Enabling Reputation Interoperability through Semantic Technologies

Rehab Alnemr
Hasso Plattner Institute
Potsdam University
Potsdam, Germany
rehab.alnemr@hpi.uni-potsdam.de

Adrian Paschke
Corporate Semantic Web dep.
Free University
Berlin, Germany
paschke@mi.fu-berlin.de

Christoph Meinel
Hasso Plattner Institute
Potsdam University
Potsdam, Germany
meinel@hpi.uni-potsdam.de

ABSTRACT

Reputation is a complex concept that has a major role in fields like social sciences, economics as well as computer science. Representing it as a simple form of *property-rating* or a vector of ratings strips it from its original notion and postulation. It does not also facilitate the derivation of meaningful conclusions from it. This paper presents a semantic model for the representation of reputation as a complex object - a so called *Reputation Object (RO)*. We also argue that using ontologies, and other semantic web technologies, to represent the *reputation object* model is essential to achieve reputation interoperability and portability between different domains. The contribution is a semantic design artifact for reputation representation that agrees with the theoretical and social formation and processing of reputation information. We also show some applications and how this ontology-based reputation model can be applied in a rule-based open reputation system. The work presented here has significant implications for e-market studies of knowledge sharing.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: [Semantic]; D.2.10 [**Design**]: [Representation]; D.2.12 [**Interoperability**]:

General Terms

Theory, Software

Keywords

Reputation, Reputation Object, Ontologies, Rule Responder

1. INTRODUCTION

In an open world of services and collaborations, trust management approaches are used to construct trust relationships between unrelated parties. These relationships are needed to initiate transactions between these parties. One of the most promising approaches to trust management is the reputation-based approach. Reputation is a social and psychological mechanism that enables unrelated parties to build trust relationships. For centuries reputation has been a social construct that existed in all human societies. Partner selection, social control and coalition formation are some of the main functions of reputation [1]. It acts as a way to reduce the complexity of our social life.

Most of the reputation mechanisms have been developed for a closed domain, a specific project, or by a private company using proprietary schemas. Each has its own method to query, store, aggregate, infer, interpret and represent reputation information. In open environments, trust depends on reputation and how reputation is represented. In most reputation systems, the context of a reputation value is not embedded within the given reputation information. Reputation changes with time and is used within a context. Every domain has its own information sources as well as its own requirements. Therefore, the representation -not the calculation- of reputation should be unified between communities in order to facilitate knowledge exchange. In this paper, we continue our work on a data model for exchanging reputation information between different domains. Enabling reputation portability and linking it to its context eases the management of reputation data, mitigate risks in open environments, and enhance the decision making process.

We propose an upper ontology for representing an entity's reputation. Entity's can be a person, software agent, a business entity, organization, a digital identity, an identity provider, a web service, a device, a product, a statement(s), an event(s), etc. We define reputation as the notion of profiling an entity's performance (-or expected performance) in different contexts. Reputation is represented using a *Reputation Object(RO)*. Reputation information sources are not limited to one context or one domain. The object can accommodate each context in which a reputation value is earned and on the domain of its creation (where a reputation is defined).

Developing such interoperable reputation objects requires a technology that can provide means of *integrating data sources* and methods to *relate the data to its semantics*. Semantic Web technologies were developed with the goal to provide common data representation framework in order to facilitate the integration of multiple sources to draw new conclusions, increasing the utility of information by connecting it to its definitions and to its context, more effi-

cient information access and analysis [7]. Thus, we use some of these technologies to develop the proposed reputation model, based on our previous analysis of the model requirements.[3][5]

This paper starts with a brief introduction to the existing approaches in reputation systems and discusses the issues raised from using a single reputation value. In Section 3 we describe our model, present its formalism and definitions. This is followed by describing how the model is developed, the involved semantic technologies, and the proposed ontology in section 4. Sections 5 and 6 show two levels of the model applications: how it can be used with known ontologies and vocabularies like `foaf` and `GoodRelations` and how it can be applied to rule-based systems. We finish with some related work in section 7 and conclusion and future work in section 8.

2. REPUTATION SYSTEMS

In this section we describe reputation systems in general and then show why existing reputation representations are not enough.

2.1 Approaches to Reputation

In users' web communities, reputation is typically computed from local experiences together with the feedback given by other entities in the community (e.g. users who have used services of that provider). The feedback is an expressed opinion of an entity about another entity that forms recommendation. There are different approaches that consider trust levels between agents as a form of reputation. Trustworthiness levels in these communities are assigned to some or all interacting agents.

Online recommendation systems are systems that use blog roll, tagging, voting/ranking, ratings, and bookmarks to create reputation value such as *Epinions.com*, *eBay*, *Amazon*, *BizRate.com*, etc. Rahman in [27] defines reputation as an expectation about an agent behavior based on information about or observations of its past behavior. There are different reputation models available that support their corresponding agents in building, managing or using trust information. Systems that use reputation values (regardless the difference in calculation, use, and meaning) are calculating reputation by employing different reputation models [31][18][20][32][30]. Each model is developed using different approaches, has different semantics for the deployed reputation concepts and reputation inference such as: subjective approach (scores given by community participants based on personal experience), objective (based on well defined metrics and repeatable criteria) and hybrid between both.

Studying reputation mechanisms require monitoring its dynamics in large communities. Therefore, several simulations are implemented to evaluate different aspects of reputation specially in e-Markets [6] [15] [29]. The simulation developed in the eRep project [1] evaluates different reputation systems in different environments. There are different threats to these reputation systems such as ones directed to the underlying network infrastructure, ballot stuffing, bad mouthing, recommender's dishonesty and so on. This topic is out of this paper scope.

2.2 Issues: Why rating is not enough?

2.2.1 Problem: Excluding the context from the reputation value

The usual representation of a reputation value in current systems is a single value that indicates the entity's expected behavior. The represented value has no embedded information about the context in which this reputation was earned although it is crucial for deriving meaningful trust. As a result, this reputation is a general one which is not connected to a particular context and can not be used outside the domain of its creation. A reputation context or criterion describes the relevant categories for a specific reputation. It is a subjective perspective; for it can represent an area of expertise (i.e. a physician is consulted for medical treatment not for financials), different tags in a social network, quality attributes in communication networks (e.g. QoS, service availability, response time, etc.), or levels of actions (i.e. a person can trust his financial analyst to handle a one thousand Euros deal but not a million Euros deal).[4]

Since *context* is not usually included in a reputation query, it is assumed that the implicit context is the domain of the reputation system (e.g. rate the seller for this purchase transaction) resulting in a too general query. In online markets, for instance, a consumer rates a seller generally for a trading/purchase transaction, leaving the details to be written in a natural language review. The rating should be rather a specific one; rating delivery, quality, price, customer service, etc.. This general score can represent a false criteria if, for instance, a consumer is giving a seller a bad score because of the late *delivery* not because of the *quality of the product*. In real life, reputation is not just a simple value but rather a complex one that relates to a *collection* of contexts in which it was earned.

This kind of generalization leads to lawsuits where sellers object on their reputation scores that resulted from an ambiguity in rating. A study about the legal challenges that face online reputation systems was presented in [11]. We showed in [5] how the use of our proposed reputation object -as opposed to using single values- can reduce the possibility of such legal hassle. It can also make the maintenance of the rating process and its history easier.

2.2.2 Problem: Difference in perceptions and difficulty in mapping

The set of evaluators (i.e. entities that rate the reputation target) have a wide variation in perceptions. It is important to take into account (in both the calculation and the presentation of a reputation value) the perspective of the evaluator. For example, if the rating target is a *security guard* in a private company, to the company's employees he is a *"very trustworthy person"*, but to an officer in the army, he is *"medium trustworthy"*. The mapping between these two perceptions requires the standardization of reputation reference models as explained in [4].

2.2.3 Problem: Incorrect modeling and variance in calculations and interpretations

In that sense, reputation was modeled in a simple way. Although some of these models are based on complicated mathematical calculations, they still do not reflect the real cognitive nature of reputation because they do not represent all the parameters that affect it. Each community use differ-

ent calculation approaches (e.g. sum, average, deterministic, bayesian, fuzzy systems, etc.), different ways of data entry or enquiry that result in different representations, interaction styles and trust rating scales. In [3] we analyzed the rating systems of some known communities and showed how they have different rating scales, visual representations, and entities to be evaluated.

2.2.4 Problem: Non-portable reputation

As a result of the aforementioned problems, transferring reputation from one domain to the other is not possible even if the services offered by these domains have some similarities. For example, a consumer can not transfer his reputation from *eBay* to *Amazon*. He has to register in both communities with two different accounts, thus building totally different reputation values in each community and starting from scratch in each one creating reputation isolated silos. Some of the scenarios where transferring reputation is beneficial are: when an agent in one platform requests an interaction with another agent in different platform, or when an agent registers in a new platform and does not want to start with zero reputation value.[4]

3. REPUTATION OBJECT MODEL

The issues raised in the last section led us to the development of the model described in this section. In this section we give a general description of the model and its benefits abstracted/detached from the semantic technologies used. In our previous work [4][5][3] we analyzed some of the aforementioned problems by focusing on reputation misrepresentation and presented a data model where the representation of an entity's reputation is replaced by a *reputation object* instead of a single value.

In this paper, we present the formal model as well as its development details. According to our definition of reputation (see Section 3), we construct this object to hold a profile of the behavior or performance of an entity in several contexts. For example, in the e-market domain a seller's reputation object reflects his expected performance and rating in several criteria such as *product-quality, payment-methods, delivery*. Each criterion in this list has a numerical value, a string, or a reference to an object value describing the evaluation of this particular criterion. Moreover, a set of criteria (e.g. price, payment method) can be aggregated to be represented by one context (i.e. financial). Aggregating a set of criteria to a single context can be done to enhance the usability of the reputation object (i.e. if it is visible to users or agents) and also to ease the ontology matching process when comparing between two reputation objects (i.e. to relate a criterion meaning like a "payment method" to its general domain or topic which is "financial").

The reputation object, however, is more than a flat list. The model structure (Figure 1) contains a description of how this value is collected (e.g. by community ratings or monitoring service), the computation function (for this criterion) used to aggregate the values each time a new one is entered, and a history list (previous values dated back to a certain time slot). This enables the destination system to map its perception (or its reputation computation function) to the one used in computing this value (i.e. a "very-good" value in system A can be "good" value in system B). The reputation object in this case is seen as a profile of the entity's expected performance which is constructed using different information sources. The degree of visibility for these criteria to the community's users (i.e. how many criteria presented for users in a web site) depends on the community (i.e. a web site can limit the number of criteria for *usability* reasons). The model therefore achieves several goals:

- the reputation of an entity is more meaningful because it is associated with the context in which it was earned
- automation of criteria assignment is possible by declaring a relevant resource as a criterion
  (ex. `URI1 is_a _:criterion`)
- one can easily extend these criteria dynamically by adding to the list of contexts/criteria in the reputation objects

The goal of our work is to have a standard way to represent the reputation of one entity to be understood by any other entity in a different system or domain. Therefore, the goal is to embed more information within the reputation statements with an explanation (or the semantics) of how to interpret it. Our model is generic enough to be used in any domain, but also can be domain-specific by incorporating information (i.e. its contexts, criteria list, and quality processes) that is specific to this domain. In a service oriented environment (SOA), a service registry can use combined sources for service's quality assessment (which leads to building the reputation object) such as service description, invocation analysis, history, rating, meta-data, and elements in Service Level Agreements (SLAs).

One of the benefits of using such model is that regardless the domain that using it, there will be enough information for better decision making. In SOA, having a profile about a service performance (i.e. its reputation object) facilitates customized service selection. The same applies for domains like e-markets (selecting a seller based on a consumer's preferences), in cloud environment (selecting a cloud provider based on the company's customized priorities), in agent-based communities (constructing trust relationships by gossiping about agents' reputation), and in SLA-breach management (identifying violation-prone services at service selection phase[19]). Not to mention the future vision of being able to exchange reputation information between related communities like eBay and Amazon, credit cards databases and C2C money transfer systems, social networks, etc.

3.1 Model Formalism and Definitions

In this subsection, we begin by introducing our definition of reputation and then we describe the elements that compose the set of reputation objects.

DEFINITION 1. *Reputation describes a profile of an entity's performance in one or many contexts based on multiple information sources*

Our model describes a more complex, yet easy to comprehend, reputation representation. The reputation of an entity is represented by a *Reputation Object*. It contains structured information on an entity (the reputation object's owner) to evaluate the expectation of its performance in one or many contexts.

DEFINITION 2. *ROs is the tuple representing all reputation objects and is defined by:*

$$ROs = (A, C, R, range, rep, order)$$

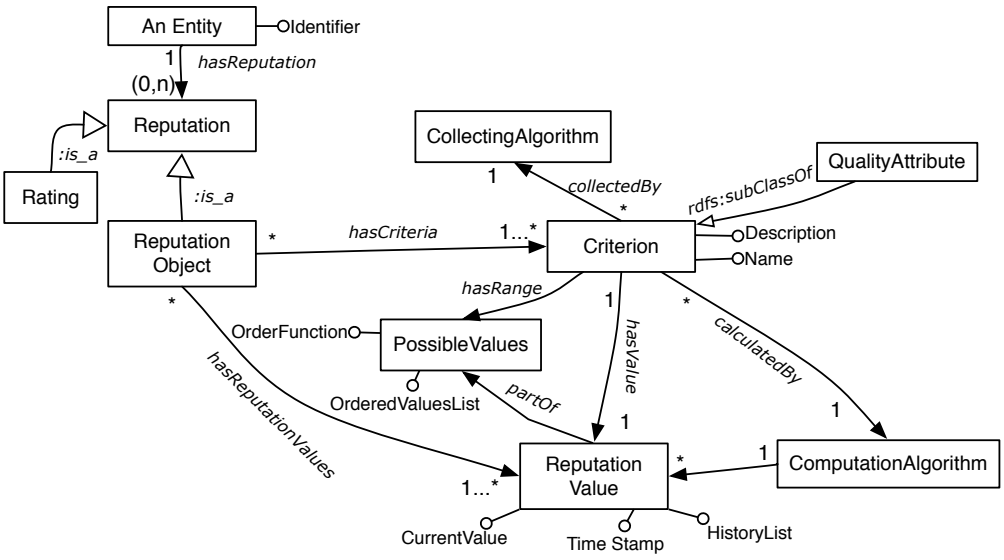

Figure 1: Reputation Object Model

where,

- A is the set of all the entities that can have a reputation or can be evaluated. Entities can be software agents, services, organizations, groups, human agents.

- C is the set of all trust criteria in a reputation profile, each criterion is connected to a set of its possible values. Some relevant criteria can be logically aggregated by an aggregation function to one context. A *context* is the relevant category in which a specific reputation is earned. It can be an objective (measurable) or a subjective one (non-measured but can have an approximated evaluation).

- *range* maps the criteria to its possible values, such that: Let V be a set of values, and $P(V)$ be the set of all possible values that a criterion (or a context) $c \in C$ can have; then

$$range : C \rightarrow P(V)$$

- R is the set that contains the relative pairs of $(entity, criteria)$ only, such that:

$$R \subseteq A \times C$$

  A criterion x and a set of values $V \subset \neg R$ if x can not be logically evaluated by V.

- For a new transaction that rates a criterion $c \in C$, its final computed value v is computed by a relevant computation function *rep*. *rep* maps a criteria to its value, where $rep : R \rightarrow V$ such that:

$$rep(a, c) \in range(c), a \in A$$

- *order* maps the set of possible values $P(V)$ to its relevant order and is used in the comparison between two values in the $P(V)$ set, where

$$order : C \rightarrow P(V^2)$$

such that: $order(c) \preceq$ is a partial order of $range(c)$ to differentiate between the relative best from the relative worst value.

4. MODEL DEVELOPMENT USING SEMANTIC WEBTECHNOLOGIES

In the previous section, we described our model and its goals. Achieving these goals requires a technology that provides common data representation framework as well as a way to connect concepts with their definitions. Semantic Web is developed with a main objective of facilitating data integration, enhancing information usage by connecting it to its definitions and context.[7] RDF is used as a mechanism for data integration across applications and the web.[2] Therefore, it was only to be expected that Semantic Web is the technology of choice to achieve reputation portability and interoperability. Developing our model using semantic web technologies achieves:

- seamless interaction between agents of different domains
- the goal of exchanging reputation information (and knowledge) and its meaning
- reputation interoperability
- the development of context-aware reputation
- customized service-provider selection
- understandability and reusability of the embedded reputation information

4.1 A Simple View: Reputation Objects in RDF Graphs

A reputation statement usually describes the target of the statement, the topic of evaluation, and the value of this evaluation (i.e a judgment or a result of monitoring process). When talking about someone's reputation most of the time one would describe it in a set of such statements. These type

Table 1: Reputation Statements about Bob

Target	Criterion	Value
Bob	Service Quality	0.87
Bob	Delivery	"very good"
Bob	Payment	<purl.org/goodrelations/v1/MasterCard>

of statements correspond to the RDF statements (or triple) form of: `<subject,predicate,object>`, where the reputation statement in this case is: `<target,context,value>`. The same as an RDF graph which is a set of RDF triples, the set of reputation statements therefore form a reputation RDF graph. Lets assume that we are rating a seller in an e-market identified by `<foaf:Person rdf:nodeID="Bob">` then a simple description of his reputation can be viewed as declaring the statements in table 1. If Bob's servie-quality, delivery, and payment are identified by URIs as well as the literal values 0.87 and "very good", this table corresponds to the RDF graph instance shown in figure 2 where the edges represent the `context`. This is a snippet of the reputation object instance that describes Bob's reputation in different criteria:

RO={<Bob,quality,0.87>, <Bob,delivery,''very good'' >, <Bob, payment,gr:MasterCard>}

For the same person "Bob" that is identified by a given URI, more statements can be asserted about him and easily *merged* to the graph representing his reputation if the `predicate` is a new `criterion`. If a new statement has an equivelent criterion, then the reputation value (`object`) is aggregated (or recomputed, according to the computation function) to produce a new current value for this criterion.

4.2 Reputation Expressiveness via Reputation Object Ontology

The next step is to formalize and develop the model components and concepts using a proper technology. RDFS (RDF Schema) can be used to describe the model classes (ex. *Reputation, ReputationObject, Context, Criterion,* etc.) and properties (ex. *reputationValue, hasCriterion,* etc.). However, after developing the schema and testing it using some use cases, we found that the model needs to be described using more expressive method. Restrictions and axioms of the model should be incorporated in its description such as: how is the reputation value obtained, can a criterion refer to another concept (criterion matching) in other platforms,

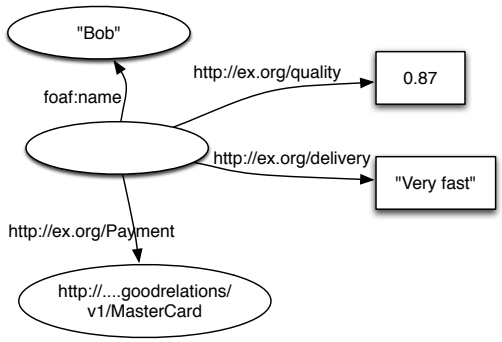

Figure 2: A graph describing part of Bob's reputation

how to aggregate values of this concept if a new evaluation value is entered, can a set of criterion be aggregated in one context, how many reputation objects can an entity have, can the reputation object be extended, cardinality, inverse relationships, influencing factors, etc.. In such case, ontologies are used to provide such level of expressiveness.

We have developed an OWL ontology to represent an entity's (`foaf:Agent`) reputation object. Tables 2 and 3 have a description of the classes and their properties.
A `ReputationObject` has:

1. `hasCriteria`: one or multiple instances of class `Criterion` or `QualityAttribute` (for a service, the criterion describing service reputation is referred to as a quality attribute). The criterion is collected using a `CollectingAlgorithm` and `hasValue ReputationValue`.

2. `hasReputationValues`: each criterion instance has a `ReputationValue` (which includes the `currentValue`, its time stamp, and a simple list of its previous values called `historyList`) that in turn has the range of values defined in `PossibleValues`. It describes the data type that the criterion can have or a specific set of values (literals or resources URI) evaluating this criterion (e.g. a set of integers $\{1, 2, 3, 4\}$ describing 4 trust levels or a set of Strings $\{''good'',''bad'',''excellent''\}$ describing a user opinion). Each time a criterion is being evaluated (i.e. a new entry value for this criterion), a new `currentValue` is calculated using the `ComputationAlgorithm` which is the reputation computation function used with this criterion such as *sum, average, etc..*

Since it is not always the case to identify intuitively what the highest reputation value is -among the defined possible value set, for instance-, the `PossibleValues` class has an `orderedList` that is ordered from the relatively highest reputation value to the lowest (e.g.$\{''excellent'',''good'',''bad''\}$). It also has the possibility to define a comparison and ordering function; `OrderFunction`. For example: if the criterion is a student grade- float number from 1 to 4 representing the GPA- the function is "greater than" when it is American GPA (4 is the highest) and the function is "less than" when it is German GPA (1 is the highest). In the presented ontology, we use a pattern *OWL-List* [12] to retain the order of the list.

The ontology makes use of other vocabulary such as `OWL`, `RDFS`, `FOAF`, `XSD`, and can integrate with vocabulary such as `Trust` or `RDF Review` to describe one criterion in a reputation object. Using this ontology, the object representing the reputation can be transferred within a domain or to another domain without negotiating on its format or semantics. The advances in ontology matching techniques ensures the matching between the criteria of a reputation object in one platform to be used in another platform.

5. IMPLEMENTATION

We used Protégé-OWL tools [1] in the development of the ontology. Currently, the ontology is being tested for stability using the default reasoning engines and adjusted accordingly. Implementation for reading, writing, and processing

[1]Protege OWL: `http://protege.stanford.edu/overview/protege-owl.html`

Table 2: Reputation Object Model Ontology Classes

Classes	Description
`Reputation`	An abstract Reputation
`Rating`	A single Rating representing an entity's reputation, refers to other ways of representing reputation, `subclassof:Reputation` and has one owner
`ReputationObject`	A Reputation Object of an entity related to multiple criteria, `subclassof:Reputation` and has one owner
`ReputationValue`	contains the current value of the criterion along with its past values if they exist
`PossibleValues`	A set of possible values that a criterion in a reputation object can have (literals or resources) which can be a static predefined set or a general range
`Context`	A reputation context that represent multiple criteria
`Criterion`	A reputation criterion to be evaluated and saved in a reputation object
`QualityAttribute`	A reputation criterion regarding quality measuers
`Algorithm`	A methodoligical method, entry point, or an engine
`ComputationAlgorithm`	The method used to compute or aggregate several reputation values (i.e. reputation function)
`CollectingAlgorithm`	The engine or method used to collect the value of a reputation criterion

an RO along with the `selection` method (given a consumer priority list) was developed in Java using Jena-API [2] which facilitates the integration of the model in any system on the implementation layer. The mapping from the RDF properties to the domain classes can be done using the implementation described in [26].

For declarative processing of the semantic reputation objects we make use of rules. *Reputation objects* are attached to the rule-based services (agents) in the Rule Responder system[3]. Rule Responder allows to deploy distributed rule inference services running a local rule engine such as Prova or Drools on an enterprise service bus. [25] The rule services, which can act as multi-agents, can communicate with each other using Reaction RuleML[4] as a standard rule interchange format. The reputation values can be used in the agent's rule logic, e.g. to implement access policies, information dissemination rules or decision management strategies. For instance, an agent might reveal more information to a trusted requestor or might internally prioritize incoming requests from other agents according to the details in their reputation objects. That is, an agent in Rule Responder can manage reputation locally in its knowledge base, but can also communicate reputation objects to other agents. For the implementation of the application scenarios in the following section we used the Prova Semantic Web rule engine[5] in Rule Responder, which supports using Semantic Web ontologies as type systems and allows queries to RDF data. [24]

6. APPLICATIONS

In this section we show some usage examples of the *Reputation Object Model* by showing a snippet of a person's RO in a social network, an agent's RO in a rule-based reputation system, and a RO of a seller in an e-commerce environment. The contents of the examples are simplified for better illustration. The abbreviation **ro** is used to denote the model name-space.

6.1 Reputation Object of a foaf:Person

A simple example to profile the reputation of a `foaf:Person` which is asserted by a property `ReputationObject` is to describe his performance in the criteria: `Social-Activity` and `Driving`. These properties can take any URI as a value or can be assigned to literal values (numericals or strings).

Listing 1: Person's RO

```
<foaf:Person rdf:about="#Alice">
  <foaf:openid rdf:resource=http://alice.org''/>
  <ro:hasReputation >
  <ro:ReputationObject rdf:ID=''AliceRO1''>
    <ro:hasCriteria>
    <ro:Criterion rdf:resource=''#Driving''>
      <ro:hasReputationValue ro:value=8/>
      <ro:collectedBy ro:CollectingAlgorithm=''#Rating''/>
    </ro:Criterion>
    <ro:Criterion rdf:resource=''#SocialActivities''>
      <ro:hasReputationValue ro:value=''active''/>
      < ro:collectedBy >
        <ro:CollectingAlgorithm rdf:resource=''www.ex.org/
            NoOfFriends''/>
      </ro:collectedBy>
    </ro:Criterion>
  </ro:ReputationObject>
</ro:hasReputation >
</foaf:Person>
```

6.2 Rule-based Open systems

Entities which are able to perform actions autonomously in a given context are called agents. In a reputation system the involved agents are those who make the evaluations, the targets of the evaluations, and those who benefit from

[2]Jena framework: `http://jena.sourceforge.net/`
[3]Rule-Responder: `http://responder.ruleml.org`
[4]RuleML Initiative: `http://ruleml.org/`
[5]Prova Engine: `http://www.prova.ws`

Table 3: Reputation Object Model Ontology Properties

Properties	domain:	range:
`hasReputation`	foaf:Agent	ReputationObject, Rating
`hasCriteria`	ReputationObject	Criterion or QualityAttribute
`hasReputationValue`	Criterion or QualityAttribute	ReputationValues
`historyList`	ReputationValue, list of the past values for a criterion in a particular time slot	Collection of PossibleValue
`currentValue`	ReputationValue, describes the current value of a criterion	PossibleValues
`hasRange`	Criterion or QualityAttribute	PossibleValues
`orderedValuesList`	PossibleValues, describes the order of the possible values for a criterion to be able to compare between 2 values	OWLList
`orderFunction`	PossibleValues, describes the comparison function (i.e. between two given reputation values) and is used as an alternative to order a dynamic set of possible values if a static list is not given	Algorithm
`calculatedBy`	Criterion or QualityAttribute	ComputationAlgorithm
`collectedBy`	Criterion or QualityAttribute	CollectingAlgorithm
`hasRatingValue`	Rating	type:literal

this evaluation. In RuleResponder rule-based agents can communicate with each other and can exchange reputation objects or specific input-output values of the reputations measurement functions (see listing 2). [25]

This allows for implementation of decentralized reputation models where agents interchange and evaluate their reputation objects, but also for centralized reputation models where special agent/service nodes act as trusted reputation management system. The interchanged reputation objects can be used in the internal rule-based decisioning policies and behavioral policies of a RuleResponder agent. For instance, an agent might give certain rights to a trusted `foaf:Person`.

Listing 2: Communicate Reputation Objects

```
...
sendMsg(Sub−CID,esb,Agent,acl_query−ref, QueryRO),
rcvMsg(Sub−CID,esb,Agent,acl_inform−ref, ReceivedRO),
evaluateReputation(ReceivedRO),
...
```

6.3 Reputation Object of a Seller

In this simplified example, a seller's reputation is described by the evaluation of two criteria: `Review` and `DeliveryMethod`. A seller or a business entity can be described by the vocabulary GoodRelations [6] which is an ontology for describing offerings and other aspects of e-commerce on the Web. The `WebPortal` specifies that the criterion `DeliveryMethod` has the reputation value *standard* if only one delivery method is available or has the value *several* otherwise.

Review is a vocabulary for sharable reviews and simple ratings [7]. The final rating value -defined by the ontology- can only be a numeric value and expresses the reviewer's value judgement on the work.

[6]GoodRelation Vocabulary: `http://purl.org/goodrelations/`

[7]RDF Review Vocabulary: `http://hyperdata.org/xmlns/rev/hReview`

Listing 3: Seller's RO

```
<gr:Reseller rdf:reference="http://www.example.org/John#">
 <ro:hasReputation >
 <ro:ReputationObject rdf:ID=''SellerRO1''>
 <ro:hasCriteria>
  <ro:Criterion rdf:resource=''http://purl.org/goodrelations/v1/
       DeliveryMethod''>
   <ro:hasReputationValue>standard</ro:hasReputationValue>
   <ro:collectedBy ro:CollectingAlgorithm=''#WebPortal''/>
  </ro:Criterion>
  <ro:Criterion>
    <review:Review>
      <review:rating>8</review:rating>
    </review:Review>
  </ro:Criterion>
 </ro:ReputationObject>
 </ro:hasReputation >
</gr:Reseller>
```

The decision rule of a customers' agent to buy a product (e.g. a book) from a certain seller depends on the rating of the reviewer and the delivery method:

Listing 4: E-Commerce Buyer Decision Rule

```
if reseller(Name:gr_Reseller, Book:gr_Book)
   hasDeliveryMethod("standard")
   hasReviewRating("excellent")
then buy(Name:gr_Reseller, Book:gr_Book)
```

7. RELATED WORK

Trust ontology is presented by Golbeck [17] as an extension to `foaf` vocabulary. The ontology is written in OWL/RDF and allows people to say how much they trust other people. The authors define reputation as a rating on a scale from 1 - 10 where 10 signifies absolute trust and 1 signifies no trust. In our model, we can use this ontology as one *criterion* in a *reputation object* to describe the overall opinion provided by other agents. Also, it can be used as a simple way of describing one agent's opinion about another in several contexts wrapped in one reputation object.

Reputation ontologies presented in [10][22] present conceptual models that focus on how this reputation obtained

and which category of reputation it is. The work in SOARI [14] focus on the mapping between two existing reputation ontologies using a common service architecture like FORe [9].

Maximilien and Singh [23] present a model of web service reputation that addresses the problem of lacking dynamic service discovery and that there is no memory of service bindings and interactions. Some limitations of their work is that they define *service quality reputation* as the aggregation of collected service quality data for a given quality, which limits the presentation of reputation to single values. The authors in [8] use client side monitoring techniques to generate information regarding functional and non-functional (QoS parameters) properties of service behavior. They also use three components to manage service reputation: Reputation Manager RM(collect feedback from the clients), Subscription Manager (disseminate information provided by RM or by the Service Directory), Extended Service Directory (manage registered services).

There are several projects that are concerned with *how trust and reputation is calculated* rather than *how they are represented*. They focus on the computation algorithm used to infer trust values in different domains (presented in our model by different `ComputationAlgorithm`) such as authors [21] present EigenTrust algorithm focusing on peer-to-peer systems based on the Page Rank algorithm used by Google, Gil and Ratnakar [16] focus on trusting content and information sources on the Semantic Web based on individual feedback about the sources, authors in [28] use social networks with trust to calculate the belief a user may have in a statement. ENSIA [13] present a set of recommendations for developing secure reputation systems, one of them is to unify the representation of reputation data. However, the document focuses on what to do not how to do it.

8. CONCLUSION

Context-aware reputation can be used as a tool to solve several problems in several domains. It is important to reach with collaborative communities to the state that reflects the behavior of their counterpart human communities, hence making them more intuitive. Most current reputation systems and tools do not consider interaction between different domains, platforms, and services. They often produce general reputation values that can not be fit to avoid legal issues. Furthermore, most of the research focus on methods of calculating reputation values.

In this paper, we presented our formal reputation object model to address the issues raised in these systems, basically by using a new representation for reputation. We argued that to capture the real concept of reputation and to enable reputation portability and interoperability, semantic web technologies should be used. The paper presents the developed ontologies and shows how it can be used with other vocabulary. We empower the decision making process by attaching reputation objects to the rule-based services (agents) in the Rule Responder system. The ontology is begin tested for stability and is also used in several use cases for evaluation.

9. REFERENCES

[1] eRep project: Social knowledge for e-governace. `http://megatron.iiia.csic.es/eRep/?q=node/93`.

[2] Rdf vocabulary description language. `http://www.w3.org/TR/rdf-schema/`.

[3] R. Alnemr, J. Bross, and C. Meinel. Constructing a context-aware service-oriented reputation model using attention allocation points. *Proceedings of the IEEE International Conference on Service Computing (SCC 2009)*, pages 451– 457, 2009.

[4] R. Alnemr and C. Meinel. Getting more from reputation systems: A context-aware reputation framework based on trust centers and agent lists. *Computing in the Global Information Technology, International Multi-Conference*, 2008.

[5] R. Alnemr, M. Quasthoff, and C. Meinel. *Taking Trust Management to the Next Level*. Handbook of Research on P2P and Grid Systems for Service-Oriented Computing: Models, Methodologies and Applications, pp. 796-816, 2009.

[6] P. Avegliano and J. S. Sichman. Using the RePart simulator to analyze different reputation-based partnership formation strategies within a marketplace scenario. *Trust in Agent Societies*, pages 226–243, 2008.

[7] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American Magazine*, May 17, 2001.

[8] D. Bianculli, W. Binder, L. Drago, and C. Ghezzi. Transparent Reputation Management for Composite Web Services. In *ICWS '08: Proceedings of the 2008 IEEE International Conference on Web Services*, pages 621–628, Washington, DC, USA, 2008. IEEE Computer Society.

[9] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web services architecture. *W3C recommendation, W3C*, 2004.

[10] Casare and Sichman. Towards a functional ontology of reputation. *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 2005.

[11] J. Chandler, K. El-Khatib, M. Benyoucef, G. Bochmann, and C. Adams. *Legal Challenges of Online Reputation Systems*. Chapter in Trust in E-Services, pp.84-111, 2007.

[12] N. Drummond, A. Rector, R. Stevens, G. Moulton, M. Horridge, H. Wang, and J. Sedenberg. Putting owl in order: Patterns for sequences in owl. In *OWL Experiences and Directions (OWLEd 2006)*, Athens Georgia, 2006.

[13] ENISA. ENISA position paper no. 2 reputation-based systems: a security analysis. October 2007.

[14] N. et al. SOARI: A service oriented architecture to support agent reputation models interoperability. *Trust in Agent Societies: 11th International Workshop, TRUST 2008, Estoril, Portugal*, 2008.

[15] T. Eymann, S. König, and R. Matros. A framework for trust and reputation in grid environments. *Journal of Grid Computing*, 6:225–237, 2008.

[16] Y. Gil and V. Ratnakar. Trusting information sources one citizen at a time. *Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy*, 2002.

[17] J. Golbeck, B. Parsia, and J. Hendler. Trust networks on the semantic web. *Proceedings of Cooperative*

Intelligent Agents,Helsinki, Finland, 2003.

[18] T. Grandison. Trust management tools, 2007.

[19] I. U. Haq, R. Alnemr, A. Paschke, E. Schikuta, H. Boley, and C. Meinel. Distributed trust management for validating SLA choreographies. *Proc. Workshop SLAs in Grids (in conjunction with Grid'09), CoreGRID Springer series, Banff, Canada*, October 2009.

[20] B. K., C. Chandrasekhar, and R. Viana. Portable reputations with egosphere, 2004.

[21] S. Kamvar, D. Mario, T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in P2P networks. *Proceedings of the 12th International World Wide Web Conference*, 2003.

[22] Malik and Bouguettaya. Rateweb: Reputation assessment for trust establishment among web services. *The VLDB Journal*, pages 885–911, 2009.

[23] E. Maximilien and M. Singh. Toward autonomic web services trust and selection. *Proceedings of the 2nd international conference on Service oriented computing, New York, NY, USA*, 2004.

[24] A. Paschke. A typed hybrid description logic programming language with polymorphic order-sorted DL-Typed unification for semantic web type systems. In *OWLED*, 2006.

[25] A. Paschke, H. Boley, A. Kozlenkov, and B. L. Craig. Rule responder: RuleML-based agents for distributed collaboration on the pragmatic web. In *ICPW*, pages 17–28, 2007.

[26] M. Quasthoff, H. Sack, and C. Meinel. Can software developers use linked data vocabulary? *Proceeding of I-Semantics'09, Graz*, 2009.

[27] A. Rahman and Hailes. Supporting trust in virtual communities. *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, Hawaii*, 6:6007, 2000.

[28] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web,. *Proceedings of the Second International Semantic Web Conference. Sanibel Island, Florida*, 2003.

[29] J. Sabater, M. Paolucci, , and R. Conte. Repage: Reputation and image among limited autonomous partners. *Journal of Artificial Societies and Social Simulation*, 9(2), 2006.

[30] J. Sabater and C. Sierra. Review on computational trust and reputation models. *Artificial Intellegence Rev.*, 24:33–60, 2005.

[31] W. Sherchan, S. W. Loke, and S. Krishnaswamy. A fuzzy model for reasoning about reputation in web services. *Proceedings of the 2006 ACM symposium on Applied computing*, 2006.

[32] G. Zacharia. Trust management through reputation mechanisms. *Applied AI*, 14:881–907, 2000.