The RDF* and SPARQL* Approach to Annotate Statements in RDF and to Reconcile RDF and Property Graphs

Olaf Hartig
@olafhartig
Not supported natively in the RDF data model

- RDF triples:
  ```rdfs
  Welles name "Orson Welles" .
  Welles mentioned Kubrick .
  Kubrick name "Stanley Kubrick" .
  Kubrick influencedBy Welles .
  ??? significance 0.8 .
  ```
Main Use Case: Statement-Level Metadata


- Certainty scores
- Weights
- Temporal restrictions
- Provenance information
- etc.
Welles name "Orson Welles".
Welles mentioned Kubrick.
Kubrick name "Stanley Kubrick".
Kubrick influencedBy Welles.

s significance 0.8.
s rdf:type rdf:Statement.
s rdf:subject Kubrick.
s rdf:.predicate influencedBy.
s rdf:object Welles.
Queries?

Example 1:

List all people that Welles had a significant influence on.

```sparql
SELECT ?x WHERE {
  ?x influencedBy Welles .
  ?t significance ?sig .
  ?t rdf:type rdf:Statement .
  ?t rdf:subject ?x .
  ?t rdf:predicate influencedBy .
  ?t rdf:object Welles .
  FILTER ( ?sig > 0.7 )
}
```
Example 2:

```sparql
SELECT ?x WHERE {
  ?x influencedBy Welles .
  Welles influencedBy ?x .
  ?t1 rdf:type rdf:Statement .
  ?t1 rdf:subject ?x .
  ?t1 rdf:predicate influencedBy .
  ?t1 rdf:object Welles .
  ?t1 significance ?sig1 .
  ?t2 rdf:subject Welles .
  ?t2 rdf:predicate influencedBy .
  ?t2 rdf:object ?x .
  ?t2 significance ?sig2 .
  FILTER (?sig1 > 0.7 && ?sig2 > 0.7)
}
```
Other Proposals: Single-Triple Named Graphs

- Example:

\[ g1 \{ \text{Kubrick influencedBy Welles} \} \]
\[ g1 \text{ significance 0.8 .} \]

- Query:

```
SELECT ?x WHERE {
  GRAPH ?g {
    ?x influencedBy Welles
  }
  ?g significance ?sig .
  FILTER ( ?sig > 0.7 )
}
```
Other Proposals: Singleton Properties

• Example:

Kubrick influencedBy Welles .
Kubrick p1 Welles .
p1 singletonPropertyOf influencedBy .
p1 significance 0.8 .

- Query:

SELECT ?x WHERE {
  ?x influencedBy Welles .
  ?x ?p Welles .
  ?p singletonPropertyOf influencedBy .
  FILTER ( ?sig > 0.7 )
}

Our Proposal: Nested Triples

Kubrick influencedBy Welles.

s  rdf:type    rdf:Statement .
s  rdf:subject    Kubrick .
s  rdf:predicate    influencedBy .
s  rdf:object    Welles .
s  significance    0.8 .

<<Kubrik influencedBy Welles>>  significance  0.8

subject    predicate    object
... and Nested Triple Patterns

```
SELECT ?x WHERE {
  ?x influencedBy Welles .
  ?t significance ?sig .
  ?t rdf:type rdf:Statement .
  ?t rdf:subject ?x .
  ?t rdf:predicate influencedBy .
  ?t rdf:object Welles .
  FILTER ( ?sig > 0.7 )
}
```

```
SELECT ?x WHERE {
  <<<?x influencedBy Welles>> significance ?sig
  FILTER (?sig > 0.7)
}
```
Grouping of Patterns with the Same Subject

• By the standard SPARQL syntax, we may write:

```sparql
SELECT ?x WHERE {
    ?x influencedBy Welles .
    ?t significance ?sig ;
        rdf:type rdf:Statement ;
        rdf:subject ?x ;
        rdf: predicate influencedBy ;
        rdf:object Welles .
FILTER ( ?sig > 0.7 )
}
```

• Hence, we may easily query for multiple metadata triples:

```sparql
SELECT ?x ?sig ?src WHERE {
<<?x influencedBy Welles>> significance ?sig ;
    source ?src . }
```
Extension of the BIND Clause

- Assign matching triples to variables:

```sparql
SELECT ?x ?sig ?src WHERE {
  BIND(<<?x influencedBy Welles>> AS ?t)
  ?t significance ?sig ;
  source ?src .
}
```

```sparql
SELECT ?x ?sig ?src WHERE {
  <<?x influencedBy Welles>> significance ?sig ;
  source ?src .
}
```
Extension of the BIND Clause

• Assign matching triples to variables:

```sparql
SELECT ?x ?sig ?src WHERE {
    BIND(<<?x influencedBy Welles>> AS ?t)
    ?t significance ?sig ;
    source       ?src .
}
```

• Now, we may even output triples in query results:

```sparql
SELECT ?t ?c WHERE {
    BIND(<<?x influencedBy Welles>> AS ?t)
    ?t certainty   ?c .
}
```
Example Query 2 Revisited

SELECT ?x WHERE {
  ?x influencedBy Welles .
  Welles influencedBy ?x .
  ?t1 rdf:type rdf:Statement .
  ?t1 rdf:subject ?x .
  ?t1 rdf:predicate influencedBy .
  ?t1 rdf:object Welles .
  ?t1 significance ?sig1 .
  ?t2 rdf:subject Welles .
  ?t2 rdf:predicate influencedBy .
  ?t2 rdf:object ?x .
  ?t2 significance ?sig2 .
  FILTER (?sig1 > 0.7 && ?sig2 > 0.7)
}
Example Query 2 Revisited

```
SELECT ?x WHERE {
  ?x influencedBy Welles
  Welles influencedBy ?x .
  ?t1 rdf:type rdf:Statement .
  ?t1 rdf:subject ?x .
  ?t1 rdf:predicate influencedBy .
  ?t1 rdf:object Welles .
  ?t1 significance ?sig1 .

  ?t2 rdf:subject Welles .
  ?t2 rdf:predicate influencedBy .
  ?t2 rdf:object ?x .
  ?t2 significance ?sig2 .

  FILTER (?sig1 > 0.7 && ?sig2 > 0.7)
}
```

```
SELECT ?x WHERE {
  \<<?x influencedBy Welles>> significance ?sig1 .
  \<<Welles influencedBy ?x>> significance ?sig2 .

  FILTER (?sig1 > 0.7 && ?sig2 > 0.7)
}
```

Two Perspectives on RDF* and SPARQL*

1. Purely syntactic sugar on top of standard RDF and SPARQL
   - Can be parsed directly into standard RDF and SPARQL
   - Can be implemented easily by a small wrapper on top of any existing RDF DBMS

2. A logical model in its own right, with the possibility of a dedicated physical schema
   - Extension of the RDF data model and of SPARQL to capture the notion of nested triples
   - Supported by Blazegraph
Further Possible Applications of SPARQL*

1. Purely syntactic sugar on top of standard RDF and SPARQL

2. A logical model in its own right, with the possibility of a dedicated physical schema

- Query datasets that use RDF reification (or singleton properties, or single-triple named graphs)
  - By straightforward translation into ordinary SPARQL queries

- Query Property Graphs *including their edge properties*
  - By translation into Gremlin, Cypher, etc.
Contributions (Perspective 1)

1. Purely syntactic sugar on top of standard RDF and SPARQL

2. A logical model in its own right, with the possibility of a dedicated physical schema

- Definition of desirable properties of RDF*-to-RDF mappings
  - Information preservation and query result preservation

- Definition of RDF reification related mappings and proof that they possess the desirable properties
Contributions (Perspective 2)

1. Purely syntactic sugar on top of standard RDF and SPARQL

2. A logical model in its own right, with the possibility of a dedicated physical schema

- Formalization of the RDF* data model
  - Extends the RDF data model with the notions of an RDF* triple and an RDF* graph
- Definition of syntax and formal semantics of SPARQL*
  - Extends the semantics of SPARQL by defining the result of a SPARQL* query $Q$ over an RDF* graph $G$
    
    $$\text{eval}(Q,G) = \{ m_1, m_2, \ldots, m_n \}$$
Contributions (Perspective 2), cont’d

1. Purely syntactic sugar on top of standard RDF and SPARQL

2. A logical model in its own right, with the possibility of a dedicated physical schema

• Results regarding redundancy in RDF* graphs
  - Example:
    
    \[\angle\text{Kubrik influencedBy Welles}\rangle\text{ certainty 0.8 .}
    
    Kubrik influencedBy Welles \hspace{1cm}\text{redundant}

  - Query results are equivalent no matter whether there is redundancy in an RDF* graph or not
Contributions in Detail

1. Purely syntactic sugar on top of standard RDF and SPARQL

2. A logical model in its own right, with the possibility of a dedicated physical schema


- Olaf Hartig: “Reconciliation of RDF* and Property Graphs.” In CoRR abs/1409.3288, 2014
  - Formal definition of direct mappings b/w RDF* and Property Graphs

  - Specifies Turtle*
  - All relevant extensions of the SPARQL spec
Hold on!

... is this all theory only???
Implementations

• Full support in the Blazegraph triple store
  – “Reification done right” (RDR)

• Full support in Cambridge Semantics’ AnzoGraph

• RDF* Tools (https://github.com/RDFstar/RDFstarTools)
  – RDF* / SPARQL* extension of Apache Jena
  – Conversion tools (RDF* ↔ RDF, SPARQL* → SPARQL)
  – Simple query execution tool (SPARQL* over RDF*)

• RDF*-PG-Connection Tools (https://github.com/RDFstar/RDFstarPGConnectionTools)
  – Conversion tools (RDF* ↔ PGs)

• Other vendors have indicated interest in standardizing something along the lines of RDF* & SPARQL*
RDF* and SPARQL* in a Nutshell

<<Kubrik influencedBy Welles>> significance 0.8

\[\text{SELECT } ?x \text{ WHERE } \{ \\
\quad \text{<<}?x \text{ influencedBy Welles}}>> \text{ significance } ?\text{sig} \\
\quad \text{FILTER (} ?\text{sig} > 0.7) \\
\} \]

1. Purely syntactic sugar on top of standard RDF and SPARQL
2. A logical model in its own right, with the possibility of a dedicated physical schema