem Dokument:

```
<card xmnls=„http://businesscard.org“
        xmnls:xsi=„http://www.w3.org/2001/XMLSchema-instance“
        xsi:schemaLocation=„http://businesscard.org/business_card.xsd“>
.......
<card>
```

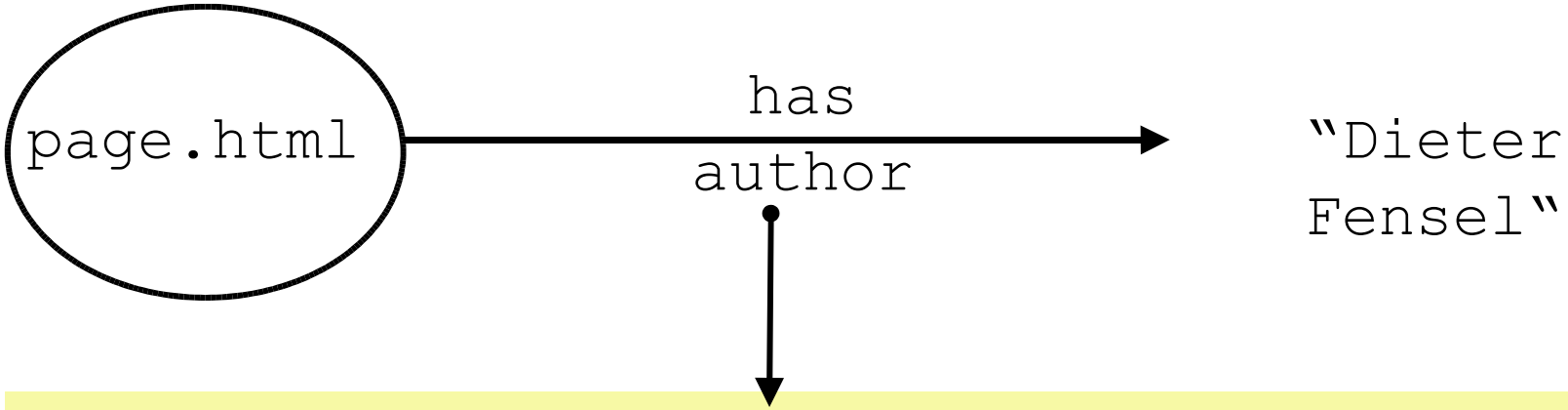
business_card.**xml**

- Das Dokument sollte gültig entspechend den spezifizierten Schemen sein

RDF (Ressource Description Framework)

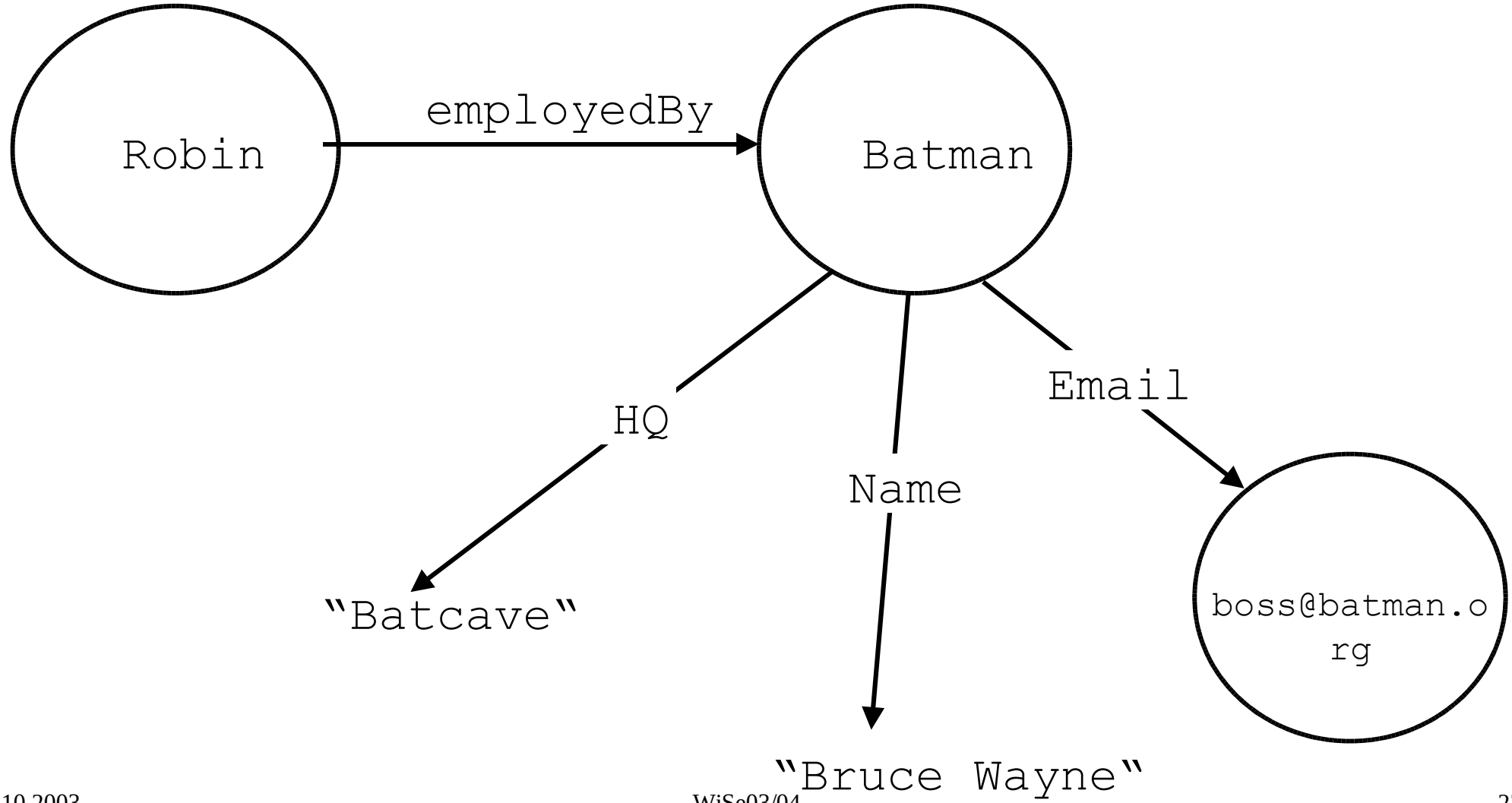
- Domäne-unabhängiger Mechanismus um meta-Data in einem maschinell lesbaren Format
- 3 Konzepte:
  - Ressourcen= alles die mit ein URI referenziert sein kann ,
  - Merkmalen= eine Relation oder Merkmalle die spezifisch für die Ressource ist,
  - Statements= (Objekt, Attribut, Wert)

RDF -Beispiel-1-



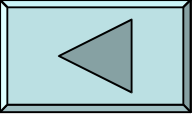

```
<rdf:rdf xmlns:rdf = “http://www.w3.org/1999/02/22-rdf-
syntax-ns#”
         xmlns:dc = “http://purl.oclc.org/DC”>

  <rdf:description rdf:about =
“http://www.Dot.com/page.html”>
    <dc:creator>Dieter Fensel</dc:creator>
  </rdf:description>

</rdf:RDF>
```

RDF -Beispiel 2-



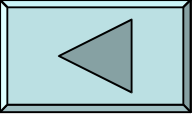
RDF-Beispiel 2-

```
<rdf:rdf
       xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:mySchema = "http://www.Batman.org/mySchema/">

  <rdf:description rdf:about = "http://www.Batman.org/Robin/">
    <mySchema:employedBy rdf:resource = "#Batman"/>
  </rdf:description>

  <rdf:description id = "Batman">
    <mySchema:HQ>Batcave</mySchema:HQ>
    <mySchema:Name>Bruce Wayne</mySchema:Name
    <mySchema:Email rdf:resource = "mailto:boss@batman.org" />
  </rdf:description>

</rdf:rdf>
```



Topic Maps

- neue Standard definiert von ISO (International Organisation for Standardization)
- man kann ein semantisches Netz mit den Ressourcen verbinden
- eine Topic Map ist ein XML Dokument basiert auf 3 Konzepten:
  - Topics (ähnlich wie Enzyklopedia - Einträge)
  - Associations = verbindet mehrere Topics
  - Occurences (eine Topic kann mit eine oder mehrere Occurences verbunden sein
- Topic-maps sind Wissens-orientiert
- RDF sind Ressourcen-orientiert



RDF Schemen

- definieren Domäne-spezifische Merkmale und Ressource-Klassen
- Die Klassen und Merkmalen sind hierachisch organisiert
- Primitive:
  - class
  - property
- Statements:
  - subclass-of
  - subproperty-of

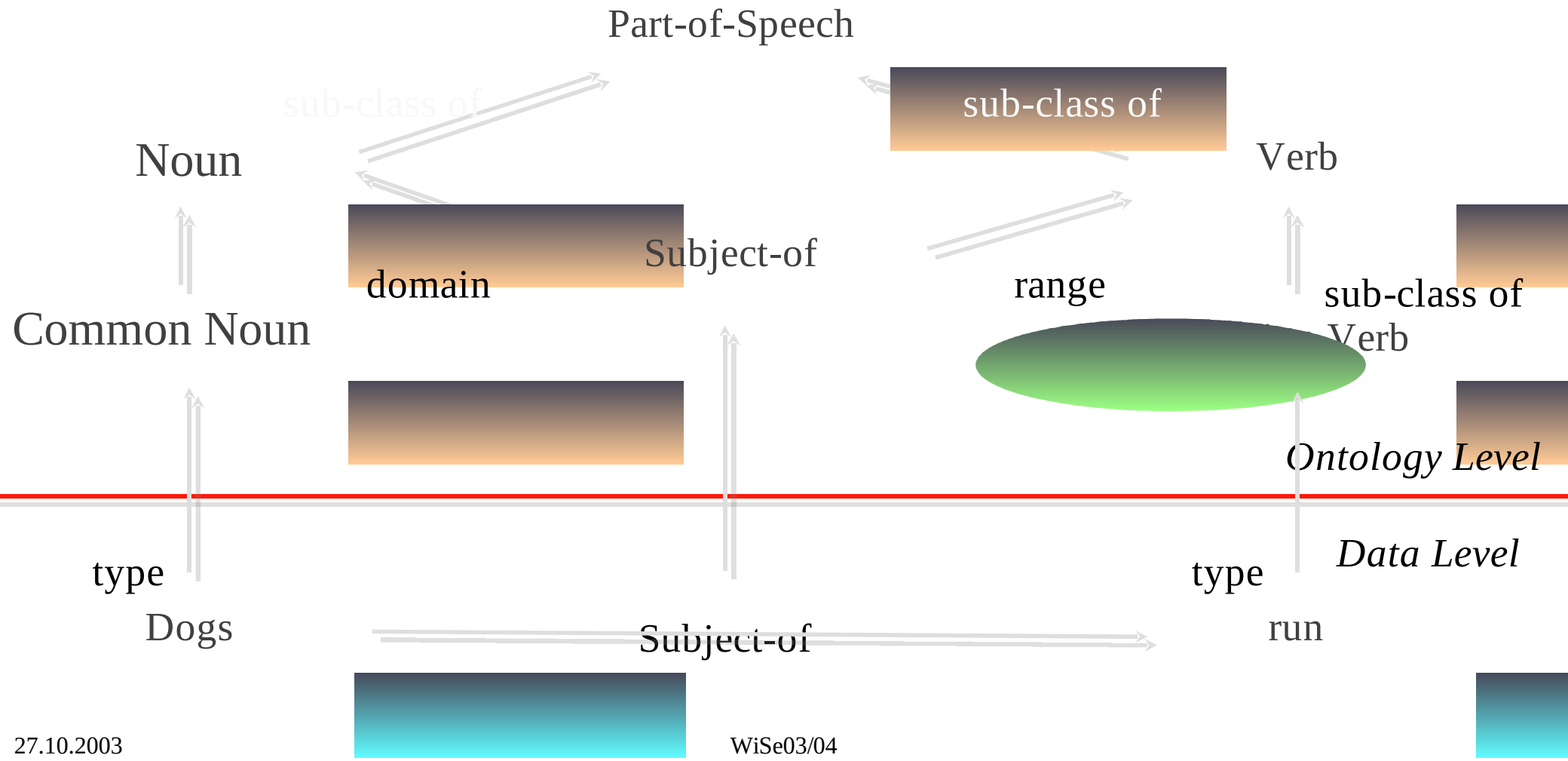
Ontologien und Ontologiesprachen

- Konzeptuelle Beschreibungen von Terminologien in verschiedenen Domänen
- benutzt um RDF schemen zu erweitern

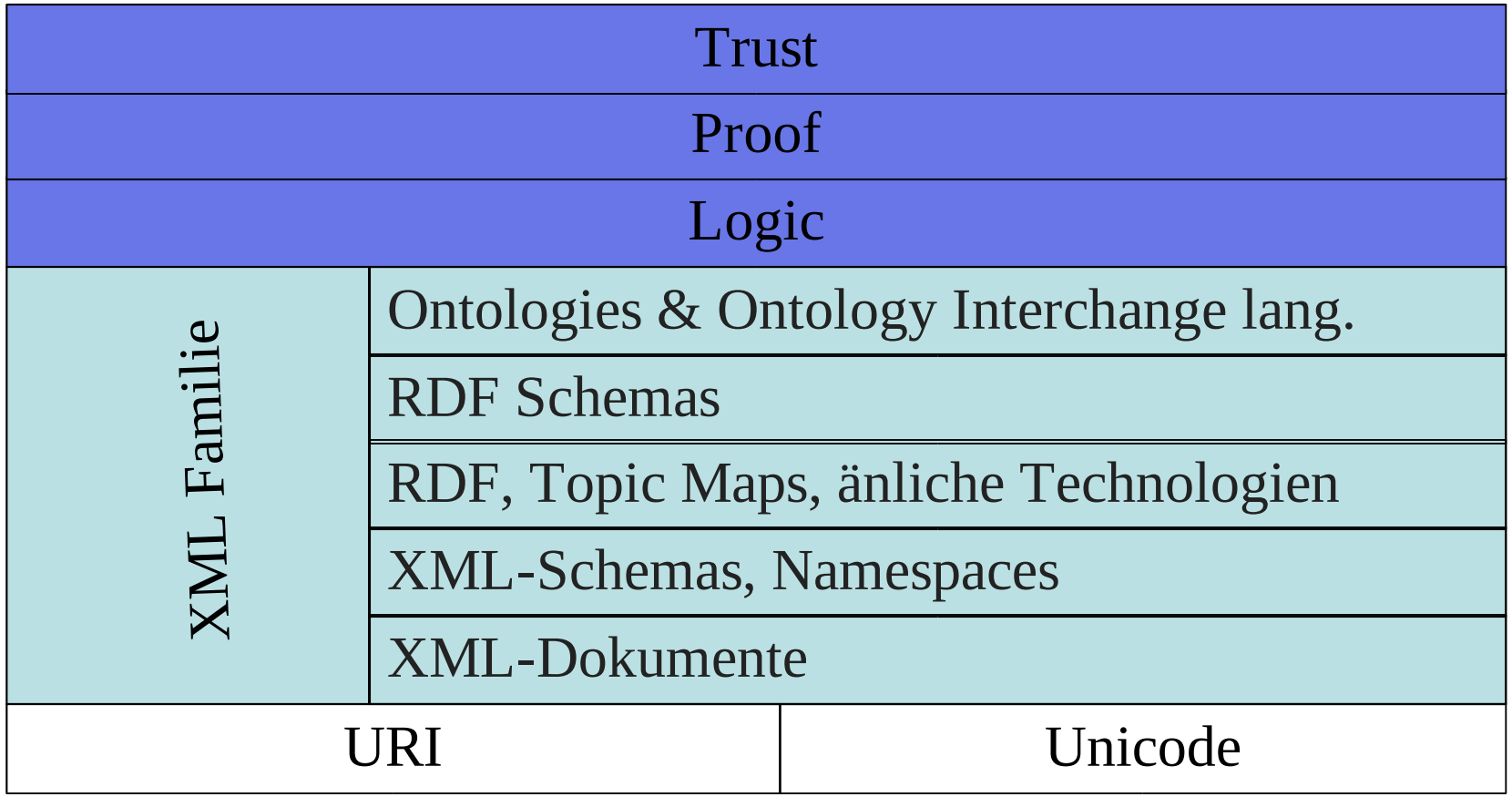
Beispiele:

- Dublin Core
- Common Bussiness Library (CBL)
- Commerce XML (cXML)

Ontologien -Beispiel



„Layer-cake“ Architektur (nach Tim Berns-Lee)-3-



Zukünftige Aktivitäten für das Semantic Web

- Logic-Ebene : Regeln für Wissensinferenz und Folgerungen
- Proof-ebene : Entwicklung von Sprachen und Mechanismen um Ressourcen mit untersiedlichen „Konfidenz“-Werten zu unterscheiden
- Trust-ebene: autonome Software Agenten werden wichtige Aktionen selbstständig durchführen