RDF(S) needs annotations

RDF Next Steps W3C Workshop

Nuno Lopes  Antoine Zimmermann  Aidan Hogan  Gergely Lukácsy  Axel Polleres  Umberto Straccia
Stefan Decker

June 26, 2010
RDF is good...

...but triples alone are often not enough:
RDF statements $s p o$ are true with respect to a certain **context**:

- **Time**: axel :worksfor :DERI true ''since 2007''
- **Provenance**: axel f:knows :ivanherman true ''in http://polleres.net/foaf.rdf''
- **f:knows**: rdfs:domain f:Person true ''in http://xmlns.com/foaf/0.1''
- **Trust/Certainty (fuzzy values)**: audiTT rdf:type :SportsCar true ''to some extent, e.g. 0.8''

...
RDF is good... 

...but triples alone are often not enough:
RDF statements s p o are true with respect to a certain context:

• Time

:axel :worksfor :DERI true ‘‘since 2007’’
RDF is good...

...but triples alone are often not enough:
RDF statements $s p o$ are true with respect to a certain **context**:

- **Time**
  
  :axel :worksfor :DERI true ‘‘since 2007’’

- **Provenance**
  
  :axel f:knows :ivanherman true ‘‘in http://polleres.net/foaf.rdf’’
  
f:knows rdfs:domain f:Person true ‘‘in http://xmlns.com/foaf/0.1’’
RDF is good... but triples alone are often not enough: RDF statements $s \ p \ o$ are true with respect to a certain context:

- **Time**
  
  :axel :worksfor :DERI true ‘‘since 2007’’

- **Provenance**
  
  :axel f:knows :ivanherman true ‘‘in http://polleres.net/foaf.rdf’’
  
  f:knows rdfs:domain f:Person true ‘‘in http://xmlns.com/foaf/0.1’’

- **Trust/Certainty (fuzzy values):**
  
  :audiTT rdf:type :SportsCar true ‘‘to some extent, e.g. 0.8’’

- etc.
RDF needs annotations...

This need comes from several sides:

- **Time**
  
  ...seems to be a practical need... Data is NOT static! some suggestions in academia [Gutierrez+ 2005] [Tappolet&Bernstein, 2009]

- **Provenance**
  
  ...seems to be a practical need... (Linked) Data is NOT universal! Named Graphs [Carroll+ 2005], Quads (Authoritative reasoning) [Hogan+ 2009]

- **Trust/Certainty (fuzzy values):**
  
  ...NOT all data is certain/trusted explored in the W3C Uncertainty Reasoning for the Web XG

- **not so new...** e.g. modules in TRIPLE
Adding information to RDF triples

Issues:

- Representation of annotations
- Semantics of annotations
Adding information to RDF triples

Issues:
- Representation of annotations
- Semantics of annotations

Our Claim:
- RDF needs agreement on representation and semantics for the most important annotation domains.
Adding information to RDF triples

- Issues:
  - Representation of annotations
  - Semantics of annotations

- Our Claim:
  - RDF needs agreement on representation and semantics for the most important annotation domains.

- One Proposal:
  - Annotated RDFS
Example: Sensor data

Even combinations of several domains may be necessary:
Example: Sensor data

Even combinations of several domains may be necessary:

sensors readings output:

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>IP Address</th>
<th>Port</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.2.15</td>
<td>4302</td>
<td>83</td>
</tr>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.3.1</td>
<td>4302</td>
<td>83</td>
</tr>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.2.6</td>
<td>4302</td>
<td>83</td>
</tr>
</tbody>
</table>
Example: Sensor data

Even combinations of several domains may be necessary:

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>IP Address</th>
<th>Port</th>
<th>Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.2.15</td>
<td>4302</td>
<td>83</td>
</tr>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.3.1</td>
<td>4302</td>
<td>83</td>
</tr>
<tr>
<td>2010-06-26</td>
<td>14:57:51</td>
<td>10.254.2.6</td>
<td>4302</td>
<td>83</td>
</tr>
</tbody>
</table>
Even combinations of several domains may be necessary:

- location of a tag in a room (pure RDF)
- time of the sensor reading (temporal annotation)
- signal strength of the sensor reading (fuzzy annotation)
Example: Sensor data

Even combinations of several domains may be necessary:

:tag4302 :locatedIn :room311 . [14:57, 15:01] [0.8]
:tag4302 :locatedIn :room310 . [15:02, 16:10] [0.7]
Example: Sensor data

Even combinations of several domains may be necessary:

:tag4302 :locatedIn :room311 . [14:57, 15:01] [0.8]
:tag4302 :locatedIn :room310 . [15:02, 16:10] [0.7]

This is not RDF
How to represent annotations?

:tag4302 :locatedIn :room311 . [14:57, 15:01]
Issue 1: Representation

:tag4302 :locatedIn :room311 . [14:57, 15:01]

Reification?

:record1 rdf:type rdf:Statement
  rdf:subject :tag4302;
  rdf:predicate :locatedIn ;
  rdf:object :room311 ;
  time:start "2010-06-26 14:57"^^xs:timeStamp;
  time:end "2010-06-26 15:01"^^xs:timeStamp .
Issue 1: Representation

Reification?

:record1 rdf:type rdf:Statement
  rdf:subject :tag4302;
  rdf:predicate :locatedIn ;
  rdf:object :room311 ;
  time:start "2010-06-26 14:57"^^xs:timeStamp;
  time:end "2010-06-26 15:01"^^xs:timeStamp .

- no semantics
Issue 1: Representation

:tag4302 :locatedIn :room311 . [14:57, 15:01]

Reification?

:record1 rdf:type rdf:Statement
    rdf:subject :tag4302;
    rdf:predicate :locatedIn;
    rdf:object :room311;
    time:start "2010-06-26 14:57"^^xs:timeStamp;
    time:end "2010-06-26 15:01"^^xs:timeStamp .

- no semantics
- not really “popular” some people prior to this WS even claimed to drop reification altogether
Issue 1: Representation

Other formats?

- N-Quads

```
:tag4302 :locatedIn :room311 . [14:57, 15:01]
_:c
_:c time:start "2010-06-26 14:57"^^xs:timeStamp ;
  time:end "2010-06-26 15:01"^^xs:timeStamp .
```

- alternatively TriG, TriX

- non-standard (yet)

- semantics of annotations still not clear
Issue 2: Semantics

- What do annotations mean for RDF(S) semantics?
- How to combine non-annotated and annotated RDF semantically?
Issue 2: Semantics

- What do annotations mean for RDF(S) semantics?
- How to combine non-annotated and annotated RDF semantically?

:axel f:knows :ivanherman . [http://polleres.net/foaf.rdf]
f:knows rdfs:domain f:Person . [http://xmlns.com/foaf/0.1]
Issue 2: Semantics

- What do annotations mean for RDF(S) semantics?
- How to combine non-annotated and annotated RDF semantically?

```plaintext
:axel f:knows :ivanherman . [http://polleres.net/foaf.rdf]
f:knows rdfs:domain f:Person . [http://xmlns.com/foaf/0.1]
:axel rdf:type f:Person . [???]
```
Issue 2: Semantics

- What do annotations mean for RDF(S) semantics?
- How to combine non-annotated and annotated RDF semantically?

```rml
:worksFor rdfs:domain :Employee .
:axel rdf:domain :Employee [???]
```
Our approach – Annotated RDF

[Straccia+, AAAI2010] Generic Framework to

1. describe annotation domains
2. give them a semantics
3. live side-by-side with non-annotated RDF
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any **annotation domain** consists of a lattice:

- the *representation* of the annotations
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:
- the representation of the annotations: [14:35, 14:57]
- an order between the elements
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
- an order between the elements: ⊆
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the \textit{representation} of the annotations: \([14:35, 14:57]\)
- an \textit{order} between the elements: \(\subseteq\)

\textit{universal} \((\top)\) and \textit{empty} \((\bot)\) annotations
Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the *representation* of the annotations: [14:35, 14:57]
- an *order* between the elements: $\subseteq$

universal ($\top$) and empty ($\bot$) annotations: $\top = [\neg\infty, +\infty]$ $\bot = []$
Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
- an order between the elements: \( \subseteq \)

universal (\( \top \)) and empty (\( \bot \)) annotations: \( \top = [-\infty, +\infty] \) \( \bot = [] \)

operator (\( \otimes \)) is a so-called t-norm
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
- an order between the elements: \( \subseteq \)

universal (\( \top \)) and empty (\( \bot \)) annotations: \( \top = [-\infty, +\infty] \)
operator (\( \otimes \)) is a so-called t-norm: \( \cap \)
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]  
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any **annotation domain** consists of a lattice:

- the **representation** of the annotations:  
  \[14:35, 14:57]\n
- an **order** between the elements:  \[\subseteq\]

  - *universal* (\(\top\)) and *empty* (\(\bot\)) annotations:  
    \(\top = [-\infty, +\infty] \; \bot = []\)

  - operator (\(\otimes\)) is a so-called t-norm:  \(\cap\)

Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
- an order between the elements: ⊆

universal (∧) and empty (⊥) annotations: T = [−∞, +∞] ⊥ = []
operator (⊗) is a so-called t-norm: ∩
operator (∨) for combining annotations
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any **annotation domain** consists of a lattice:

- the *representation* of the annotations: [14:35, 14:57]
- an *order* between the elements: ⊆

*universal* (⊤) and *empty* (⊥) annotations:  ⊤ = [−∞, +∞] ⊥ = []

operator (⊗) is a so-called t-norm: ∩
operator (∨) for *combining* annotations: ∪
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any annotation domain consists of a lattice:

- the representation of the annotations: [14:35, 14:57]
- an order between the elements: ⊆

universal (⊤) and empty (⊥) annotations: \( \top = [-\infty, +\infty] \), \( \bot = [] \)

operator (⊗) is a so-called t-norm: \( \cap \)
operator (∨) for combining annotations: \( \cup \)

\[ [09:25, 11:49] \lor [10:35, 12:57] = [09:25, 12:57] \]
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . [09:25, 11:49]
:tag4302 :locatedIn :room311 . [10:35, 12:57]

Any **annotation domain** consists of a lattice:

- the *representation* of the annotations: \([14:35, 14:57]\)
- an *order* between the elements: \(\subseteq\)

*universal* (\(\top\)) and *empty* (\(\bot\)) annotations: \(\top = [-\infty, +\infty], \bot = []\)

operator (\(\otimes\)) is a so-called t-norm: \(\cap\)
operator (\(\lor\)) for combining annotations: \(\cup\)

\([09:25, 11:49] \lor [14:35, 15:57] = [09:25, 11:49], [14:35, 15:57]\)
Annotation Domains

Temporal domain example:

:tag4302 :locatedIn :room311 . {[09:25, 11:49]}
:tag4302 :locatedIn :room311 . {[10:35, 12:57]}

Any annotation domain consists of a lattice:
- the representation of the annotations: \{[14:35, 14:57]\}
- an order between the elements: \(\subseteq\)

universal \(\top\) and empty \(\bot\) annotations: \(\top = \{(-\infty, +\infty]\) \(\bot = \{\}\)
operator \(\otimes\) is a so-called t-norm: \(\cap\)
operator \(\lor\) for combining annotations: \(\cup\)

Other domains: Examples

Trust/Fuzzy

:tag4302 :locatedIn :room311 . 0.9
:tag4302 :locatedIn :room310 . 0.2

annotations: [0,1]
order: ≤
⊗: min ⊕: max
⊤ = 1, ⊥ = 0
Other domains: Examples

Trust/Fuzzy

\[
\begin{align*}
\text{:tag4302 :locatedIn :room311 . 0.9} \\
\text{:tag4302 :locatedIn :room310 . 0.2}
\end{align*}
\]

annotations: \([0,1]\)

order: \(\leq\)

\(\otimes: \text{min} \quad \lor: \text{max}\)

\(\top = 1, \quad \bot = 0\)

Provenance

\[
\text{:axel rdf:type Person .} \\
[\text{xmlns.com/foaf/0.1/ \^ polleres.net/foaf.rdf}]
\]

annotations: \(\text{prop.}\)

formulae in DNF over URIs

order: \(\models\)

\(\otimes: \land \quad \lor: \lor\)

\(\top = \text{disj. of all URIs}, \quad \bot = \text{conj. of all URIs}\)
Other domains: Examples

Trust/Fuzzy

<table>
<thead>
<tr>
<th>Annotation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>:tag4302 :locatedIn :room311</td>
<td>0.9</td>
</tr>
<tr>
<td>:tag4302 :locatedIn :room310</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Provenance

<table>
<thead>
<tr>
<th>Annotation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>:axel rdf:type Person</td>
<td></td>
</tr>
<tr>
<td>[xmlns.com/foaf/0.1/ ∧ polleres.net/foaf.rdf]</td>
<td></td>
</tr>
</tbody>
</table>

Annotations: \([0,1]\)

Order: \(\leq\)

\(\otimes: min\) \(\vee: max\)

\(\top = 1, \bot = 0\)
Other domains: Examples

Trust/Fuzzy

\[
\text{:tag4302 :locatedIn :room311 . 0.9} \\
\text{:tag4302 :locatedIn :room310 . 0.2}
\]

annotations: \([0,1]\)
order: \(\leq\)
\(\otimes: \text{min} \; \vee: \text{max}\)
\(\top = 1, \; \bot = 0\)

Provenance

\[
\text{:axel rdf:type Person .} \\
[\text{xmlns.com/foaf/0.1/ \^ polleres.net/foaf.rdf}]
\]

annotations: \(\text{prop.}\)
formulae in DNF over URIs
order: \(\models\)
\(\otimes: \land \; \vee: \lor\)
\(\top = \text{disj. of all URIs}, \; \bot = \text{conj. of all URIs}\)

Our generic semantics allows to combine domains:

\[
\text{:tag4302 :locatedIn :room311 . ([14:25, 14:57], 0.8)}
\]
Integration with RDF

Transparent integration of annotated and classical RDF

```
:stefan foaf:name "Stefan Decker" .
:tag4302 :assignedTo :stefan .
:tag4302 :locatedIn :room311 .  [14:25, 14:57]
```
Integration with RDF

Transparent integration of annotated and classical RDF

:stefan foaf:name "Stefan Decker" . \([-\infty, +\infty]\]
:tag4302 :assignedTo :stefan . \([-\infty, +\infty]\]
:tag4302 :locatedIn :room311 . \([14:25, 14:57]\]

Possible approaches:
- use $\top$ as annotation
Integration with RDF

Transparent integration of annotated and classical RDF

```
:stefan foaf:name "Stefan Decker" . [_:a, _:b]
:tag4302 :assignedTo :stefan . [_:a, _:b]
:tag4302 :locatedIn :room311 . [14:25, 14:57]
```

Possible approaches:

- use ⊤ as annotation
- triple is valid at a time interval common throughout the graph
  requires blank nodes in annotations
Integration with RDF

Transparent integration of annotated and classical RDF

Possible approaches:

- use $\top$ as annotation
- triple is valid at a time interval common throughout the graph requires blank nodes in annotations
- triple is valid until “now” ([Temporal RDF, Gutierrez et al, 2005]) represents current time
Integration with RDF

Transparent integration of annotated and classical RDF

:stefan foaf:name "Stefan Decker" . \([-\infty, +\infty]\]
:tag4302 :assignedTo :stefan . \([-\infty, +\infty]\]
:tag4302 :locatedIn :room311 . \([14:25, 14:57]\]

Possible approaches:

- use $\top$ as annotation “upwards compatible”
- triple is valid at a time interval common throughout the graph
  requires blank nodes in annotations
- triple is valid until “now” ([Temporal RDF, Gutierrez et al, 2005])
  represents current time
Inference rules are **independent** of the annotation domain
Inference rules are **independent** of the annotation domain

- RDFS “rdfs:domain” rule:

  ```
  ?p rdfs:domain ?c
  ?s ?p ?o
  \(\Rightarrow\) ?s rdf:type ?c
  ```
Inference rules are **independent** of the annotation domain.

- **RDFS “rdfs:domain” rule:**
  
  ```
  ?p rdfs:domain ?c
  ?s ?p ?o
  ⇒ ?s rdf:type ?c
  ```

  **Example:**

  ```
  :worksFor rdfs:domain :Employee
  :nuno :worksFor :DERI
  ⇒ :nuno rdf:type :Employee
  ```
Inference rules are **independent** of the annotation domain

- **Annotated RDFS “rdfs:domain” rule:**

  \[ \begin{align*}
  \text{?p} & : \text{rdfs:domain} \quad \text{?c} & : \text{?v1} \\
  \text{?s} & : \text{?p} \quad \text{?o} & : \text{?v2} \\
  \Rightarrow & \quad \text{?s} : \text{rdf:type} \quad \text{?c} & : \text{?v1} \otimes \text{?v2}
  \end{align*} \]

Example:

\[ \begin{align*}
:\text{worksFor} & : \text{rdfs:domain} : \text{Employee} \\
:\text{nuno} & : \text{worksFor} : \text{DERI} \\
\Rightarrow & \quad :\text{nuno} : \text{rdf:type} : \text{Employee}
\]
Annotated RDFS Inference rules

Inference rules are **independent** of the annotation domain

- Annotated RDFS “rdfs:domain” rule:

  \[
  \begin{align*}
  & p \ rdfs:domain \ c \quad \forall v_1 \\
  & s \ p \ o \quad \forall v_2 \\
  \Rightarrow & \ s \ rdf:type \ c \quad \forall v_1 \otimes \forall v_2
  \end{align*}
  \]

Example:

:worksFor rdfs:domain :Employee \([-\infty, +\infty]\)  
:nuno :worksFor :DERI \["2009-01-01", "2010-06-26"]  
\Rightarrow :nuno rdf:type :Employee \["2009-01-01", "2010-06-26"]
Annotated RDFS Inference rules

Inference rules are **independent** of the annotation domain

- **Annotated RDFS “rdfs:domain” rule:**
  
  \[
  \begin{align*}
  & ?p \text{ rdfs:domain } ?c \quad ?v1 \\
  & ?s \ ?p \ ?o \quad ?v2 \\
  \Rightarrow \ ?s \ \text{ rdf:type } ?c \quad ?v1 \ \otimes \ ?v2
  \end{align*}
  \]

**Example:**

\[
\begin{align*}
:\text{worksFor} & \text{ rdfs:domain } :\text{Employee} \quad [\infty, +\infty] \\
:nuno & :\text{worksFor} :\text{DERI} \quad ["2009-01-01", "2010-06-26"] \\
\Rightarrow \ & :nuno \ \text{ rdf:type } :\text{Employee} \quad ["2009-01-01", "2010-06-26"]
\end{align*}
\]

- **Extra rule to group annotations triples (\(\lor\)):**
  
  \[
  \begin{align*}
  & ?s \ ?p \ ?o \quad \lambda_1 \\
  & ?s \ ?p \ ?o \quad \lambda_2 \\
  \Rightarrow \ ?s \ ?p \ ?o \quad \lambda_1 \lor \lambda_2
  \end{align*}
  \]
Annotated RDFS Inference rules

Inference rules are **independent** of the annotation domain

- **Annotated RDFS “rdfs:domain” rule:**

  \[
  \begin{align*}
  ?p & \text{ rdfs:domain } ?c & ?v1 \\
  \Rightarrow & ?s \text{ rdfs:domain } ?c & ?v1 \otimes ?v2
  \end{align*}
  \]

**Example:**

\[
\begin{align*}
:\text{worksFor} & \text{ rdfs:domain } :\text{Employee} & [\text{\(-\infty, +\infty\)}] \\
:nuno & :\text{worksFor} :\text{DERI} & [\text{"2009-01-01", "2010-06-26"}] \\
\Rightarrow & :\text{nuno} \text{ rdfs:domain } :\text{Employee} & [\text{"2009-01-01", "2010-06-26"}]
\end{align*}
\]

- **Extra rule to group annotations triples (\(\lor\)):**

\[
\begin{align*}
:nuno & :\text{worksFor} :\text{DERI} & [\text{"2008-05-01", "2010-01-01"}] \\
:nuno & :\text{worksFor} :\text{DERI} & [\text{"2009-01-01", "2010-06-26"}] \\
\Rightarrow & :\text{nuno} :\text{worksFor} :\text{DERI} & [\text{"2008-05-01", "2010-06-26"}]
\end{align*}
\]
**Our Claim:**
- RDF needs **agreement** on representation and semantics for important annotation domains e.g. time, provenance, trust

**Representational Issues:**
- several options (reification, N-quads, TriG/X)
- reification the only standards compliant thus far, sub-optimal

**Semantics of annotations:**
- Our proposal: Annotated RDFS
  - allows arbitrary ordered annotation domains
  - give them a semantics on top of RDFS
  - live side-by-side with non-annotated RDF
  - SPARQL(1.1) compatible...

**TODO for us here?**
- At the least: Representation to add context to triples
- Needs to be “upwards-compatible”
- wish-list: tackle semantic vacuum on context for important domains (e.g., time, provenance, trust/fuzzy)
Annotated SPARQL:

Extend SPARQL to allow querying annotated RDF

- “Annotation aware” SPARQL
Annotated SPARQL:

Extend SPARQL to allow querying annotated RDF

- “Annotation aware” SPARQL

  “Where was Stefan between 14:30 and 15:00?”

```
SELECT ?Room WHERE {
  ?Tag :assignedTo :stefan ;
  :locatedIn ?Room . ["14:30", "15:00"]
}
```
Annotated SPARQL:

Extend SPARQL to allow querying annotated RDF

- “Annotation aware” SPARQL

  “Where was Stefan between 14:30 and 15:00?”

```sparql
SELECT ?Room WHERE {
  ?Tag  :assignedTo  :stefan ;
  :locatedIn   ?Room . ["14:30", "15:00"]
}
```

- Evaluation based on an extension of the SPARQL relational algebra to support annotations
“When were Stefan and Axel in the same room?”

```
SELECT ?Room ?TimeInterval WHERE {
  ?Tag1 :assignedTo :stefan ;
  :locatedIn ?Room . ?TimeInterval
  ?Tag2 :assignedTo :axel ;
  :locatedIn ?Room . ?TimeInterval
}
```
"When were Stefan and Axel in the same room?"

```
SELECT ?Room ?TimeInterval WHERE {
  ?Tag1 :assignedTo :stefan ;
  :locatedIn ?Room . ?TimeInterval
  ?Tag2 :assignedTo :axel ;
  :locatedIn ?Room . ?TimeInterval
}
```

Answers:

```
(?Room, ?TimeInterval) = (:room311, {
  ["09:13", "10:35"],
  ["11:23", "12:47"]})

(?Room, ?TimeInterval) = (:conferenceRoom, {
  ["14:25", "14:57"]})
```