Chris Bizer, Freie Universität Berlin, chris@bizer.de
Jeremy Carroll, Hewlett-Packard Laboratories Bristol, Jeremy.Carroll@hp.com

2004/02/27

Modelling Context using Named Graphs

Abstract

In this document we describe our ongoing work on using named graphs for modelling context within Semantic Web applications. We give a short introduction into named graphs and define our understanding of context. Afterwards we describe several use cases which show how named graphs could be utilised for capturing context.

1 Named Graphs 2
1.1 Syntaxes for Named Graphs 2
1.2 Querying Named Graphs 3
2 Context 3
3 Use Cases 3
3.1 Data Syndication 3
3.2 Signing RDF graphs 4
3.3 Scoping Assertions 5
3.4 Defining Access Rights 6
3.5 Expressing Privacy Preferences 6
3.6 Modelling Believes 7
3.7 Trust Evaluations 7
4 Current Design Issues 8
5 Related Work 9
6 Glossary 10

1 Named Graphs

The concept of Named Graphs is defined in [CarrollStrickler04, Section 6]. Graph are named either with global names by the use of an uri element or with file scoped bnodes, by the use of an optional id element. To say anything about the graph, e.g. provenance information, some triples are needed that involve the graph name. These triples can be included within the graph, which then includes assertions about itself, or they can be in a separate graph in the same document, or they can be in a separate document (which requires the use of a uri node naming the graph). Named graphs differ from merely extending RDF triples to RDF quads, in that the full extent of the graph is known, and the graph is not treated with the open world assumption. Unlike a subject resource, which may have additional properties not mentioned in a document, the assertion of a named graph asserts that this graph is exactly the triples given, and there are not any others that have been omitted.

1.1 Syntaxes for Named Graphs

There are currently two syntaxes for named graphs. The xml-based TriX syntax defined in [CarrollStrickler04] and a textual non-XML syntax called TriG, which will be used for the examples in this document.

Example TriG document:

```
@prefix rdf:        <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix dc:         <http://purl.org/dc/elements/1.1/>
@prefix ex:         <http:// www.example.org/vocabulary#>
@baseURI            <http://www.example.org/exampleDocument#>

G1 (Monica ex:name “Monica Murphy“.
    Monica rdf:type ex:Person .
    Monica ex:homepage <http://www.monicamurphy.org>)

_:G2 (Monica ex:hasSkill ex:Programming)

G3 (G1 dc:author Chris.
    G1 dc:date “2/10/2003”.
    _:G2 dc:author Chris.
    _:G2 dc:date “2/9/2003”.
    G3 dc:author Chris.
    G3 dc:date “2/2/2004”)
```

The document contains three graphs. The statements within a graph are enclosed by round brackets, not to be mixed up with the N3 list syntax. The first graph in the example is named `http://www.example.org/exampleDocument#`G1. The second graph is named with the bnode _:G2. The third graph contains provenance information about the other graphs and about itself.

1.2 Querying Named Graphs

For querying named graphs we use TriQL a graph patterns based query language inspired by Andy Seaborne's RDQL. A graph pattern consists of an optional graph name and a set of triple patterns.

Example Query: Select all Persons together with the URL of their homepage, which have the skill programming, using only information which has been published after "1/1/2003".

```
SELECT ?x, ?y
WHERE
     ?a (?b rdf:type ex:Person .
         ?b rdf:hasName ?x .
         ?b ex:homepage ?y)
     (?a dc:date ?c)
AND  ?c > "1/1/2003"
```

The example query uses two graph patterns. The first uses the variable ?a to refer to graph names and three triple patterns. In the second graph pattern the graph name is abbreviated because it doesn't matter. Queries are executed across all graphs in a document or in a repository.

2 Context

The question "What is a context?" has been discussed in artificial intelligence for at least twenty years without much agreement. So using the term "Context" usually leads to general confusion because of a lack of shared conceptualization.

Within the Semantic Web Community, work on context if found in [Guha95], [Klyne00], [Klyne02], [MacGregorKo03] and [Beckett03]. Current RDF software, which implements some notion of context, usually sees context as an identifiable resource which somehow groups or "surrounds" statements, statings or models.

We use the term "Context" in the following sense: The context of a statement is the graph in which it is contained.

3 Use Cases

In this section we give several examples how named graphs could be used for modeling context.

3.1 Data Syndication

Requirements

- Provenance tracking, e.g. source, author, date.
- Provenance chains: "A said that B said that C."

Related work on provenance is found in [Dumbill03] and [Prudhommeaux02].

Example: Simple Provenance Tracking

```
G1 (Monica ex:name “Monica Murphy“.
    Monica rdf:type ex:Person)

G2 (G1 dc:author Chris.
    G1 dc:date “2/10/2004”)

G3 (Monica ex:hasSkill ex:Programming)

G4 (G3 dc:author Peter.
    G3 dc:date “2/3/2004”)
```

Example Query: Find all information stated by Chris.

```
SELECT ?x ?y ?z
WHERE
   ?a (?x ?y ?z)
      (?a dc:author Chris)
```

Example: Provenance Chains
Peter states, that Chris said that Andy said, that Monica Murphy is a person.

```
G1 (Monica ex:name “Monica Murphy“.
    Monica rdf:type ex:Person)

G2 (G1 ex:saidby Andy.
    G1 ex:SourceURL Doc1.trix.
    G1 dc:date “2/10/2004”)

G3 (G2 ex:saidby Chris.
    G2 ex:SourceURL Doc2.trix.
    G2 dc:date “2/10/2004”)

G4 (G1 dc:author Peter.
    G2 dc:author Peter.
    G3 dc:author Peter)

G5 (G4 dc:author Peter.
    G4 dc:date “2/10/2004”)
```

3.2 Signing RDF graphs

Related work on signing RDF graphs: [Carroll03].

Example: Provenance and Signing

```
G1 (Monica ex:hasStatus Admin.
    Monica rdf:type ex:Person.
    G1 ex:author Andy.
    G1 dc:date “2/10/2004”)

G2 (G1 ex:hasSignature “xd2shfl22k4jdsre…”.
    G1 ex:Signer Andy)

G3 (Andy ex:publicKeyURL http://bla.bla.bla)
```

Example Query: Get me all authors of the stateing “Monica ex:hasStatus Admin” together with their public key and their signature of the graph which contains “Monica ex:hasStatus Admin”.

```
SELECT ?a ?b ?c ?d
WHERE
?d (Monica ex:hasStatus Admin)
   (?d dc:signer ?a)
   (?a ex:publicKeyURL ?b)
   (?d ex:hasSignature ?c)
```

3.3 Scoping Assertions

Requirements

- Something like N3 formula
- Decontextualization and lifting rules

Related work: [Guha95], [Klyne00], [Klyne02].

The examples in this section are taken from [Klyne02].
It is not clear to us whether these work.

1. Graham’s Metal/Water Example:

```
A consistsOf Metal .
B consistsOf Water .
{ Metal denserThan Water } log:implies { A sinksIn B } .
```

translates to

```
G1 (A consistsOf Metal .
    B consistsOf Water .
    _:G2 log:implies _:G3)

_:G2 (Metal denserThan Water)
_:G3 (A sinksIn B)
```

2. Graham’s Combining Different Theories Example:

```
{ Mass a FixedValue } in NewtonianMechanics .
{ Mass a Variable } in RelativityTheory .
{ RelativityTheory approximates NewtonianMechanics }
  when { RelativeVelocities lessThan halfC } .
```

translates to

```
G1 (_:G2 in NewtonianMechanics .
    _:G3 in RelativityTheory .
    _:G4 when _:G5)
_:G2 (Mass a FixedValue)
_:G3 (Mass a Variable)
_:G4 (RelativityTheory approximates NewtonianMechanics)
_:G5 (RelativeVelocities lessThan halfC)
```

4. Personal Theories as a Combination of Scoping and Provenance

```
G1 (A consistsOf Metal .
    B consistsOf Water .
    Metal denserThan Water)

G2 (A sinksIn B)
G3 (A swimsIn B)

G4 (G1 log:implies G2)
G5 (G1 log:implies G3)

G6 (G1 dc:author Chris.
    G2 dc:author Chris.
    G4 dc:author Chris)

G7 (G1 dc:author Peter.
    G3 dc:author Peter.
    G5 dc:author Peter)
```

3.4 Defining Access Rights

Related work: [Intellidimension03].

```
G1 (Monica ex:hasStatus Admin.
    Monica rdf:type ex:Person.
    G1 requiresAccessRight Admin)
```

3.5 Expressing Privacy Preferences

Related work: [Kim02], [McBride02].

```
G1 (Monica ex:name “Monica Murphy“.
    Monica rdf:type ex:Person.
    G1 ex:allowedUsage ex:AllPurposes)

G2 (Monica ex:eMail <mailto:monica@murphy.org>.
    G2 ex:disallowedUsage ex:marketing)
```

3.6 Modelling Believes

Example: Peter wants to state that he doesn't believe Chris' stating, that Monica has the skill programming.

```
G1 (Monica ex:name "Monica Murphy".
    Monica rdf:type ex:Person.
    Monica ex:hasSkill ex:Programming.
    G1 dc:author Chris)

G2 (Monica ex:hasSkill ex:Programming)
G3 (G2 ex:truthValue ex:false)

G4 (G2 dc:author Peter.
    G3 dc:author Peter)
```

Peter has stated that he doesn't believe that Monica has the skill programming. Has he also stated that he doesn't believe Chris' stating?

3.7 Trust Evaluations

Related work: [Bizer04].

Statings published on the Semantic Web have to be seen as claims rather than as facts. Using information found on the Semantic Web, an information consumer will have to decide

1. to which degree he trusts information providers to share the same conceptualization with him and
2. to which degree he trusts the information itself.

Thus we have the understanding, that a stating is neither asserted nor unasserted. It is uncertain. The stating becomes asserted to a user through a trust decision.
These trust decision are subjective and usually relay on task-specific trust policies in the offline world. One the Semantic Web, trust decisions can be based on:

- Reputational metainformation in the form of information- or information source ratings provided by other information consumers or known from past interactions.
- Provenance metainformation about the context in which a stating has been claimed, e.g. who said what, when and why. Meta-information about the author could be used for role-based trust policies like "Distrust everything a vendor says about its competitor."
- Related information about the same topic provided by other authors, which could be used for policies like "Believe information which has been stated by at least 5 independent sources."

The future Semantic Web Trust layer will require fine-grained provenance and reputation tracking. Named graphs might be a suitable data model for capturing the necessary background information.

4 Current Design Issues

Our work on named graphs and their application within Semantic Web applications is still in an early stage. In this section we describe open questions and current design issues.

Relation between Document, Graphset and Graph

The graphset element in trix is merely a syntactic necessity for XML documents, and should not convey any meaning. If we want to talk about a collection in RDF there are plenty of mechanisms. So if we want to talk about a collection of graphs we use one of those. Thus, the graphset tag should be changed to something semantic-free (e.g. trix). Another point is that one is tempted to use the document URL as graphset URI, e.g.

```
Doc1.trix: G1 (S P O) G2 (S’ P’ O’)
Doc2.trix: G3 (Doc1.trix truthValue false)
```

We think that a clear distinction between content level (graphs) and distribution level (documents/services) and a clear distinction between URIs (naming) and URLs (retrieving) is needed. Thus the URL used to retrieve a trix document refers to the document and not to the contained graphs. The point here is to stop before going on the slippery slop to graphsetsetsets.

Set of Graphs versus Nested Subgraphs

In [CarrollStrickler04] all graphs in a graphset are treated equal. In N3 there is a top-level graph which can include nested subgraphs expressed with formula. We discuss if we should adopt the top-level graph idea but tend not to do so.

Literals as Subject and Blank Nodes as Predicate

The restriction in RDF that literals cannot be subjects is spurious, as shown by the semantic possibility of a literal being a subject and examples such as

```
 (_:r P O .
  _:r owl:sameAs “literal” )
```

Thus we are inclined to permit literals as subjects.
A similar argument could be made for blank nodes as predicates, but this requires new semantic machinery, and so perhaps should be avoided.

Blank Node Scope

Blank nodes currently have graph scope in RDF. In TriX they have graphset scope. In N3 they have document scope. We currently discuss the appropriate scoping for named graphs.

I
De Re versus De Dicto

In [CarrollStrickler04], the graph element has the boolean valued attribute “asserted”, which indicates if a graph is asserted or quoted. Quoted graphs are interpreted as de dicto, asserted

graphs as de re (see glossary for a definition of re re and de dicto). We currently discuss if this attribute is necessary and tend to remove it.

The arguments are that:
1. Semantic Web applications might only require de re, because communication presupposes a shared conceptualization, whereas de dicto denies this.
2. Statements about a graph, like ex:saidBy, imply how a graph should be treated.
3. A de dicto/de re transformation which would be required for integrating information from different authors would require using trust mechanisms in order to evaluate whom one trusts to use the right interpretation of a URI. This gets fussy and might be over-engineered for many applications.
4. There is also a suitable workaround for expressing quotation by writing a URI as a typed literal. e.g. "ex:thisURI"^^xsd:string or maybe "ex:thisURI"^^xsd:anyURI

5 Related Work

[Beckett03] Dave Beckett: Redland Notes - Contexts. http://www.redland.opensource.ac.uk/notes/contexts.html

[Beckett04] Dave Beckett: New Syntaxes for RDF. http://www.ilrt.bris.ac.uk/discovery/2003/11/new-syntaxes-rdf/paper.pdf

[Bizer04] Chris Bizer: Semantic Web Trust and Security Resource Guide. http://www.wiwiss.fu-berlin.de/suhl/bizer/SWTSGuide/index.htm

[Carroll03] Jeremy Carroll: Signing RDF graphs. ISWC 2003. http://www.hpl.hp.com/techreports/2003/HPL-2003-142.html

[CarrollStrickler04] Jeremy J. Carroll, Patrick Stickler: RDF Triples in XML. http://www-uk.hpl.hp.com/people/jjc/tmp/trix.pdf

[Dumbill03] Edd Dumbill: Tracking provenance of RDF data. http://www-106.ibm.com/developerworks/xml/library/x-rdfprov.html

[Guha95] R.Guha: Contexts: A Formalization and Some Applications. - http://www.guha.com/guha-thesis.ps

[Hayes04] Pat Hayes: Re:Clarification of Reification vs Quotation. http://lists.w3.org/Archives/Public/www-rdf-comments/2004JanMar/0050.html

[Intellidimension03] Intellidimension: RDF Gateway - Context Based Security. http://www.intellidimension.com/default.rsp?topic=/pages/rdfgateway/dev-guide/security/context.rsp

[Kim02] Anya Kim, Lance J. Hoffman, C. Dianne Martin: Building Privacy into the Semantic Web: An Ontology Needed Now. http://semanticweb2002.aifb.uni-karlsruhe.de/proceedings/Position/kim2.pdf

[Klyne00] Graham Klyne: Contexts for RDF Information Modelling. http://www.ninebynine.org/RDFNotes/RDFContexts.html

[Klyne02] Graham Klyne: Circumstance, provenance and partial knowledge. http://www.ninebynine.org/RDFNotes/UsingContextsWithRDF.html

[MacGregorKo03] Robert MacGregor, In-Young Ko: Representing Contextualized Data using Semantic Web Tools. http://km.aifb.uni-karlsruhe.de/ws/psss03/proceedings/macgregor-et-al.pdf

[McBride02] Brian McBride, Rigo Wenning, Lorrie Cranor: An RDF Schema for P3P. W3C Note 25 January 2002. http://www.w3.org/TR/p3p-rdfschema/

[Miller00] Libby Miller: Statements/Statings. http://www.ilrt.bris.ac.uk/discovery/2000/11/statements/

[Prudhommeaux02] Eric Prud'hommeaux: Source attribution in RDF. http://www.w3.org/2001/12/attributions/

6 Glossary

- **'de dicto' and 'de re'**: Following [Hayes04], the contrast between de dicto (of the speech) and de re (of the thing) can be illustrated by the distinction between direct quotation of speech, as in "Louis said, 'Superman is Clark Kent' " vs. "Louis said that Superman is Clark Kent" . The first, de dicto, reports Louis' actual words (and is false, in the story) while the second, de re, reports what she said about someone, using the speaker's words (and if the speaker knows more about Superman than Louis does, might well be true: even though Louis herself wouldn't identify the guy using the term "Superman", she might well have said that Clark Kent was Clark Kent, and of course as we know, Clark Kent *is* Superman.) … A way to summarize all this is that RDF makes the blanket assumption that all URIrefs are talking about one single 'reality' and so they always refer in the same way.

- **RDF Triple**: The syntactic form of a triple. De dicto, without shared conceptualization.

- **RDF Statement**: Assertion. De re, with shared conceptualization.

- **RDF locally asserted Statement:** Assertion within a context, unasserted outside the context. De re, with shared conceptualization, e.g. Statement within a N3 formula.

- **RDF Stating**: De re in a distributed environment. A stating is the result of somebody claiming a Statement. The truth value of Statings is uncertain and might even depend on the subjective points of view of the information consumer. See also [Miller00].

- **Context:** There is no general agreement of the concept behind the term context. We use the term in the following sense: The context of a statement is the graph in which it is contained.