Ontological Analysis of Human Relations for Semantically Consistent Transformations of FOAF Data

Miroslav Vacura and Vojtěch Svátek

Faculty of Informatics and Statistics,
University of Economics
W. Churchill Sq.4, 130 67 Prague 3,
Czech Republic
vacuram|svatek@vse.cz

Abstract. The FOAF project has prominent importance for capturing human relations in Linked Data. We analyze the FOAF data structures and their extensions from the point of view of formal ontology and discuss problems inherent in its design. We also point out necessary considerations for transforming the FOAF data structures by supplying additional knowledge into them, while achieving/maintaining semantic consistency.

1 Introduction

The Linked Data initiative was started by Tim Berners-Lee as an architectural vision for the Semantic Web. It explores the idea of Semantic Web as puting emphasis on making links so both people and machines can explore the interconnected web of data. If the data are linked then "when you have some of it, you can find other, related, data" [1]. Just like in HTML where there are relationships and hypertext links between documents, the Linked Data initiative wants to encourage a similar approach in the case of general data content, described by RDF. The key requirements for Linked Data are quite simple:

1. Use URIs as names for things.
2. Use HTTP URIs so people can look up those names.
3. When someone looks up a URI, provide useful information, using standards (RDF*, SPARQL).
4. Include links to other URIs, so that they can discover more things.

Guidance provided by these general points was later extended by technical documents like [3] and [12], as well as conference overview papers like [5] and [4]. Linked Data can be now crawled with an appropriate browser following RDF links; a search engine can also search these information sources similarly to conventional relational databases. However, unlike HTML, which only provides a generic linking capability, links in Linked Data environment can have different

types: we can e.g. specify that one person is author of a paper, or that this person *knows* another.

In our paper we focus on the problem of 'injecting' additional knowledge into Linked Data. We provide a case study based on one of the key projects in Linked Data – FOAF [6]. We analyze this 'standard' from the point of view of ontological engineering, and provide guidelines for injecting knowledge while maintaining semantic consistency.

The rest of the paper is organized as follows: The next section brings the basic characterization of the FOAF project, its history and extensions. It also points out some basic issues and complexities. Section 3 analyses the formal ontological structure of the *relationship vocabulary* extension of FOAF, and Section 4 proceeds with a detailed analysis of properties this vocabulary defines. Section 5 investigates the possibilities of leveraging on the previous analysis for supplying additional structures to FOAF data while maintaining semantic consistency, and presents some transformations patterns that can be utilized for such a purpose. Finally, Sections 6 provides some conclusions acknowledgments.

2 Relation knows in FOAF

In this section we will discuss some problems related to the FOAF project and the 'knows' relation. The FOAF project is well known in the Linked Data community and the 'knows' relation is an intuitive relation well understood by everyone. Since 2004 there were more than 1 million FOAF documents and 79% of them utilized the knows property [7].

The Friend of a Friend (FOAF) project was started with the ambition of creating a Web of machine-readable pages describing people, the links between them and the things they do, work on, create and like.[1]

For us the most important property of FOAF is knows, defined as "a person known by this person (indicating some level of reciprocated interaction between the parties)." It is understood as property of a person, however it is defined clearly as *symmetric* relation, because the specification requires "some form of reciprocated interaction" and stresses that "if someone knows a person, it would be usual for the relation to be reciprocated" [6].

The word "knows" is vague, and the FOAF specification doesn't resolve this vagueness in any formal way. It is described in natural language in the basic FOAF specification, and any explication or formalisation is lacking. For at least partial disambiguation of what this relationship means we have to turn to the *relationship FOAF module* developed in 2002 by E. Vitiello.[2] The RDF schema of this module defines several subproperties of property knows: friendOf, acquaintanceOf, parentOf, siblingOf, childOf, grandchildOf, spouseOf, enemyOf, antagonistOf, and ambivalentOf.

Inclusion of some of these properties seems debatable. For example, if a person describes someone as his/her enemy, then the person surely knows this

[1] http://www.foaf-project.org

[2] http://www.perceive.net/schemas/20021119/relationship/

enemy; however, the opposite may not be true – one may not know his/her enemy. Also inclusion of such subproperty in a *friend* of a *friend* vocabulary seems counterintuitive, because the general intuition may be that knows is in the semantic context of FOAF a positive (or at least neutral) relation between people. Another problem may arise if we in an application formally define the knows relation as symmetric as suggested in the FOAF specification. The property enemyOf is clearly not symetric (it is asymetric). Properties like childOf are obviously antisymmetric, and defining an antisymmetric property as subproperty of a symmetric one is logically inconsistent. This just emphasizes the problem of vagueness of the term knows.

Since 2004 the relationship module has been modified to a more general *relationship vocabulary* and is continually maintained and enhanced.[3] The following subproperties were added: ancestorOf, apprenticeTo, closeFriendOf, collaboratesWith, colleagueOf, descendantOf, employedBy, employerOf, engagedTo, friendOf, grandparentOf, hasMet, influencedBy, knowsByReputation, knowsInPassing, knowsOf, lifePartnerOf, livesWith, lostContactWith, mentorOf, neighborOf, participant, participantIn, Relationship, worksWith, and wouldLikeToKnow. The *relationship vocabulary* is now based on OWL, as some of the properties are explicitly declared with regard to the OWL standard. Still, however, the *relationship vocabulary* has not become too popular. A quick survey using the Swoogle[4] search engine revealed that only less than 0.1% indexed FOAF documents use this extension.

We have not been able to find out whether any particular methodology was used for choosing these subproperties or they were added just ad hoc or based on suggestions by participants of FOAF-DEV mailing list.[5] The main limitation of the *relationship FOAF module* is that it consists of a fixed and very limited set of subproperties. This has been partially overcome by extending it and turning it into a generic vocabulary. The description of extended properties now includes some semantics with a more complex subproperty structure. Still, probably for backward compatibility, properties like childOf are considered to be subproperties of knows. The new property knowsOf, which is not symmetric, was introduced, although only recently (February 2010) its semantics was changed such that the asymmetric knowsOf is no longer subproperty of the symmetric FOAF property knows; now, correctly, knows is subproperty of knowsOf.

It could be also noted that x childOf y does not imply x knows y. Such consideration was not important when we thought of *relationship* as a module or extension of FOAF, but when considered as a generic vocabulary that could be possibly included in any complex knowledge or reasoning system then it is important that it should not lead to logically incorrect conclusions.

Similarly, a recent (February 2010) revision of the *relationship vocabulary* acknowledged that for distant descendants it may not be possible to know reciprocally each other, so the property descendantOf is no longer subproperty of

[3] `http://purl.org/vocab/relationship`
[4] `http://swoogle.umbc.edu`
[5] `http://lists.foaf-project.org/mailman/listinfo/foaf-dev`

knows. However, the editors failed to notice that we run into exactly the same problem when we consider the property grandparentOf and even parentOf. For a grandparent (and even parent) of a person could die before the person was born, so there are real-world cases where people could not know their children or grandchildren. This can also happen in some other special circumstances.

Another problem of the *relationship vocabulary* is the definition of domains and ranges of the described properties. The problem becomes visible in the case of properties like employedBy – both its domain and range are set to class Person. In a real-world scenario an entity that employs other persons is usually a legal entity – company, institute or some other type of organisation. This may be called *legal person*; however, in the FOAF vocabulary such kind of entity is represented by class Organisation, which is by explicit semantic statement disjoint with class Person. If we use this formalization then we cannot express that a (physical) person is employed by an organisation without introducing logical inconsistency into our system. The employment relation is in FOAF usually expressed by the property workplaceHomepage with range Document. Then this document can be related to an Organisation using the property homepage. It would be intuitive to say that if the workplaceHomepage of a Person is a Document that is the homepage of an Organisation that it implies that the Person is employedBy an Organisation. But this is impossible in the scope of *relationship vocabulary* semantics, which defines employedBy as relation between two (physical) persons.

The reason why we go into such depth with the analysis of the FOAF project and the related *relationship* vocabulary is to see what difficulties are there when we are to link these data to some even very little different semantic system. Logical relations between properties and subproperties are important because they are used even in the most simple reasoners and information aggregators like e.g. Tabulator [2].

3 Formal Ontological Structure of *Relationship Vocabulary*

The previous section comprised informal discussion of some problems identified in the FOAF relationship module and in its more recent *relationship vocabulary* extension. This section will only focus on the latter, and will provide a more detailed analysis of its ontological (and logical) structure.

If we are to process knowledge captured in the FOAF format, we have to properly understand its ontological structure. A failure to do so may result in introducing semantic inconsistency to the knowledge thus transferred to an application.

If we go through the list of terms defined in the *relationship vocabulary*, there is one thing that immediately catches one's attention. The majority of terms describes standard properties that have the class Person as both domain and range. An example is the property livesWith – a relation between two persons, in this case symmetric. But in the *relationship vocabulary* there are also three terms that don't fit within this description: participant, participantIn and Relationship.

The term Relationship designates a class rather than a property. The terms participant and participantIn designate two properties, in turn. The domain of property participantIn is class Person, but its range is class Relationship. In the case of property participant it is the other way around. Intuitively we would expect these two properties to be inverse of each other, however, this is not formally declared in the *relationship vocabulary*.

An interesting fact we observe is that *relationship vocabulary* does not include one ontology pattern for human relations but actually *two* of them.

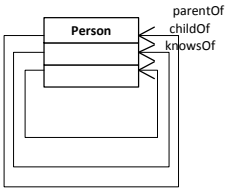

Fig. 1. Ontology pattern of *relationship vocabulary* 1

The first pattern that follows the legacy of the original FOAF is depicted in Figure 1. Human relations are defined as properties that have the class Person as both domain and range. This is formally just an extension of the original property knows – based on an idea that the new properties will be just subproperties of this property, thus maintaining "backward compatibility".

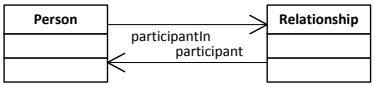

Fig. 2. Ontology pattern of *relationship vocabulary* 2

The second pattern introduces class Relationship that is described as a "class whose members are a particular type of connection existing between people related to or having dealings with each other." Based on this description it would seem that members of this class are reifications of *types* of relations – so we have one member per type of relation (note that we do not mean RDF reification here, but individuals representing types, i.e., indirectly, sets of other individuals). We have one member representing the relation "FriendOf", another representing the relation "livesWith", and so on. Let's take for example the relation "FriendOf" and we will call its reification FriendOf – it will be an instance of class Relationship. Now let's say that Petr and John (members of class Person) are friends:

$$\mathsf{participantIn}(\textsc{Peter}, \textsc{FriendOf})$$
$$\mathsf{participantIn}(\textsc{John}, \textsc{FriendOf})$$

Now let's say that Mary and Jane are also friends. So we can again add two assertions:

$$\mathsf{participantIn}(\textsc{Mary}, \textsc{FriendOf})$$
$$\mathsf{participantIn}(\textsc{Jane}, \textsc{FriendOf})$$

Now we have in our knowledge base four assertions (RDF triplets), but how no information on who is friend of who, see Figure 3. Such a structure only provides information about who is in any friendship relation at all, and seems therefore semantically inadequate.

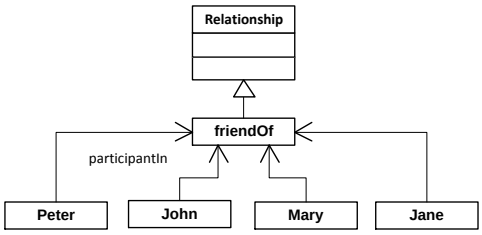

Fig. 3. Instance FriendOf of class Relationship and other instances.

We must conclude that for class Relationship to be of realistic use it must have members that are not reifications of *types* of relations but reifications of actual relations. This subtle ontological difference means that we have an instance of class Relationship for every individual relation. So relation FriendOf between Peter and John would be reified to instance FriendOf1 and relation FriendOf between Mary and Jane would be reified to instance FriendOf2.

$$\mathsf{participantIn}(\textsc{Peter}, \textsc{FriendOf1})$$
$$\mathsf{participantIn}(\textsc{John}, \textsc{FriendOf1})$$
$$\mathsf{participantIn}(\textsc{Mary}, \textsc{FriendOf2})$$
$$\mathsf{participantIn}(\textsc{Jane}, \textsc{FriendOf2})$$

The resulting ontological structure is in Figure 4. We can see that now we can still recognize who is friend of who. FriendOf1 can be easily recognized as reification of individual relationship of Peter and John, and similarly FriendOf2 can be recognized as reification of relationship of Mary and Jane.

Still it is hard to see how we can model asymmetric properties using this ontology pattern. Let's take for example fanOf – how can we describe that Peter is fan of Beethoven but Beethoven is not fan of Peter? We can perhaps use negative property assertions of OWL 2 (or the pattern-based approach for OWL 1 described in [9]), but this would go against the intended simplicity of FOAF, and the definition of *relationship vocabulary* never mentions such need for higher languages.

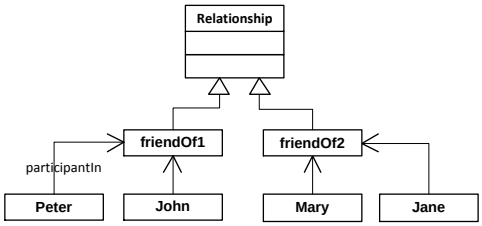

Fig. 4. Instances FRIENDOF1 and FRIENDOF2 of class Relationship and other instances.

It seems that the unclear and confusing definition of these class and properties belong to the reasons why they aren't more generally utilized. Using the Swoogle search engine we were not able to find any document, except various cached versions and copies of the original *relationship vocabulary* RDF, that would use these constructs. We believe that under such circumstances the maintainers of the *relationship vocabulary* should either review and rework these constructs and the relevant documentation or drop them completely.

4 Analysis of properties

The core of some of the problems we identified can be found in confusing the epistemic and ontological state of affairs.

- **Epistemic** state of affairs concerns with what is *known* to conscious agents. We may ask e.g. "Does x know that y is his/her child (enemy, neighbor, ancestor etc.)?" In none of these cases the answer is obvious and it may require further empiric investigation, which in this case consists in *questioning* of person x.
- **Ontological** state of affairs deals with what is matter of fact independently of knowledge (epistemic state) of particular conscious agents. We therefore ask: "Is matter of fact that y is child (enemy, neighbor, ancestor etc.) of x?" Again, in the case of such questions the answer may not be obvious and it may require empiric research, but usually *not* questioning but e.g. DNA test to find out if y is child of x. Then it may be found that "y is child of x" is true even if there is no knowledge (epistemic state) of this fact in either x nor y.

Principles of epistemic reasoning are usually formalized by *epistemic logic*, see e.g. [10]. Standard epistemic logic is based on introduction of notational convention Kxp, which we read as "x knows p", where x is a "knower" (i.e. conscious agent) and p is a proposition. The relation *knows* is problematic because of its vagueness – what exactly do we mean if we say "Person x knows person y"? What exactly does the person x know? What is the proposition p that s/he knows? We may use the approach inspired by [10, p. 6] and say $(\exists p)$ (p identifies person $y \wedge Kxp$) or in short form $(\exists p)$ $(p@I(y) \wedge Kxp)$ where $p@Q$

abbreviates "p answers question Q" and $I(y)$ is the question for identity of y. Using another approach we may conclude that the best way to formally model the vague *knows* relation is to model it by standard first-order predicate without epistemic extension and consider it normal empiric relation between two people.

Still these considerations about epistemic and ontological level of reasoning may provide us some help. For every predicate P in the *relationship vocabulary* we may ask whether the following proposition holds:

$$(\forall x)(\forall y)(P(x,y) \leftrightarrow Kx(P(x,y))) \tag{1}$$

That means that the relation P between persons x and y holds if and only if person x *knows* that relation P holds between persons x and y. This is a non-trivial assertion because while $(Kxp \rightarrow p)$ is the most general principle of epistemic logic, our proposition also says the reverse: that for a predicate P holds:

$$(\forall x)(\forall y)P(x,y) \rightarrow Kx(P(x,y)) \tag{2}$$

It is thus never the case that the assertion $P(x,y)$ evaluates to true without person x also knowing that it evaluates to true. Such feature of assertions may be true for some predicates but not for others. It means that such predicates are in a sense equivalent on epistemic and ontological level. Because of this we can refer to such a metaproperty as to 'being an *ontoepistemic* predicate'.

The predicates that do have such a feature are in many cases those describing our mental state. Such properties are usually called *mental properties* [11]. It might be true that ontoepistemic predicates are only mental predicates, however we are puzzled by properties such us apprenticeOf, which we believe are not pure mental (they may have social or institutional content) but still it seems unlikely that a person could be other person's apprentice without knowing it. We will postpone the solution of this theoretical problem to further investigation.

If we consider an ontoepistemic predicate then the domain of such a predicate is that of "knowers", and when the situation that $P(x,y)$ is true occurs then the knower x also knows it. Formally:

$$Oe(P) \equiv (\forall x)(\forall y)(P(x,y) \leftrightarrow Kx(P(x,y))) \tag{3}$$

E.g. the predicate *hates* is ontoepistemic because if x *hates* y then also x always knows that s/he hates y (this is an easy example because the *hates* property is mental). On the other hand the predicate *isFatherOf* is not ontoepistemic because there can be situations when x *isFatherOf* y but x does not know this.

We can now determine which of the predicates defined in the *relationship vocabulary* are ontoepistemic. We have also performed a detailed analysis of these predicates from the point of view of formal ontology. These results are presented along the *relationship vocabulary* definitions in Table 1. We also independently determined which of these properties are symmetric, asymmetric and antisymmetric, and compared the results with the *relationship vocabulary* definitions. In the first column there is the name of the property, the second presents what

superproperties this property has in the *relationship vocabulary* (we omitted differentFrom because it is defined as superproperty for all properties). The third column presents superproperties as based on our analysis. The column *Ontoepistemic* defines whether the property has this metaproperty. Column *RV sym.* contains information about symmetricity as defined in the *relationship vocabulary*, and the last column *Sym.* includes our results for symmetricity.

Property	**RV Super-prop.**	**Super-prop.**	**Ontoepistemic**	**RV Sym.**	**Sym.**
acquaintanceOf	k, kO	k, kO	yes	sym.	sym.
ambivalentOf	-	kO	yes[6]	-	asym.
ancestorOf	-	-	no	-	antisym.
antagonistOf	k, kO	kO	yes	-	asym.
apprenticeTo	k, kO	k, kO	yes	-	antisym.
childOf	k, kO	-	no	-	antisym.
closeFriendOf	k, kO	k, kO	yes	sym.	sym.
collaboratesWith	k, kO	k, kO	yes	sym.	sym.
colleagueOf	k, kO	-	no[7]	sym.	sym.
descendantOf	-	-	no	-	antisym.
employedBy	k, kO	kO	yes	-	asym.
employerOf	k, kO	-[8]	no	-	asym.
enemyOf	k, kO	kO	yes	-	asym.
engagedTo	k, kO	k, kO	yes	sym.	sym.
friendOf	k, kO	k, kO	yes	sym.	sym.
grandchildOf	k, kO	-	no	-	antisym.
grandparentOf	k, kO	-	no	-	antisym.
hasMet	k, kO	k, kO	yes[9]	sym.	sym.
influencedBy	-	-	no[10]	-	asym.
knowsByReputation	-	kO	yes	-	asym.
knowsInPassing	k, kO	kO	yes	-	asym.
knowsOf	-	kO	yes	-	asym.
lifePartnerof	k, kO	k, kO	yes	sym.	sym.
livesWith	k, kO	k, kO	yes[11]	sym.	sym.
lostContactWith	k, kO	kO	yes	sym.	asym.
mentorOf	k, kO	k, kO	yes	-	antisym.
neighborOf	k, kO	-	no	sym.	sym.
parentOf	k, kO	-	no	-	antisym.
siblingOf	k, kO	-	no	sym.	sym.
spouseOf	k, kO	k, kO	yes	sym.	sym.
worksWith	k, kO	-[12]	no	sym.	asym.
wouldLikeToKnow	-	kO	yes	-	asym.

Table 1. Properties of *relationship vocabulary*

[6] The definition says that x "has mixed feelings or emotions" towards y. We suppose that a conscious agent is aware of his/her feelings or emotions. Therefore s/he also knowsOf the person towards whom s/he has these emotions.

We have seen in Section 2 that according to recent update of *relationship vocabulary* property knows is now subproperty of knowsOf. We also know that property knows is symmetric while property knowsOf is asymmetric. The brief look at Table 1 reveals that these refinements were not reflected in the descriptions of properties. Properties that are asymmetric cannot be subproperties of symmetric property knows. This is an easy conclusion. More importantly – properties that are not ontoepistemic are, from point of view of formal ontology, not subproperties of property knowsOf. If a property P is not ontoepistemic then $P(x, y)$ does not imply that x knowsOf y. Again an example of such property P may be parentOf. However it is not necessary to think of such an issue as of mistake. We will show how to deal with it in the next section.

5 Supplying Additional Structures to Descriptions of Human Relations

A practical scenario for applying the previous ontological analysis is that of designing an application that would exploit FOAF data beyond their typical context (such as navigational browsing or social network visualisation). As we pointed out, due to problematic assumptions and implicit knowledge, such data could become semantically inconsistent when linked to other data; they should therefore undergo transformations. We are interested in transformations of linked data on human relations, mainly consisting in enriching them with additional information, which is implicitly present in the vocabularies.

We want to maintain *semantic consistency*, i.e. assure that the semantics of the data before and after transformation remains the same. These issues related to consistency may be classified to several categories based on their characteristics.

There are less important issues that we may characterize as typos or mere omission. These probably didn't have any impact on data created so far and are of merely of formal importance. In the definition of properties there is no differentiation between properties that are not symmetric and those that are antisymmetric. Also a more precise natural language description of relationship (maybe with examples) should be sometimes useful for general users to differentiate between such properties as collaboratesWith, worksWith, colleagueOf and

[7] The definition says: "A property representing a person who is a member of the same profession as this person." We suppose that usually people don't necessary know all people who are members of the same profession. It is also different from relation collaboratesWith, which requires symmetric knowledge of both persons involved.

[8] An employer who has thousands of employees usually does not know each of them.

[9] We understand this 'has met' as at least 'having been introduced to', i.e. not just 'having occurred at the same place in the same time'.

[10] A person doesn't necessarily know that s/he was (in his/her work etc.) been influenced by someone else.

[11] We understand it as a social relation, so it is ontoepistemic.

[12] This relation is defined as "a property representing person who works for the same employer as this person". This does not imply that they know each other.

similar. Our research using Swoogle revealed that some users are confused by these properties and use them incorrectly. Using inappropriate property simply because of confusion may introduce unnecessary semantic inconsistency to data.

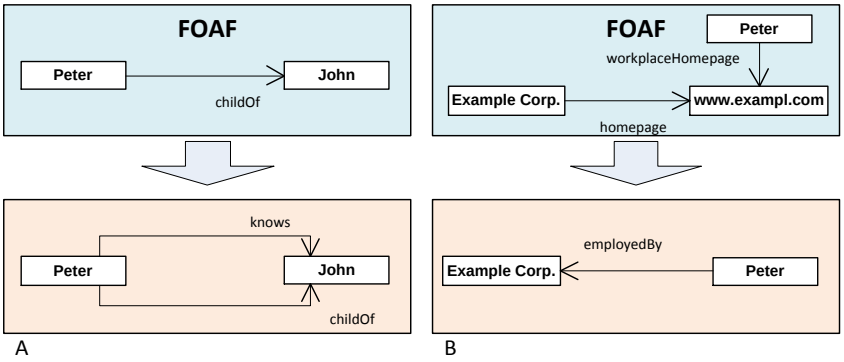

Fig. 5. Transformation patterns.

When considering FOAF data in the context of a different, semantically sounder ontology, it is not necessary to e.g. understand FOAF's internal declaring of properties like childOf subproperty of knows to be an ontological engineering mistake. Rather we could understand it as stating some additional knowledge. While from the point of view of formal ontology the relation childOf does not imply the relation knows, we propose that we should approach FOAF formal property definitions as stating a specific kind of *prior* knowledge: we should understand the statement describing childOf as subproperty of knows as declaring that whenever we have a FOAF statement that (x childOf y) we implicitly assert that (x knows y). If we accept such understanding then we could use our Table 1 as basis for developing transformation patterns that can be used to supply additional structures to FOAF data, without committing to FOAF modelling in general (for data coming from other namespaces). The differences between columns *RV Super-prop.* and *Super-prop.* may help us identify implicit *a priori* knowledge that has to be taken care of when performing such transformation. An example of a simple transformation pattern that makes implicit knowledge explicit is in Figure 5A. We can also say that according to our analysis properties that are ontoepistemic are subproperties of property knowsOf, so when using FOAF data in an application we should use a similar appropriate pattern or at least check whether the target data structure reflects such a priori constraints.

Another transformation pattern can be easily designed to overcome some semantic limitations of FOAF mentioned in Section 2. An example of such pattern is in Figure 5B. Here we concatenate two properties into another one, i.e. infer the 'employment' relationship from the 'mediating' webpage.

Finally, we may also proceed somewhat the other way around: 'unfold' a complex relationship from a simple FOAF relationship. For example, many

FOAF relationships, such as knowing a person by having met him/her, or being someones collaborator on a project, can be modelled by an *event-participation pattern*. Such an 'unfolding' transformation, relying on additional hints, may be used to disambiguate or enrich the semantic content of relations, based on transformation-based reference to complex content pattern reflecting the internal structure and semantics of the relation. Similarly as suggested above, the formal characteristics in Table 1 may be used for extended checking of semantic consistency during knowledge transformation or injection.

6 Conclusions

We have analyzed FOAF and its extensions for describing human relations from point of view of formal ontology. We focused on the property knows and pointed out some important issues. We also analyzed the ontological structure of the *relationship vocabulary* extension of FOAF, and identified some confusing ontological definitions. Detailed analysis of its properties revealed some interesting characteristics and assumptions that we believe are not generally valid. We pointed out that these could be understood as *a priori* knowledge and when exploiting FOAF data in an external context we must use appropriate transformation patterns. We have also presented examples of such transformation patterns and formulated directions of following research.

Acknowledgments

This work has been partially partially supported by the IGS 4/2010 and by the CSF grant no. P202/10/1825 (PatOMat - Automation of Ontology Pattern Detection and Exploitation).

References

1. Tim Berners-Lee. LinkedData. `http://www.w3.org/DesignIssues/LinkedData.html`, 2009.
2. Tim Berners-Lee, Yuhsin Chen, Lydia Chilton, Dan Connolly, Ruth Dhanaraj, James Hollenbach, Adam Lerer, and David Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *Proceedings of the 3rd International Semantic Web User Interaction*, 2006.
3. Christian Bizer, Richard Cyganiak, and Tom Heath. How to publish linked data on the web, 2007.
4. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2009.
5. Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In Huai et al. [8], pages 1265–1266.
6. Dan Brickley and Libby Miller. FOAF Vocabulary Specification 0.97. `http://xmlns.com/foaf/spec/`, 1 2010.

7. Li Ding, Lina Zhou, Tim Finin, and Anupam Joshi. How the semantic web is being used: An analysis of foaf documents. In *In Proceedings of the 38th International Conference on System Sciences*, 2005.
8. Jinpeng Huai, Robin Chen, Hsiao-Wuen Hon, Yunhao Liu, Wei-Ying Ma, Andrew Tomkins, and Xiaodong Zhang, editors. *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*. ACM, 2008.
9. Olaf Noppens. Negative property assertion pattern (npas). In *Proc. 1st ISWC 2009 workshop on Ontology pattern (WOP), Chantilly (VA US)*, 2009.
10. Nicholas Rescher. *Epistemic logic: a survey of the logic of knowledge*. Univ of Pittsburgh Press, 2005.
11. David Robb and John Heil. Mental Causation. In Edward N. Zalta, editor, *Stanford Encyclopedia of Philosophy*. 2008.
12. Leo Sauermann, Richard Cyganiak, and Max Völkel. Cool uris for the semantic web. Technical Memo TM-07-01, DFKI GmbH, Deutsches Forschungszentrum f ür Künstliche Intelligenz GmbH, February 2007. Written by 29.11.2006.
13. Vojtech Svatek, Ondrej Svab-Zamazal, and Valentina Presutti. Ontology naming pattern sauce for (human and computer) gourmets. In *Proc. 1st ISWC 2009 workshop on Ontology pattern (WOP), Chantilly (VA US)*, 2009.