Using Context- and Content-Based Trust Policies on the Semantic Web

The current discussion about a future Semantic Web trust architecture is focused on reputational trust mechanisms using explicit trust ratings. What is often overlooked is the fact that, besides of ratings, huge parts of the application-specific data published on the Semantic Web are also trust relevant and therefore can be used for flexible, fine-grained trust evaluations. We propose the usage of context- and content-based trust mechanisms and outline a trust architecture which allows the formulation of subjective, task-specific trust policies as a combination of reputation-, context-, and content-based trust mechanisms.

Trust Mechanisms

Reputation-Based Trust Mechanisms include rating systems like the one used by eBay and Web-Of-Trust mechanisms. The general problem with these approaches is that they require explicit and topic-specific trust ratings and that, in many situations, providing such ratings and keeping them up-to-date puts an unrealistically heavy burden on information consumers.

Context-Based Trust Mechanisms use meta-information about the circumstances in which information has been claimed, e.g. who said what, when, and why. They include role-based trust mechanisms, using the author's role or his membership in a specific group, for trust decisions. Example policies from this category are "Prefer product descriptions published by the manufacturer over descriptions published by a vendor" or "Distrust everything a vendor says about his competitor."

Content-Based Trust Mechanisms: These approaches do not use metadata about information, but rules and axioms together with the information content itself, and related information about the same topic published by other authors. Example policies following this approach are "Believe information which has been stated by at least 5 independent sources" or "Distrust product prices that are more than 50% below the average price."

Trust Architecture

For storing aggregated information we use Named Graphs, an extension to RDF which avoids the usage of reification while attaching provenance information to graphs. For querying aggregated information we use TriQL.P a query language which allows the expression of trust policies within queries and returns justification trees together with the query results.

Set of Named Graphs Serialized with TriG

```
g:ex rdf:type ex:Graph .
g:ex rdfs:label "Example Document" .

101 | ex:Monica ex:name "Monica Murphy" .
ex:Monica ex:email <mailto:monica@monicamurphy.org> .
ex:Monica ex:skill ex:Programming .
ex:Monica ex:skill ex:Management .
ex:Monica ex:affiliation ex:projectKnowledgeNet .

102 | ex:Chris ex:name "Chris Brown" .
ex:Chris ex:email <mailto:chris@bizer.de> .
ex:Chris ex:skill ex:Management .
ex:Chris ex:skill ex:Programming .
ex:Chris ex:affiliation ex:projectInterVal .

103 | (ex:Graph1 swp:assertedBy _:w1) .
  _:w1 ex:authenticate ex:Chris .
  _:w1 dc:date "2003-09-03"^^xsd:date .
ex:Graph1 :G1 { ex:Monica ex:skill ex:Programming .
ex:Graph1 :G1 { ex:Monica ex:skill ex:Management .

104 | (ex:Graph2 swp:assertedBy _:w2) .
  _:w2 dc:date "2003-09-04"^^xsd:date .
ex:Graph2 :G2 { ex:Chris ex:skill ex:Programming .
ex:Graph2 :G2 { ex:Chris ex:skill ex:Management .

105 | (ex:Graph3 swp:assertedBy _:w3) .
  _:w3 ex:affiliation ex:projectInterVal .
ex:Graph3 :G3 { :G1 swp:assertedBy _:w1 .
ex:Graph3 :G3 { :G2 swp:assertedBy _:w2 .

106 | ex:Monica ex:email <mailto:monica@monicamurphy.org> .
ex:Monica ex:skill ex:Programming .
ex:Monica ex:skill ex:Management .
ex:Monica ex:affiliation ex:projectKnowledgeNet .
```

TriQL.P Example Query

The following TriQL.P query retrieves all individuals having the skill "Programming", based only on claims by people who have an affiliation to at least 2 projects involving "Programming".

```
SELECT ?person WHERE { 
  ?graph a km:Project .
  ?person km:affiliation ?project .
  ?project km:topic "Programming".
  ?warrant km:assertedBy ?author .
  \(?warrant a km:Warrant .
  \(?project a km:Project .
  \(?author a km:Person .

  FILTER (?project = ex:projectKnowledgeNet .
  ||?project = ex:projectInterVal 
  )

  AND COUNT(?project) > 2

  FILTER(?warrant a km:Warrant)
  FILTER(?project km:topic "Programming")
}
```

The query might return Monica because Chris claims that Monica has the skill "Programming" and Chris works for project InterVal and project KnowledgeNet, both having the topic "Programming". In this case the following justification tree would be attached to the binding "?person = ex:Monica".

Justification Tree

```
```

More information about our trust architecture, TriQL.P example queries and justification trees are found at: http://www.wiwiss.fu-berlin.de/suhl/bizer/TriQLP