

TRIPLE—A Query Language for the Semantic Web



Michael Sintek

sintek@dfki.de

DFKI GmbH



Stefan Decker

stefan@db.stanford.edu

Stanford University Database Group

ICEC'2001, Workshop on *Semantic Web-based
E-Commerce and Rules Markup Languages*
Vienna, November 2nd, 2001

A Very Short Introduction to RDF (“Resource Description Framework”) I

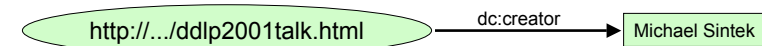
- (Syntactical) basis of the “semantic web”
- Similar to semi-structured data: graphs (RDF ~ OEM)
- Three basic object types:
 - *resources*: all things being described; named by URIs
 - *properties*: attributes to describe a resource
 - *statements*: subject + predicate + object;
are all resources (object can also be a literal)
- Example:
 - “the *creator of this talk* is *Michael Sintek*”
 - subject: <http://www.dfki.uni-kl.de/.../ddlp2001talk.html>
 - predicate: <http://www.purl.org/dc/.../creator>
 - object: “Michael Sintek”

Overview

- Introductions to
 - RDF, RDF Schema
 - DAML
- TRIPLE
 - Motivation
 - Language Description
 - Layered Architecture
 - TRIPLE₀ in RDF
 - E-Commerce Example
 - Realization
 - Conclusion

A Very Short Introduction to RDF II

- Representation
 - as graph:



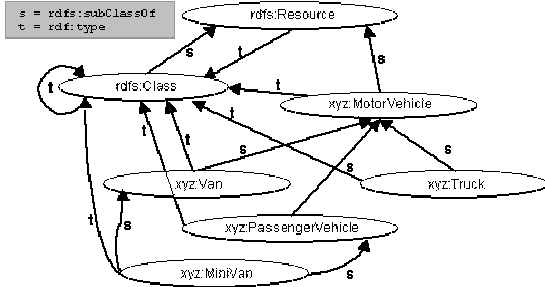
- serialized in XML (“RDF/XML”):

```

<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://www.purl.org/dc/elements/1.0/">
  <rdf:Description about="http://.../ddlp2001talk.html">
    <dc:creator>Michael Sintek</dc:creator>
  </rdf:Description>
</rdf:RDF>
  
```

A Very Short Introduction to RDF Schema

- *RDF* = simple data model
- *RDF Schema* allows definition of vocabularies for RDF data
- Simple frame system / ontology language:
 - classes, subclasses, properties, sub-properties, domain, range
- Extension of *RDF in RDF*
- Example:



TRIPLE: Motivation

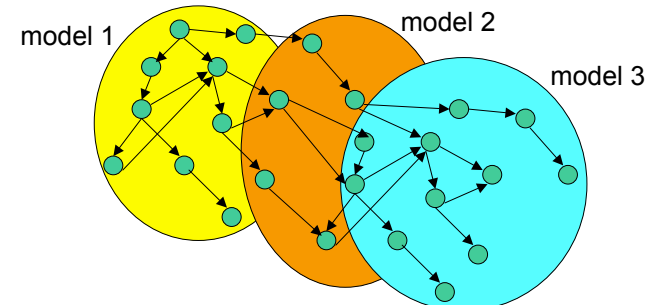
- *RDF* used in various scenarios:
 - meta-data for documents on the web (e.g., in PDF files)
 - distributed knowledge (e.g., ontologies)
 - exchange of complex semi-structured or object-oriented data (e.g., between companies)
 - message content language in agent systems (e.g., FIPA)
- Needed: query and inference language for *RDF*:
 - intelligent information retrieval (search heuristics etc.)
 - ontology mapping, information integration, ...
- Existing/ongoing approaches:
 - SiLRI, DQL, RQL, N3, Squish, ...

A Very Short Introduction to DAML (“DARPA Agent Markup Language”)

- Motivation:
 - support WWW content that is easily used by intelligent agents and other programs
 - enable the *Semantic Web*
- DAML language (“DAML+OIL”) extension of *RDF[S]*:
 - description logics:
 - class expressions: union, intersection, complement
 - inverse, transitive, ... properties
 - cardinality constraints
 - support for concrete types (from XML Schema)
 - usually enacted by DL classifier (e.g., FaCT)

What’s Wrong With Existing Approaches?

- Built-in *semantics* (e.g. SiLRI, RQL)
 - but: many *RDF*-based languages with different semantics (DAML+OIL, *RDF Schema*, UML/*RDF*, ...)
- No support for *RDF models*
 - one large heap of *RDF* data



TRIPLE: Language Overview

- Native support for
 - Resources & namespaces, abbreviations
 - Models (sets of RDF statements)
 - Reification
- Rules with expressive bodies (full FOL syntax)
- Transformations
- Syntactical extension of Horn Logic
- Syntactically similar to F-Logic (and SiLRI):
 - $subject[predicate \rightarrow object]$ (“molecule”)
- has both ASCII and RDF syntax

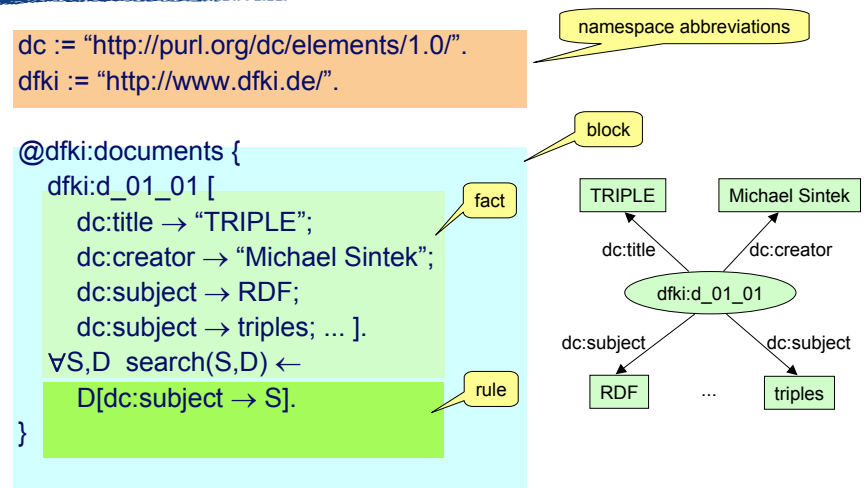
Language Description II

- Reification:
 - $stefan[believes \rightarrow \langle Ora[isAuthorOf \rightarrow homepage] \rangle]$
- Logical formulae:
 - usual logical connectives and quantifiers: $\wedge \vee \neg \forall \exists$
 - all variables introduced via \forall (or \exists)
- Clauses:
 - facts: $s[p_1 \rightarrow o_1; p_2 \rightarrow o_2; \dots]$.
 - rules: $\forall X s_1[p_1 \rightarrow X] \leftarrow s_2[p_2 \rightarrow X] \wedge \dots$
- Blocks:
 - $@model \{ clauses \}$
 - $\forall Mdl @model(Mdl) \{ clauses \}$

Language Description I

- Namespace and resource abbreviations:
 - $rdf := \text{“http://www.w3.org/1999/02/22-rdf-syntax-ns#”}$.
 - $isa := rdf.subClassOf$.
- Statements, triples, molecules:
 - $subject[predicate \rightarrow object]$
 - $subject[p_1 \rightarrow o_1; p_2 \rightarrow o_2; \dots]$
 - $s_1[p_1 \rightarrow s_2[p_2 \rightarrow o]]$
- Models, model expressions, parameterized models:
 - $s[p \rightarrow o]@m$ “triple $\langle s, p, o \rangle$ in model m ”
 - $s[p \rightarrow o]@(m_1 \cap m_2)$ model intersection
 - $s[p \rightarrow o]@sf(m1, X, Y)$ Skolem function

Example: Dublin Core



Layered Architecture

- TRIPLE supports the definition of semantical RDF extensions in a modular way
 - RDF Schema (and other “simple” frame systems): semantics can be directly defined in TRIPLE as a parameterized model (see next slide)
 - OIL, DAML+OIL (i.e., expressive ontology languages, DL): requires interaction with foreign reasoning components (e.g., DL classifier)
- Goal: use various semantics in one inference (e.g., for information integration)

TRIPLE/RDFS: RDF Schema Model

```

rdf := 'http://www.w3.org/...rdf-syntax-ns#'.
rdfs := 'http://www.w3.org/.../PR-rdf-schema-...#'.
type := rdf:type.
subPropertyOf := rdfs:subPropertyOf.
subClassOf := rdfs:subClassOf.
FORALL Mdl @rdfschema(Mdl) {
  transitive(subPropertyOf).
  transitive(subClassOf).
  FORALL O,P,V O[P->V] <-
    O[P->V]@Mdl.
  FORALL O,P,V O[P->V] <-
    EXISTS S S[subPropertyOf->P] AND O[S->V].
  FORALL O,P,V O[P->V] <-
    transitive(P) AND
    EXISTS W (O[P->W] AND W[P->V]).
  FORALL O,T O[type->T] <-
    EXISTS S (S[subClassOf->T] AND O[type->S]).
}
    
```

namespace abbreviations

resource abbreviations

model block

“copy” triples from Mdl

transitivity of subPropertyOf and subClassOf

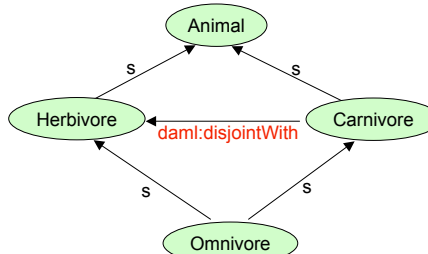
TRIPLE/DAML+OIL

- daml_oil(Mdl) model realized by accessing a DL classifier (e.g., FaCT)
- access only allowed in rule bodies
- results in hybrid rule language similar to Carin, but more pragmatic approach: powerful but incomplete

TRIPLE/DAML+OIL Example

```

daml := 'http://www.daml.org/.../daml+oil#'.
animals := 'http://www.example.org/animals#'.
@animals:ontology {
  animals:Animal[rdf:type -> daml:Class].
  animals:Herbivore[rdf:type -> daml:Class;
  rdfs:subClassOf -> animals:Animal].
  animals:Carnivore[rdf:type -> daml:Class;
  rdfs:subClassOf -> animals:Animal;
  daml:disjointWith -> animals:Herbivore].
  animals:Omnivore[rdf:type -> daml:Class;
  rdfs:subClassOf -> animals:Herbivore;
  rdfs:subClassOf -> animals:Carnivore].
}
FORALL Ont @check(Ont) {
  FORALL C unsatisfiable(C) <-
  C[daml:subClassOf -> daml:Nothing].
}
    
```



s = rdfs:subClassOf

find all unsatisfiable classes (will detect Omnivore)

TRIPLE₀ in RDF

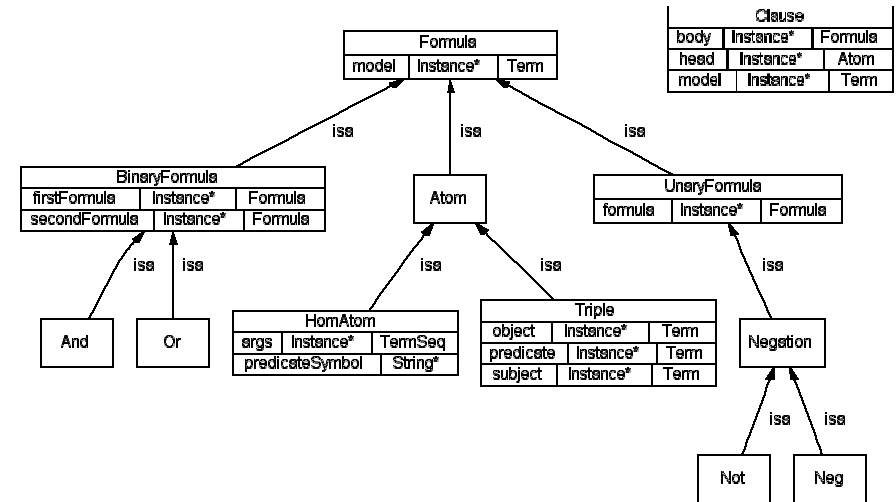
- A subset of TRIPLE, called TRIPLE₀, has a straight-forward representation in RDF
- TRIPLE₀ is (roughly) TRIPLE without quantifiers
- Definition in RDFS:

```

<rdf:Class rdf:ID="Clause"/>
<rdf:Property rdf:ID="head">
  <rdf:domain rdf:resource="&triple;Clause"/>
  <rdf:range rdf:resource="&triple;Atom"/>
</rdf:Property>
<rdf:Property rdf:ID="body">
  <rdf:domain rdf:resource="&triple;Clause"/>
  <rdf:range rdf:resource="&triple;Formula"/>
</rdf:Property>
<rdf:Property rdf:ID="model">
  <rdf:domain rdf:resource="&triple;Clause"/>
  <rdf:range rdf:resource="&triple;Term"/>
</rdf:Property>
  
```

- RDFS diagram (generated with Protégé's OntoViz):

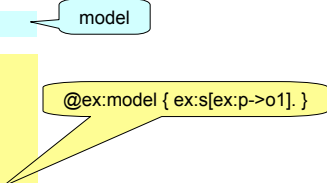
TRIPLE₀ in RDF: RDFS Diagram



TRIPLE₀ in RDF: Example

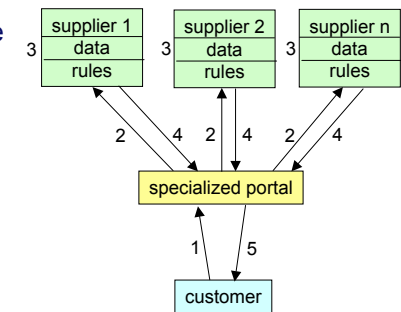
```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE rdf:RDF [
  <ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <ENTITY triple "http://www.w3.org/2001/06/30/triple#">
  <ENTITY ex "...">
]>
<rdf:RDF xmlns:rdf="&rdf;" xmlns:triple="&triple;" xmlns:ex="&ex;" xmlns="&ex;">
  <triple:SimpleModel rdf:about="&ex;model"/>
  <triple:Clause>
    <triple:head>
      <triple:Triple>
        <triple:subject>
          <triple:Resource rdf:about="&ex;s"/>
        </triple:subject>
        <triple:subject>
          <triple:Resource rdf:about="&ex;p"/>
        </triple:subject>
        <triple:predicate>
          <triple:Resource rdf:about="&ex;p"/>
        </triple:predicate>
        <triple:object>o1</triple:object>
      </triple:Triple>
    </triple:head>
    <triple:model rdf:resource="&ex;model"/>
  </triple:Clause>
</rdf:RDF>
  
```



E-Commerce Example

- 1 customer sends (complicated) preferences to portal
- 2 portal creates rules reflecting these preferences and sends them to various suppliers
- 3 suppliers enact these rules in the context of their product data and additional rules (e.g., rules determining discounts)
- 4 results are sent back (as set of facts)
- 5 portal sorts results and sends them back to customer



Realization: Mapping to Horn Logic

- First implementation (and informal semantics) by mapping to Horn Logic / XSB system (Prolog with tabled resolution)
- Lloyd-Topor transformation for quantifiers etc.
- RDF-specific transformations given as rewrite rules:

$$A : N \rightarrow \text{resource}(A, N)$$

$$O[P \rightarrow V] \rightarrow \text{statement}(O, P, V)$$

$$S @ M \rightarrow \text{true}(S, M) \text{ for statements } S$$

$$\langle S \rangle \rightarrow S \text{ for statements } S$$

$$O[P_1 \rightarrow V_1; P_2 \rightarrow V_2; \dots] @ M \rightarrow O[P_1 \rightarrow V_1] @ M \wedge O[P_2 \rightarrow V_2] @ M \wedge \dots$$

$$\text{true}(S, M_1 \cap M_2) \rightarrow \text{true}(S, M_1) \wedge \text{true}(S, M_2)$$

$$\text{true}(S, M_1 \setminus M_2) \rightarrow \text{true}(S, M_1) \wedge \neg \text{true}(S, M_2)$$

$$X := Y. S(X) \rightarrow \forall X (X = Y \wedge S(X))$$

for clause sequences $S(X)$

Conclusions

- TRIPLE is a new RDF-specific query and inference language
- Allows specification of/access to multiple semantics
- Every Horn Logic inference engine can be used
- First implementation available (for TRIPLE₀): <http://www.dfki.uni-kl.de/frodo/triple/>
- Representation of TRIPLE₀ in RDF exists
- Part of RuleML initiative: <http://www.dfki.uni-kl.de/ruleml/>