Noname manuscript No.
(will be inserted by the editor)

Knowledge Management in Semantic Social Networks

Markus Schatten

Received: date / Accepted: date

Abstract An object-oriented model of semantic social networks is proposed and formally analyzed. Methods for project, role, and team management based on the semantic model are defined and implemented in niKlas, a semantic wiki language based on frame logic developed by the author. The new approach to semantic social networks allows dynamic change of social network semantics and the establishment of the well known fishnet organization in a social network. In the end possible applications to knowledge management are presented.

Keywords semantic social networks · role management · team management · project management · knowledge management · fishnet organization

1 Introduction

The main aim of this paper is to present a formalism that would allow for an integration of social networks and semantics - namely semantic social networks - and to show what possible benefits such a formalism would have to knowledge management (KM). The motivation for defining semantic social network models in the context of KM is straightforward: KM deals with the management of knowledge produced by social entities. Thus we define the relations between these entities as a social network and try to formalize their knowledge using semantic descriptors in order to allow for automated querying of the emerging social knowledge. Various Web 2.0, Semantic Web and Web 3.0 systems and approaches have been proposed for KM including blogs, folksonomies, RSS feeds, wiki and semantic wiki systems, different social media as well as others.[1] A system that we want to point out here is the ⊤AOPīs system that has already been described for its semantic wiki

Markus Schatten
University of Zagreb,
Faculty of Organization and Informatics,
Pavlinska 2, 42000 Varaždin, Croatia
Tel: +385 42 390 891
Fax: +385 42 213 413
E-mail: markus.schatten@foi.hr

[1] Please refer to (Tredinnick, 2006) for a good review.

subsystem (Schatten et al, 2008, 2009a,b), KM facilities (Žugaj and Schatten, 2008; Maleković and Schatten, 2008; Žugaj and Schatten, 2009) as well as project management (Schatten and Žugaj, 2007).

The ┳AOPīs system, which is a social semantic Web or Web 3.0 application, aiming to provide a platform for self-organizing communities for which it provides suitable tools like semantic wiki systems, forums, blogs, ranking mechanisms, content filtering, tagging etc., uses frame logic (Kifer et al, 1995) and especially the $\mathcal{F}$LORA-*2* reasoning engine (Yang et al, 2003) as its main formalism. It starts off with a completely object-oriented model of knowledge similar to the one described in (Martin and Odell, 1998). It uses a self developed formatting language entitled niKlas that allows for inline queries against a dynamically created social knowledge base.

Even if there have been very good reports on the usage of social networks and social network analysis in KM (Mueller-Prothmann and Finke, 2004; Von Kortzfleisch et al, 2007; Jones, 2001) none of them tried to establish a formalism that would be able to combine insights from the Semantic Web (Berners-Lee et al, 2001) and allow for automated reasoning in a Web 3.0 environment. Thus a completely different approach will be taken here. A part of the ┳AOPīs system, namely its semantic social network model, hasn't been reported on yet. The system makes use of the object-oriented social network model in various ways, which shall be described in this paper. Besides the usage of the model in niKlas queries, intelligent agent implementation shall also be demonstrated. The presented formalisms will be put into an organizational KM context by organizing a dynamic fishnet structure, role management and knowledge based team management.

First we will give a literature review in order to establish the context of this paper in section 2. Afterwards we shall make our selves familiar with the concept of a fishnet organization which is described in section 3. Section 4 presents the object-oriented framework of ┳AOPīs that will allow us to define our model of a semantic social network in section 5 and create such an organization. Afterwards in section 7 we put the semantic social network model to use in defining user roles and their dynamic network dependent changes. Section 8 shows how the model can be applied to organize teams depending on their particular knowledge and skills.[2] In section 9 we show how the previously defined methods can be implemented using niKlas , $\mathcal{F}$LORA-*2* and Python as a little helper language. In section 10 we will evaluate our approach and compare it to other similar formalisms. In the ending section 11 we draw our conclusions and give guidelines for future research.

2 Related Work

There have been interesting reports and publications in the field of Semantic Web and social networks integration. Various reports illustrate the need for social network metadata within semantic metadata by arguing about FOAF (the Friend of a Friend project) and/or XFN (XHTML Friends Network) (Downes, 2005; Finin et al, 418-435; Breslin and Decker, 2007). Most of the literature (Golbeck et al, 2003; Downes, 2005; Finin et al, 418-435; Mika, 2005; Aleman-Meza et al, 2006;

[2] It should be stated here that teams or subgroups can be organized by any imaginable set of properties, not only skills and knowledge.

Jung and Euzenat, 2007), however deals with or is based on FOAF which aims on making "*... it easier to share and use information about people and their activities (...), to transfer information between Web sites, and to automatically extend, merge and re-use it on-line*" (Brickley and Miller, 2000). FOAF establishes a simple ontology for describing people on the social Web, and is the *de facto* standard in the field, which is probably the reason of intensive publishing.

Mika (2005), for example, demonstrates the use of semantic technologies in extraction, aggregation and visualization of on-line social networks. Similar studies like (Cantador and Castells, 2006; Gloor and Zhao, 2006) also implement various Web mining techniques in order to extract the semantics from the underlying social structure. While it is important to analyze existing on-line social networks, the process of mining (often unstructured or semi-structured) data, inevitably causes a loss of the social system's semantics. Especially, if FOAF is used as a facet for filling in data, all the other data (which isn't described by FOAF) is dismissed. The approach presented herein does not rely on FOAF, but implements an evolving ontology of its own, which allows it to store and take care of all collected semantics in one place. Additionally, due to extended expressiveness, the new formalism is able to emulate FOAF (e.g. part of the emerging knowledge-base can be exported to FOAF if needed).

Few studies have dealt with reasoning in social networks. Aleman-Meza et al (2006) use semantic technology and social networks to conclude about conflict of interest in scientific review procedures, while Jung and Euzenat (2007) try to infer relations between people, ontologies and concepts by extracting similarities between them. The process of automated deduction is important, since it can yield additional semantics about the social network. Automated reasoning about Semantic Web data relies on the implicit presumption that the given data is true, which doesn't have to be the case. Especially in a social network environment, where eventually lots of different people interact, this problem becomes more obvious. Thus it does not wonder that Golbeck et al (2003), for example, try to establish a Web of trust for the Semantic Web by asking the important question "How to trust a given ontology?". The formalism presented herein, will take the issue of trusting user supplied data very seriously. We will build on the presumption that a statement of some user is as trustworthy as is the user inside the social network he participates in.

None of the cited work deals with the establishment of a formal semantic social network model and thus isn't comparable to the model proposed herein. Mika (2007) on the other hand, established a tripartite ontology model called Actor–Concept–Instance model. The model is based on principles observed on special Web 2.0 applications called folksonomies (from folk and taxonomy) like Delicious (Schachter, 2003) or CiteULike (Oversity, 2004). His tripartite hypergraph model consists of three sets of verticles $A = \{a_1, a_2, \ldots, a_k\}$ - the set of actors, $C = \{c_1, c_2, \ldots, c_l\}$ - the set of concepts or classes and $I = \{i_1, i_2, \ldots, i_m\}$ - the set of instances or objects. A folksonomy is defined as the a subset of the Cartesian product $T \subseteq A \times C \times I$. In effect, a tuple $(a, c, i) \in T$ means actor a uses concept c to classify instance i. The representing hypergraph of a folksonomy is then defined as a simple tripartite hypergraph $H(T) = (A \cup C \cup I, \{\{a, c, i\} | (a, c, i) \in T\})$. The tripartite graph can be reduced into three bipartite graphs, namely AC (the graph of actors and concepts), AI (the graph of actors and instances) and CI (the graph of concepts and instances). By using the respective adjacency matrices of

these graphs one can now calculate various new networks like the co-affiliation matrix $(|AC||AC|')$[3] e.g. a social network connecting people with shared concept affiliations.

There are other formalisms not strictly related to the Semantic Web, that are nevertheless similar to both the formalism outlined herein as well as Mika's approach. One such formalism is the MetaMatrix model (Krackhardt and Carley, 1998; Carley, 2002, 2003). The MetaMatrix model proposes to structure all organizations into four domains: (1) People (individuals, personnel), (2) Knowledge (skills, resources), (3) Events (tasks) and (4) Organizations (groups). By combining these domains pairwise for an organization various networks emerge that then can be analyzed in various ways (table 1).

Table 1 The MetaMatrix model

Meta-Matrix entities	**People**	**Knowledge/ Resources**	**Events/ Tasks**	**Groups/ Organizations**
People	Social network	Knowledge Network/ Resource Network	Attendance Network/ Assignment Network	Membership network
Knowledge/ Resources		Information Network/ Substitution Network	Needs network	Organizational capability
Events/ Tasks			Temporal Ordering/ Task Flow/ Precedence	Institutional support or attack
Organizations				Inter-organizational network

Each of the networks defines a matrix. For example the knowledge network defines the matrix $K = \text{People} \times \text{Knowledge}$ where a 1 in position i, j indicates that person i has knowledge j. If we multiply K by K' we get a matrix KK' which tells us for each cell how many knowledge artifacts individuals i and j have in common. Or if we multiply the information network's matrix I as KIK' we acquire for a particular person i all the people who have similar or interdependent knowledge.

Both the Actor–Concept–Instance and the MetaMatrix model are fairly similar to the approach outlined herein. In fact, semantic social networks as defined in this article, can be used to simulate both of them which implies that the presented formalism is more expressive. On the other hand, neither of the two models allow for automated reasoning or additional semantics. We will come back to these two models in section 10 and present a detailed comparison.

[3] Here by $|X|$ we mean the adjacency matrix of graph X.

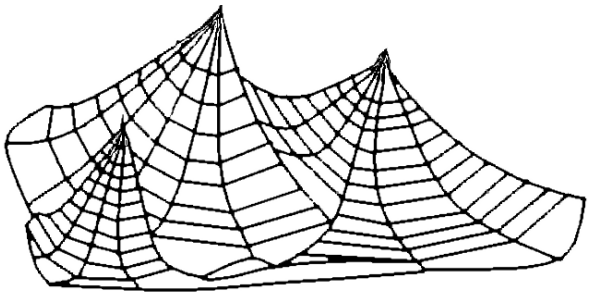

Fig. 1 The fishnet organization (Johansen and Swigart, 2000)

3 The Fishnet Organization

The concept of a heterarchic organization (or network organization) is based on the following principles: an organization consisting of organizational units[4] that are mutually connected through information links (often based on modern information technology), are mutually independent, heterarchically organized (as opposed to hierarchy), and operate internally and externally (with their environment) in most cases sharing some common goal (Žugaj and Schatten, 2005, p. 106).

The idea of a heterarchical organization comes from the neuropsychological research of the human brain conducted by Warren McCulloch in 1945. He concluded that the human brain must have a heterarchical organization as opposed to previously defined hierarchical models, and described this organization as a neural network which is specifically designed for parallel information processing (Reihlen and Rohde, 2002, p. 3).

If we apply such a concept to an organization, we get a structure which interrelationships are not strictly defined, but rather activated, or self regulated depending on the particular situation (Žugaj and Schatten, 2005, p. 106).

An interesting metaphor for this kind of organization is the fishnet organization, depicted on figure 1. If we observe a fisher's net on the coast, it seems completely non-hierarchical, but if we take one node and lift it up, we get a hierarchical structure. By lifting further nodes and putting down the old ones, we can see the dynamical creation of new and the destruction of old hierarchical structures. Thus the fishnet organization tries to combine the modern concept of heterarchy and the usual human habit of tendency to hierarchy and order (Žugaj and Schatten, 2007).

The fishnet organization is particularly interesting to knowledge management, if we presume that organizations in a turbulent environment have to use their best knowledge to take advantage of opportunities. Thus the dynamically created hierarchies can be implemented as projects, teams and other organizational units that are established dynamically depending on environmental conditions. The main question is how to identify teams that have the best knowledge needed for a given situation? How should the hierarchy be established, or how to find people who will

[4] Organizational units can in this context be either individuals, teams, departments, divisions and even entire organizations or groups of organizations by the fractal organization principle (Žugaj and Schatten, 2005, pp. 149-151)

fit best into various leadership roles? We shall tackle these questions by defining a new semantic social network model.

4 ┬aOPīs Object-Oriented Framework

Before establishing the semantic social network model, we need to introduce the object-oriented framework that has been partially described in (Schatten et al, 2008). Most Web 3.0 systems allow users to add semantics to content published on the system through different (often predefined) meta-information. While predefined structure (in form of ontology schema, class hierarchy, taxonomy, vocabulary etc.) fosters knowledge base consistency, it greatly limits the expressiveness of the formal language available to system's users. We believe that structure in social systems and likewise in Web 3.0 applications (which are a reflection of the former) is an emergent phenomenon, and thus Web 3.0 systems need to provide as little as possible language constraints or no constraints at all. Guided by this principle the formalism in ┬aOPīs is open to any modifications needed by a particular domain. Users are the ones expected to organize their knowledge about their organization (domain) and thus the knowledge bases on ┬aOPīs are dynamic (e.g. changeable in time), self-organizing (their structure and semantics is emergent), and their only validation mechanism is their actual usability as perceived by the users.

To develop a capable formalism we decided to use a simple tagging system to allow users to approach content in an object-oriented manner. There are two reasons for that: firstly tagging systems are easy to use and very well distributed among lots of social applications; secondly among various knowledge representation formalisms we consider object-orientation to be intuitive enough to be understood by most users and yet formal enough to provide sufficient reasoning facilities.

In order to consider a domain of interest in an object-oriented fashion one needs to be able to model specific concepts like objects, classes (types, concepts), relations (links, tuples), attributes (properties), methods (actions, reactions, messages), states etc. Thus we provide the following conceptualization:

Let the whole content stored on the system be a domain of interest D. Objects inside this domain are for example specific wiki pages, users, projects, forum posts, blog posts and others, having their particular classes, relations, attributes, methods etc. Any object on creation is generic, meaning that users can specialize them in order to reflect the domain of interest. Thus the domain D is an extensible set of objects as shown in the following equation:

$$D = \{o_1, o_2, ..., o_n\}$$

To allow concretization of generic objects we introduce attribute-value tags that reflect specific characteristics of objects inside a domain. Any object can be thought of as a relation that consists of a finite number of attribute-value tuples, as shown in equation:

$$o_i = \{(a_1, v_1), (a_2, v_2), ..., (a_m, v_m)\}$$

This set also includes standard attributes like author(s), title, body, name surname, e-mail address etc. There is a small predefined vocabulary in ┬aOPīs that is shown on figure 2, which can be overridden/changed/extended. Nevertheless, the

predefined vocabulary isn't necessary for the formalism depicted here: any system based on it can define it's own vocabulary or work without any vocabulary at all. The vocabulary in ᴛᴀOPīs was included merely to reflect the system's internal relations and class hierarchy.

Fig. 2 Predefined class hierarchy in ᴛᴀOPīs

We also introduce object's relations to be defined as labeled outgoing links of any object whether to other objects on the system or to pages outside of ᴛᴀOPīs which can be used for knowledge base amalgamation purposes. These relations are reflected as additional attribute-value pairs whereby the label represents the attribute and the value the object the relation links to. Thus the set of an objects outgoing relations is shown in the following equation.

$$\mathrm{rel}(o_i) = \{(r_1, o_{r1}), (r_2, o_{r2}), ..., (r_l, o_{rl})\}$$

In the end we introduce a set of methods represented through web services or script extensions of the form $(\mathrm{m}_i(p_{i1}, p_{i2}, ..., p_{ia_i}), res_i)$ where m_i is the methods name, $p_{i1}, ..., p_{ia_i}$ are the methods parameters, a_i is the methods arity, and res_i is the methods return value. These methods are represented through the set:

$$\begin{aligned}\mathrm{met}(o_i) = \{\ & (\mathrm{m}_1(p_{11}, p_{12}, ..., p_{1a_1}), res_i), \\ & (\mathrm{m}_2(p_{21}, p_{22}, ..., p_{2a_2}), res_2), \\ & \vdots \\ & (\mathrm{m}_l(p_{k2}, p_{k2}, ..., p_{ka_k}), res_k)\ \}\end{aligned}$$

Such a conceptualization allows us to map the meta data of a Web 3.0 system to frame logic syntax which is defined as follows (Kifer et al, 1995; Yang et al, 2003):

Definition 1 The alphabet of an F-logic language $\mathcal{L}$ consists of:

- a set of object constructors, $\mathcal{F}$;
- an infinite set of variables, $\mathcal{V}$;
- auxiliary symbols, such as, (,), [,], $\rightarrow$, $\twoheadrightarrow$, $\bullet\!\!\rightarrow$, $\bullet\!\!\twoheadrightarrow$, $\Rightarrow$, $\Rightarrow\!\!\!\!\!\!\Rightarrow$, etc.; and
- usual logical connectives and quantifiers, $\vee$, $\wedge$, $\neg$, $\longleftarrow$, $\forall$, $\exists$.

Object constructors (the elements of $\mathcal{F}$) play the role of function symbols in F-logic whereby each function symbol has an arity. The arity is a non-negative integer that represents the number of arguments the symbol can take. A constant is a symbol with arity 0, and symbols with arity ≥ 1 are used to construct larger terms out of simpler ones. An id-term is a usual first-order term composed of function symbols and variables, as in predicate calculus. The set of all variable free or ground id-terms is denoted by $U(\mathcal{F})$ and is commonly known as Herbrand Universe. Id-terms play the role of logical object identities in F-logic which is a logical abstraction of physical object identities.

A language in F-logic consists of a set of formulae constructed out of alphabet symbols. As in a lot of other logics, formulas are built out of simpler ones by using the usual logical connectives and quantifiers mentioned above. The most simple formulas in F-logic are called F-molecules.

Definition 2 A molecule in F-logic is one of the following statements:

- An is-a assertion of the form $C :: D$ (C is a non-strict subclass of D) or of the form $O : C$ (O is a member of class C), where C, D and O are id-terms;
- An object molecule of the form O [a ';' separated list of method expressions] where O is a id-term that denotes and object. A method expression can be either a non-inheritable data expression, an inheritable data expression, or a signature expression:
  - Non-inheritable data expressions can be in either of the following two forms:
    - A non-inheritable scalar expression $ScalMethod@Q_1, ..., Q_k \rightarrow T, (k \geqslant 0)$.
    - A non-inheritable set-valued expression $SetMethod@R_1, ..., R_l \twoheadrightarrow \{S_1, ..., S_m\}$ $(l, m \geqslant 0)$.
  - Inheritable scalar and set-valued expression are equivalent to their non-inheritable counterparts except that $\rightarrow$ is replaced with $\bullet\!\!\rightarrow$, and $\twoheadrightarrow$ with $\bullet\!\!\twoheadrightarrow$.
  - Signature expression can also take two different forms:
    - A scalar signature expression $ScalMethod@V_1, ..., V_n \Rightarrow (A_1, ..., A_r)$, $(n, r \geqslant 0)$.
    - A set valued signature expression $SetMethod@W_1, ..., W_s \Rightarrow\!\!\!\!\Rightarrow (B_1, ..., B_t)$ $(s, t \geqslant 0)$.

All methods' left hand sides (e. g. Q_i, R_i, V_i and W_i) denote arguments, whilst the right hand sides (e. g. T, S_i, A_i and B_i) denote method outputs. Single-headed arrows ($\rightarrow$, $\bullet\!\!\rightarrow$ and $\Rightarrow$) denote scalar methods and double-headed arrows ($\twoheadrightarrow$, $\bullet\!\!\twoheadrightarrow$ and $\Rightarrow\!\!\!\!\Rightarrow$) denote set-valued methods.

Definition 3 Let att(o) be the set of attribute-value pairs of object o, rel(o) the set of relation-object's identifier pairs of object o, and met(o) the set of methods-return value pairs. An *object* with id-term o in ᴛᴀOPīs then is represented with the F-molecule:

$$
\begin{aligned}
o[\\
& a_1 \rightarrow v_1; \\
& a_2 \rightarrow v_2; \\
& \vdots \\
& a_m \rightarrow v_m; \\
& r_1 \rightarrow o_{r1}; \\
& r_2 \rightarrow o_{r2}; \\
& \vdots \\
& r_l \rightarrow o_{rl}; \\
& \mathrm{m}_1(p_{11}, p_{12}, ..., p_{1a_1}) \Rightarrow res_1; \\
& \mathrm{m}_2(p_{21}, p_{22}, ..., p_{2a_2}) \Rightarrow res_2; \\
& \vdots \\
& \mathrm{m}_k(p_{k1}, p_{k2}, ..., p_{ka_k}) \Rightarrow res_k \\
&].
\end{aligned}
$$

Such a definition implies that attribute-value tags (a_i, v_i) associated with a given object or an object with id-term o, as well as relation-object's identifier pairs are considered to be *non-inheritable scalar methods* whereby the attribute (a_i) is the methods name that has no arguments $(k = 0)$ and values (v_i) to be outputs. If there are more than one equivalent attributes or relation names for a given object with distinct values than the method is considered to be a *non-inheritable set-valued method*. In the end signature expressions are considered to be web services and script extensions that act as methods of a specific object.

Now we are able to introduce special attributes labeled with common object-oriented programming constructs like `class`, `subclass`, `rule` etc. Such attributes are used to provide additional semantics to the dynamically constructed domain ontology. Special attribute labels like `class` and `subclass` are used to create is-a assertions. For example a wiki page tagged with the attribute `class` and value `student` is considered to be an object that is a member of class `student`. If the same object is additionally tagged with the attribute `subclass` and value `person` than the class `student` is considered to be a non-strict subclass of class `person`. All other tags provided on a wiki page are the special attributes of this object, thus the object mentioned previously if additionally tagged with tags like `name:Foo`, `surname:Bar`, `address:Linus Lane 27` would yield the following sentence in a F-logic knowledge base:

$$
\begin{aligned}
student &:: person \ \wedge \\
o_x &: student \ [\\
name &\rightarrow 'Foo'; \\
surname &\rightarrow 'Bar'; \\
address &\rightarrow 'Linus\ Lane\ 27' \ ]
\end{aligned}
$$

Where o_x denotes the logical object-id of the wiki page under consideration. Thus, classes and class hierarchy are created dynamically by the users of the system tagging specific objects (e.g. other users, wiki pages, blog comments etc.).

Definition 4 An object with id-term o is considered to be a member of class c if its corresponding F-molecule contains a non-inheritable scalar method class→c.

Definition 5 A class c_1 is considered to be a non-strict subclass of class c_2 if there is at least one object that is a member of class c_1 which corresponding F-molecule contains a non-inheritable scalar method subclass→c_2.

Definition 6 A given object with id-term c, which corresponding F-molecule contains the attribute `class` which value is also `class`, is considered to be a class descriptor. All its attribute-value pairs are converted to inheritable scalar expressions (scalar or set-valued depending on the number of equivalent attributes with distinct values) except for the `class:class` pair.

In this way we allow for meta modeling, by stating that instances of class `class` are classes of their own. For instance if a forum post entitled `car` is tagged with the tags `class:class`, `model:string`, `color:string`, and `year:integer` it would correspond to the following sentence in frame logic:

$$\begin{aligned} &car[\\ &model \bullet\!\!\rightarrow string; \\ &color \bullet\!\!\rightarrow string; \\ &year \bullet\!\!\rightarrow integer] \end{aligned}$$

Such interpretation allows us then to create instances of such a defined class as well as to define the schema of a domain of a given wiki system.[5]

Another important concept is the definition of rules. Rules are defined in terms of objects tagged with special attribute `rule`).

Definition 7 If some object is tagged with special attribute `rule`, and value of the form `Head :- Body` then this attribute-value pair is removed from the object descriptor and the following rule is added to the knowledge-base:

$$Head \leftarrow Body$$

For instance if some wiki page was tagged with `rule : ?x:boy :- ?x:person[ sex->male, age->?a ], ?a<18` the following rule would be added to the knowledge-base:

$$?x \ : \ boy \ \leftarrow ?x \ : \ person[\ sex \rightarrow male \ ; \ age \rightarrow ?a \ ] \wedge ?a \ < \ 18$$

Since we were able to map concepts from ŧAOPīs to concepts from F-logic we can now state that the syntax of ŧAOPīs is equivalent to the syntax of F-logic defined above.

[5] The schema (defined or inferred) is used in ŧAOPīs for input suggestion mechanisms. Such mechanisms try to minimize syntactic errors due to different user input. For a better understanding of such input mechanisms please refer to (Schatten et al, 2009a)

5 Semantic Social Networks

Social networks are comprised of sets of interconnected nodes or agents. Nodes represent people while the relations between them represent their mutual interconnections. When adding annotations to such relations comprised of values from some extensible set (e.g. friend, co-worker in organization X or project Y, colleague, family member, brother, sister, town, street etc.) one adds additional semantics to such networks that allows us to divide the network into subnetworks (e.g. the network of all co-workers on project Y).

We shall take a more formal approach to defining semantic social networks using graph theory.[6] A social network can be represented as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ denotes the set of actors, and $\mathcal{A}$ denotes the set of relations between them. If the relations are directed (e.g. support, influence, message sending etc.) we can conceptualize a social network as a directed graph. If the relations additionally can be measured in a numerical way, social networks can be represented as valued digraphs.

One of the main applications of graph theory to social network analysis is the identification of "most important" actors inside a social network. There a lots of different methods and algorithms that allow us to calculate the importance, prominence, degree, closeness, betweenness, information, differential status or rank of an actor.[7] Herein we would like to outline one of such metrics introduced by Bonacich (1972) called eigenvector centrality, but for our purpose any other metric can be used that can yield an approximation of the probability that a certain person will say the truth in a meta data statement.

By modeling a domain (be it using an ontology, knowledge base, UML[8] diagram or any other formalism) one expresses his own knowledge about the domain. This in particular means that the main concept in modeling is knowledge. I. Nonaka once stated that knowledge is personal, a "justified true belief" (Nonaka and Takeuchi, 1995). Thus one implicitly presumes that the data one expresses in his domain model is true. If one asks now *what is the truth?* he comes to one of the fundamental questions in philosophy. F. Nietzsche argued that one cannot prove the truth which is nothing more than the invention of fixed conventions for merely practical purposes, especially those of repose, security and consistence (Nietzsche, 1873). According to this view, no one can prove that this article isn't just a fantasy of the reader reading it.

Nonaka's definition includes, by intention or not, two more crucial words: *justified* and *belief*. An individual will consider something to be true that he believes in, and from that perspective, the overall truth will be a set of statements that the community believes in. This mutual belief makes this set of statements justified. The truth was once that the Earth was flat until philosophers and scientists started to question that theory. The Earth was also once the center of the universe. So an interesting fact about the truth, from this perspective, is that it evolves depending on the different beliefs of a certain community.

In an environment where a community of individuals collaborates in modeling a domain there is a chance that there will be disagreements about the domain

[6] There are off course other approaches like sociometrics.

[7] See (Wasserman and Faust, 1994) for an in depth discussion of such metrics.

[8] Unified Modeling Language

which can yield certain inconsistencies in the model. A good example of such disagreements are the so called "editor wars" on Wikipedia the popular free online encyclopedia. A belief about the War in Iraq will most likely differ between an American and an Iraqi, but they will most probably share the same beliefs about fundamental mathematical algebra.

Following this perspective, our conceptualization of meta data statements as units of formalized knowledge, will consider the probability of giving a true statement a matter of justification. A person is justified if other members of a social system believe in his statements. Bonacich takes a similar approach, and gives a metric that calculates the centrality of a node based on the centrality's of its adjacent nodes. Eigenvector centrality assigns relative values to all nodes of a social network based on the principle that connections to nodes with high values contribute more to the value of the node in question than equal connections to nodes with low values.

PageRank is a variant of the Eigenvector centrality measure, which we decided to use herein (Brin and Page, 1998; Page et al, 1999). A very convenient feature of PageRank is that the sum of all ranks is 1. Thus, semantically, we can define the ranking value of persons (or actors in the social network) participating in a given environment as *the probability that a person will say the truth in the perception of the others*. In the following we will use the ranking, obtained through such an algorithm in this sense as the trust of the social network towards the particular actor in question.

We are now able to define a basic typed (or tag annotated) semantic social network.

Definition 8 Let $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ be an extensible set of relation types or tags. Let $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be a set of actors or nodes whereby each node corresponds to one object as defined in definition 3, and let $\mathcal{E} = \{(\alpha_i, \alpha_j, t) | \alpha_i, \alpha_j \in \mathcal{A}, t \in \mathcal{T}\}$ be a set of tag annotated edges. A basic typed or tag annotated semantic social network $\mathcal{SSN}$ is defined as the triple $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$.

This definition implies that each node is a generic object that is defined by the relations (edges) it participates in. Between any two nodes there can be an arbitrary number of edges up to $|\mathcal{T}| = m$. For example consider the following simple semantic social network on figure 3.

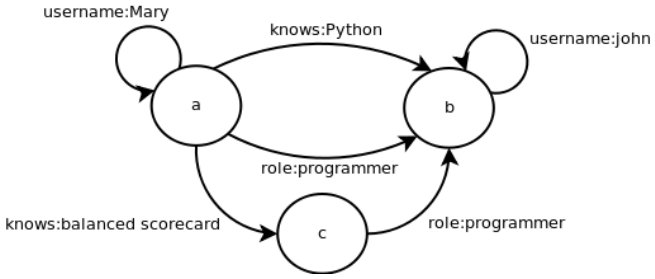

Fig. 3 Basic typed semantic social network example

The semantics of the network are as follows:

- Actor a thinks her own username is Mary.
- Actor a thinks that actor b knows Python and that his role should be programmer.
- Actor a thinks that actor c knows balanced scorecard.
- Actor b thinks his own username is John.
- Actor c thinks that actor b's role should be programmer.

Following the given definitions 3 and 8 node b would correspond to the following F-molecule:

$$\begin{aligned} b \quad & [\\ username & \rightarrow John; \\ role & \rightarrow programmer; \\ knows & \rightarrow Python\ ] \end{aligned}$$

The F-molecule corresponds to the current state of the semantic social network. If the network would change in any way (addition/deletion/change of new nodes, edges or tags) so would the corresponding knowledge base.

A problem with this definition is that it implicitly subsumes that all actors will always say the truth about them selves and other actors in the network, which doesn't have to be the case. This is why we need to extend the definition in order to include trust. To do so we need to define trust as a metric that will reflect the current state of the social network.

Definition 9 Let $t_{\text{trust}} \in \mathcal{T}$ be a distinct tag. Let $(\alpha_i, \alpha_j, t_{\text{trust}}) \in \mathcal{E}$ denote that actor α_i trusts actor α_j. Let $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$ be a basic typed semantic social network. The trust level or rank $\pi(\alpha)$ of some actor α is defined as any function $\pi : \mathcal{A} \longrightarrow [0, 1]$.

The trust level function associates any actor with his respective rank depending on the social network. As indicated above, the trust level is interpreted as the probability that a given member of a social network will say the truth in the perception of the others. In ᴛAOPīs PageRank is used as the trust level function, but any other metric could be used for this purpose. Now we are able to define a probability annotation to any relation from $\mathcal{E}$.

Definition 10 Let $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$ be a basic typed semantic social network. Then the positive annotation $t_{\alpha_i} \bar{\bar{\wedge}} \Sigma^+(t_{\alpha_i})$ of a tag on some actor α_i, t_{α_i} is defined as follows:

$$\Sigma^+(t_{\alpha_i}) = \sum_{(\alpha_i, \alpha_j, t_{\alpha_i}) \in \mathcal{E}} \pi(\alpha_j)$$

An extension to such a probability annotation is the situation when tags are of negative valency. This happens when a particular user disagrees to a meta data statement of another user. Such an annotation would be defined as follows:

Definition 11 Let $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$ be a basic typed semantic social network. Then the negative annotation $t_{\alpha_i} \overline{\overline{\wedge}}\, \Sigma^-(t_{\alpha_i})$ of a tag on some actor α_i, t_{α_i} is defined as follows:

$$\Sigma^-(t_{\alpha_i}) = \sum_{(\alpha_i, \alpha_j, -t_{\alpha_i}) \in \mathcal{E}} \pi(\alpha_j)$$

To make use of this two definitions we define the complete annotation:

Definition 12 Let $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$ be a basic typed semantic social network. Then the complete annotation $t_{\alpha_i} \overline{\overline{\wedge}} \Sigma(t_{\alpha_i})$ of a tag on some actor α_i, t_{α_i} is defined as follows:

$$\Sigma(t_{\alpha_i}) = \begin{cases} \Sigma^+(t_{\alpha_i}) - \Sigma^-(t_{\alpha_i}) & \text{if } \Sigma^+(t_{\alpha_i}) > \Sigma^-(t_{\alpha_i}) \\ 0 \quad \text{if } \Sigma^+(t_{\alpha_i}) \leqslant \Sigma^-(t_{\alpha_i}) \end{cases}$$

Such a definition is needed in order to avoid possible negative probability (the case when dissagreement is greater then approval).

Now we are able to define a trust annotated semantic social network.

Definition 13 Let $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$ be a basic typed semantic social network. Let further $t_{\text{trust}} \in \mathcal{T}$ be a distinct tag, and Σ be a complete annotation defined over t_{trust}. A trust annotated semantic social network $\mathcal{SSN}^{\Sigma}$ is defined as the tuple $\mathcal{SSN}^{\Sigma} = (\mathcal{A}, \mathcal{E}, \mathcal{T}, \Sigma, t_{\text{trust}})$.

It should be stated here that this definition is a generalization of a fishnet structure as defined in (Schatten and Žugaj, 2007). In this paper multiple social networks were analyzed where each network was some project on which project members voted for each other. The member with the highest rank was proclaimed project leader. The same holds here if we instead of one trust tag define the set $P = \{t_{\text{trust }1}, t_{\text{trust }2}, \ldots, t_{\text{trust }n}\}$ and interpret the set to be the set of projects or organizational units in some organization. Thus a trust annotated semantic social network can resemble any fishnet structure given a set of trust tags.

6 Emulating a Fishnet Structure

The easiest usage example of the trust annotated social network is the one of a fishnet organization. For instance observe the semantic social network on figure 4. Let $P =$ { project:A, project:B, project:C } be the set of trust tags that correspond to projects A, B and C respectively. Each edge $\alpha_i \xrightarrow{t} \alpha_j$ is interpreted as actor α_i votes for actor α_j on property t. We can divide this network into three separate networks and observe each project individually.

For example the semantic social network of project C, if PageRank is used as the trust level π, could be defined as:

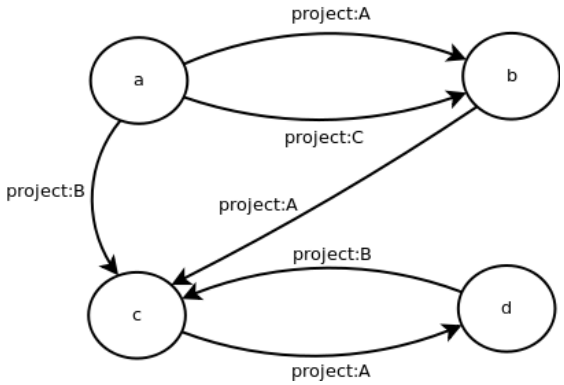

Fig. 4 Trust annotated semantic social network example

$$\begin{aligned}\mathcal{A} &= \{a, b, c, d\}\\ \mathcal{E} &= \{(a, b, \text{project:C})\}\\ \mathcal{T} &= \{\text{project:C}\}\\ \Sigma &\text{ is defined over project:C using PageRank}\\ t_{\text{trust}} &= \text{project:C}\end{aligned}$$

Table 2 summarizes the dynamic hierarchies using the PageRank algorithm on the semantic social network from figure 4. Consequently, actor d is leader of project A, actor c is leader of project B and actor b is leader of project C. A good feature of such a definition is that the hierarchy depends only on the current state of the social network. By changing the social network by, for example, adding a new project, a new hierarchy would be established depending on the votes of members. Thus such a definition takes care of social network dynamics.

Table 2 Rankings of members on projects A, B and C

User	project:A	project:B	project:C
a	0.07	0.07	0.07
b	0.11	0.25	**0.43**
c	0.15	**0.62**	0.25
d	**0.67**	0.06	0.25

7 Knowledge Based Role Management

Role management is of special interest to KM. Consider the following problem: in a dynamic environment roles have to be taken by members of some organizational unit. In order to take a role, a member needs to have certain skills and/or

knowledge. On the other hand other members should support the particular member taking the role. How to find the most adequate person for a given role by considering the social network?

Assume that some role object entitled sales manager has been defined as shown on figure 5.[9] Note that actor d can be any node or number of nodes from a given (super-)network which is why dotted lines were used to represent it.

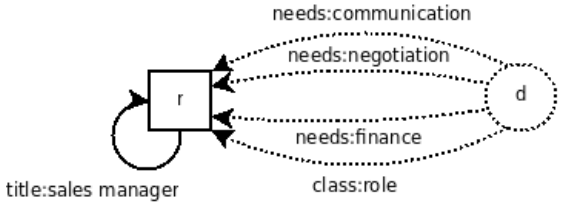

Fig. 5 Role object

The object corresponds to the following F-molecule:

$$\begin{aligned} r &: role\,[\\ title &\rightarrow \text{sales manager}; \\ needs &\twoheadrightarrow \{\text{communication}, \text{negotiation}, \text{finance}\}\,] \end{aligned}$$

Assume further that a semantic social network has been defined as shown on figure 6, from which an actor has to be chosen for the role r.

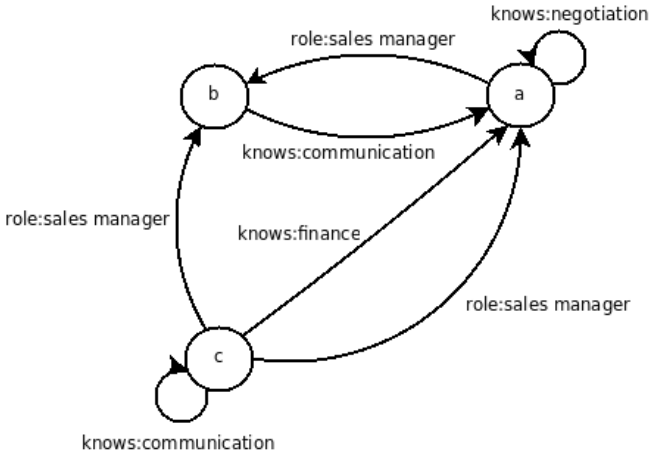

Fig. 6 A semantic social network dealing with user roles and individual knowledge

[9] In ŦAOPĪS not only people can be tagged but various other objects like wiki pages as well. You can imagine the role object as a wiki page (or any other resource) tagged with the adequate attribute-value tags.

Now, before considering any actor we need to make sure that he has all the needed knowledge. This can be done by adding the following deductive rule to the knowledge base ('?'-perpended symbols represent logic variables):

$$\begin{aligned} \text{?actor [candidate for}{\rightarrow}\text{?role]} \longleftarrow\ & \text{?role : role} \\ \wedge\ & \text{?role [needs}{\twoheadrightarrow}\text{?knowledge]} \\ \wedge\ & \text{?actor [knows}{\twoheadrightarrow}\text{?knowledge]} \end{aligned}$$

Using this rule, we can now infer the candidate for relation as shown on figure 7. The inferred relation is denoted with the dot-dash line.

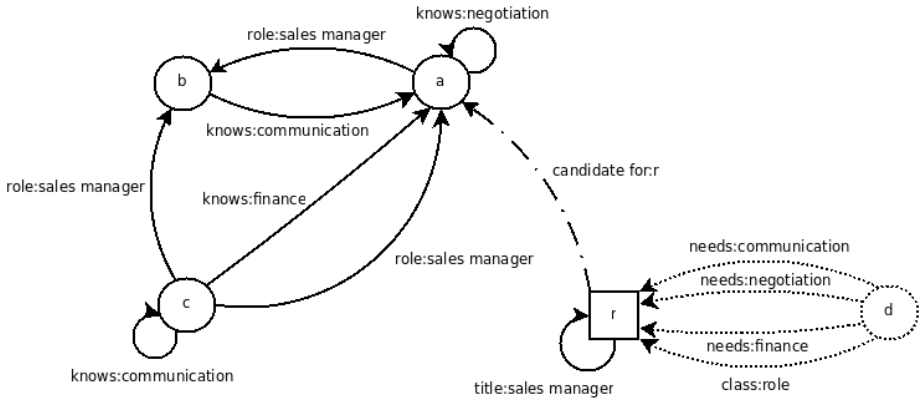

Fig. 7 Inferred relation of candidates

Since only one actor is a candidate for the role, we can conclude that actor a should be the sales manager. If this wasn't the case, further computation had to be done. For example consider the following semantic social network on figure 8.

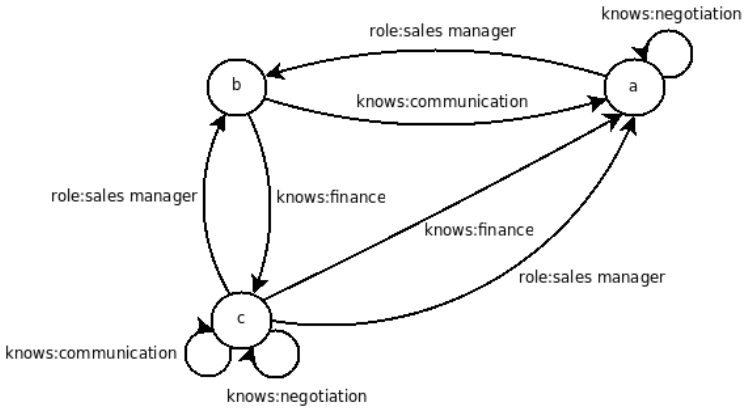

Fig. 8 Semantic social network with more than one candidate for role r

Now we have the case that more than one actor (namely actors a and c) have all the needed knowledge for role r. In order to compute the optimal actor for the given role we shall compute their ranking defined over *role:sales manager* using PageRank. Table 3 summarizes the actors rankings, and shows that actor a has the highest ranking of the candidates, which makes it the optimal solution for role r.

Table 3 Members' ranking computed by using PageRank over *role:sales manager* from the semantic social network on figure 8

Actor	ranking
b	0.81
a	0.12
c	0.07

These examples allow us to formulate the following algorithm: First we query the knowledge base for all candidate actors and put them into S. If there are no candidate actors (no optimal solution) or there is only one, the algorithm stops. If there is more than one solution left we query the knowledge base for the actors from S which have the highest rank. If there is still more than one solution left, we query for those actors from S which have the highest sum of complete annotations Σ for all needed properties. In the end if there are more than one actor in S, all are considered to be optimal.

The algorithm can be extended to find the best suboptimal solutions if the previous returns $S = \varnothing$ as follows: First we query for actors that have the most needed properties and put them into S. If there is only one solution it is suboptimal. If there are more we query the knowledge base for the actor(s) from S with the highest rank. If there are still more solution we query for the actor(s) from S with the highest sum of complete annotations Σ of all its/their needed properties and put it/them into S. All obtained solutions are suboptimal.

8 Knowledge Based Team Management

Another interesting problem that can be solved using semantic social networks is team management. Given a semantic social network and a set of needed knowledge (or any properties) for a given task, find the minimal set of actors that can fulfill it.

Consider for example the semantic social network on figure 9.

Assume that we want to find the minimal team that has programming, system modeling and communication skills. We can formalize the problem as follows: given the semantic social network $\mathcal{SSN} = (\mathcal{A}, \mathcal{E}, \mathcal{T})$, let N_k be the set of needed knowledge. Find the minimal set of actors $\mathcal{A}^{\min} = \{\alpha_i | \alpha_i \in \mathcal{A}\}$ for which it holds that set $E_k = \{k_j | \alpha_i \in \mathcal{A}^{\min} \wedge \alpha_i \text{ [knows}\rightarrow k_j \text{]}\}$ is a superset or equal to N_k.

The following algorithm solves the problem: First we query for all candidate actors α that have at least one needed property and put them into $\mathcal{A}_c$. Then we

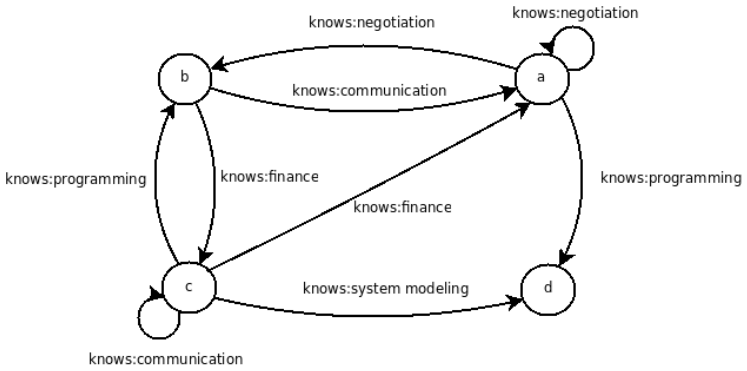

Fig. 9 Semantic social network with potential team members

find the smallest subset(s) of $\mathcal{A}_c$ for which it holds that the union of knowledge sets is a superset or equal to N_k. The resulting set is the set of minimal teams.

By using this algorithm with inputs:

$$
\begin{aligned}
\mathcal{SSN} &= (\mathcal{A}, \mathcal{E}, \mathcal{T}) \\
\mathcal{A} &= \{a, b, c, d\} \\
\mathcal{E} &= \{ \\
&\quad (a, a, \text{knows:negotiation}), \\
&\quad (a, b, \text{knows:negotiation}), \\
&\quad (a, d, \text{knows:programming}), \\
&\quad (b, a, \text{knows:communication}), \\
&\quad (b, c, \text{knows:finance}), \\
&\quad (c, a, \text{knows:finance}), \\
&\quad (c, b, \text{knows:programming}), \\
&\quad (c, c, \text{knows:communication}), \\
&\quad (c, d, \text{knows:system modeling})\} \\
\mathcal{T} &= \{ \\
&\quad \text{knows:communication}, \\
&\quad \text{knows:finance}, \\
&\quad \text{knows:negotiation}, \\
&\quad \text{knows:programming}, \\
&\quad \text{knows:system modeling}\} \\
N_k &= \{ \\
&\quad \text{knows:communication}, \\
&\quad \text{knows:programming}, \\
&\quad \text{knows:system modeling}\}
\end{aligned}
$$

which is the problem defined earlier the query would yield the following results:

$$\mathcal{A}_\mathrm{c} = \{a, b, c, d\}$$
$$\{\mathcal{A}^{\min}\} = \{\{a, d\}, \{c, d\}\}$$

Thus the minimal teams that could handle a given task would be teams consisting of actors $\{a, d\}$ and $\{c, d\}$. If we wanted to find an optimal solution between the minimal teams we could define a trust relation that could give us the ranking of team members and calculate the complete annotation. By summing up the complete annotations of needed knowledge for each particular team we could decide upon the optimal team.

9 Implementation and Usage in KM

Modern organizations have new needs and need to leverage their knowledge faster then ever before. The construction of sophisticated knowledge bases, decision support systems as well as other intelligent systems often takes time (and money) but doesn't yield results as fast as needed.

Using a Web 3.0 system as a self-organizing corporate knowledge base could be constructed by letting employees interact with the system and formalize their knowledge about business. Common Enterprise 2.0 systems already use the lessons learned from Web 2.0, but by introducing semantic technologies such systems could be improved.

Through the use of semantic social networks natural leaders could be identified, and due to the fast information flows new opportunities could be dealt with sooner. Simple querying mechanisms could be developed to facilitate managers with decision support. By connecting such a system to existing databases and information systems through web services as outlined before an integral knowledge management solution could be established that would reflect the current state of the organization and its environment.

Suppose, for example, that some organization consists of five organizational units:

- Marketing
- Sales
- Accounting
- Production
- Human resource

Each department has its own knowledge base in form of a semantic social network. Suppose further that all employees can tag their selves and other employees with their skills. For example, a sales employee could be tagged as:

- `skill : presentation`
- `skill : communication`
- `skill : management`

A manager wants to get an overview of the employees skills. A query in niKlas similar to the following would do the job for him[10]:

[10] Please refer to (Schatten et al, 2009b) or `http://AnonymousWebSite` for an outline of niKlas syntax.

```
[query
  ?dept:organization,
  ?skills = collectset{
    ?_y |
      ?_:person[
        skill->?_y,
        member_of->?dept ]
  },
  sort(?dept,asc). ]
[amalgamate
  "Marketing"
  "Sales"
  "Accounting"
  "Production"
  "Human resource"
]
?dept --> ?skills
[/query]
```

The query would yield a list similar to the following:

```
Marketing --> [communication, presentation, design]
Sales --> [communication, presentation, negotiation]
Accounting --> [finance, excel]
Production --> [database, Linux, presentation, programming]
Human resource --> [communication, databases, management]
```

Or, for example, if a HR manager would like to know if any department hasn't got any communication skills, a query similar to the following could be issued:

```
[query
  ?dept:organization,
  ?_y = collectset{
    ?_y |
      ?_:person[
        skill->?_y,
        member_of->?dept
      ]
  },
  not(
    member(communication,?_y)@_prolog(basics)
  ),
  sort(?dept,asc).
]
[amalgamate
  "Marketing"
  "Sales"
  "Accounting"
  "Production"
  "Human resource"
```

```
]
?dept
[/query]
```

The query would yield a list of departments who are in desperate need for a communication skills seminar.

Suppose further, for example, that employees are tagged with their current projects they work on. The following query could provide a manager with a list of employees that work on more than 4 projects, and need to be sent on vacation:

```
[query
  ?_e:person[
    name->?name,
    surname->?surname
  ],
  ?projects = collectset{
    ?_p |
      ?_e[
        project->?_p
      ]
  },
  ?_count = count{
    ?_x |
      member(?_x,?projects)@_prolog(basics)
  },
  ?_count > 4,
  sort(?surname,asc).
]
[amalgamate
  "Marketing"
  "Sales"
  "Accounting"
  "Production"
  "Human resource"
]
?name ?surname : ?projects
[/query]
```

The result is, as expected, shown on figure 10.

Fig. 10 List of employees working on more than four projects (`TAOPIS`)

We could also easily build a simple agent querying projects on one ore more ᴛAOPīs instances[11]. The following listing presents a simple Python script that downloads a given projects knowledge base.

```
# -*- coding: utf-8 -*-
import urllib
import re
import sys

if len( sys.argv ) > 1:
  url = sys.argv[ 1 ]
  proorg_re = re.compile( r'proorg=(.*)' )

  proorg = proorg_re.findall( url )
  proorg = proorg[ 0 ]

  kb = urllib.urlopen( url )
  lines = kb.readlines()

  kb_f = open( proorg + '.flr', 'w' )

  for i in lines:
    kb_f.write( i )
  kb_f.close()
  kb.close()
  print proorg
else:
  raise Exception,'No url supplied!'
```

Using this script the following predicate could be implemented in *FLORA-2* , loading any knowledge base from an URL.

```
loadKB( ?url ) :-
  str_cat( 'python get_kb.py ', ?url, ?cmd )@_prolog( string ),
  shell_to_list( ?cmd, [ [ ?kb ] ], ?_ )@_prolog( shell ),
  _add(?kb).
```

Now since the knowledge bases are now local, an agent would look similar to:

```
?- _add( loadKB ).

load_projects :-
  loadKB('http://ASystem/x.php?project=Jupiter'),
  loadKB('http://ASystem/x.php?project=Saturn'),
  loadKB('http://ASystem/x.php?project=Neptun').

?- load_projects.

/* Agent definition ... */
```

[11] ᴛAOPīs allows *FLORA-2* and OWL ontology export.

Using such an agent definition a query similar like the following could be issued to find the minimal team that has communication, programming and system modeling skills as described in section 8.

```
?n = [ communication, programming, 'system modeling' ],
?_team = collectset{
  ?_m |
    ?_m:member[ knows->?_k ],
    member(?_k,?n)@_prolog(basics)
},
?m = min{
  ?_len |
    subseq(?_team,?minteam,?_)@_prolog(basics),
    ?knowl = collectset{
      ?_k |
        ?_mm:member[ knows->?_k ],
        member(?_mm,?minteam)@_prolog(basics)
    },
    subseq(?knowl,?n,?_)@_prolog(basics),
    length(?minteam,?_len)@_prolog(basics)
},
subseq(?_team,?minteam,?_)@_prolog(basics),
?knowl = collectset{
  ?_k |
    ?_mm:member[ knows->?_k ],
    member(?_mm,?minteam)@_prolog(basics)
},subseq(?knowl,?n,?_)@_prolog(basics),
length(?minteam,?m)@_prolog(basics).
```

Such a would yield a solution similar to:

```
?n = [communication,programming,system modeling]
?m = 2
?minteam = [d, c]
?knowl = [communication,finance,programming,system ↘
→modeling]

?n = [communication,programming,system modeling]
?m = 2
?minteam = [d, a]
?knowl = [communication,finance,negotiation,programming,↘
→system modeling]

2 solution(s) in 0.0040 seconds

Yes

flora2 ?-
```

Whereby the variable ?n is bound to the needed knowledge, ?m to the size of the minimal team, ?$minteam$ to the list of members of the minimal team and ?$knowl$ is bound the the knowledge the team has.

Some of the presented queries are, off course, to complex to be learned and issued by normal users. This is why an important next step in ŦAOPĪS is the development of easy-to-use querying mechanisms which will hide the complex formalisms in background of the system . Since most common queries deal with simple symbolic or arithmetic constraints we believe that a user interface similar to query-by-example approaches could be established. Another approach might be visual wiki search (Štritof and Schatten, 2010): by visualizing certain parts of the social knowledge base users might be able to select information interesting to them. These selected patterns might then be translated to queries as outlined above.

10 Evaluation and Comparison

In order to provide a framework for comparison to other models, the system was tested on generated benchmark data.[12] To construct random social networks the Watts-Strogatz algorithm was used (Watts and Strogatz, 1998). The sizes of the benchmark networks were 10, 50, 250, 500 and 1000 nodes resembling micro, small, medium sized, big and very big organizations. The average degree of the networks was 2 for the 10-node network, and 10 for the other networks (building on the presumption that every organization member will express their opinion about 10 other people on average). For each test 11 networks were generated: one network of voters for some organizational role, and 10 networks for each knowledge known by some actor. The knowledge networks were distributed in 10% intervals, e.g. knowledge artifact 1 was known by 100% of the actors, knowledge artifact 2 by 90% etc. In this way the queries for role and team management were harder to solve, since some knowledge was less frequently known by actors. Additionally role and project (team) objects were appended to each knowledge base which both required all 10 possible knowledge artifacts (e.g. for an actor to be eligible for the role he needs to know knowledge 1 to 10; for a team to be eligible for the project all 10 knowledge artifacts had to be known by its members).

For each test 10 runs on randomly generated networks were performed by collecting data about actual memory usage and CPU processing time. Each test was comprised of a knowledge base compile, a query for the role candidate, and a query for a minimal team. All tests were performed on an Intel® Core™ 2 Duo CPU T7700 2.40GHz, with 2895 MiB RAM. The test results are presented in table 4.

As one can see from the table, the most memory consuming query (5.05 MiB) is the minimal team query for the 1000 node network. The most CPU time expensive operation is the compile (127.37 seconds) for the 1000 node network. It holds for all queries that the compile time is greater then the querying time. This fact is not too troublesome, since the compile operation is performed only on knowledge base updates. Update operations will only be performed if some actor changes its

[12] The benchmark data as well as data generation and testing tools' source code is available at `http://rapidshare.com/files/445015836/benchmark.tar.gz` (will be made available on a public server of the authors institution)

Table 4 Test results (used memory is expressed in MiB, CPU time in seconds)

	Compile		Role		Team	
Network	Memory	CPU time	Memory	CPU time	Memory	CPU time
10	2.04	2.07	1.80	0.64	1.88	0.64
50	2.07	2.76	1.86	0.65	2.02	0.66
250	2.19	11.67	2.11	0.90	2.62	1.43
500	2.32	33.96	2.42	1.36	3.42	3.83
1000	2.59	127.37	3.05	1.93	5.05	17.32

opinion or learns a new knowledge, and are thus relatively infrequent. If update operations become more frequent (e.g. in highly dynamic organizations) the system is likely to run slow.

On the other hand the minimal team query CPU time for the 1000 node network reflects the fact that deductive systems do not scale well. Since the query consists of multiple subset-related operations, it is expensive regarding time and memory. Thus, the system isn't suitable for very large organizations regarding minimal team queries, due to a likely combinatoric explosion. Figure 11 provides a graphical outline of the obtained results.

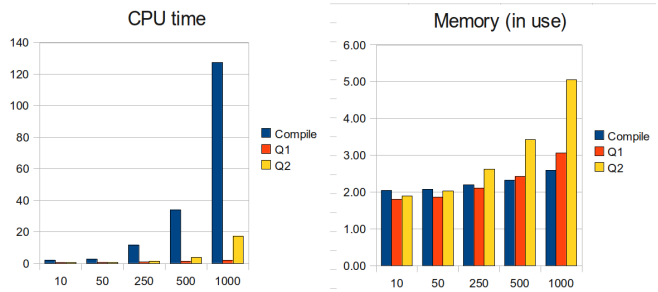

Fig. 11 CPU time and memory consumption of the benchmark tests

To come back to the related models from section 2, the Actor-Concept-Instance model developed by Mika (2007) can be simulated using semantic social networks. Consider the following simple semantic social network depicted on figure 12.

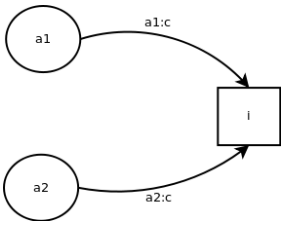

Fig. 12 Actor-Concept-Instance semantic social network

We can define this semantic social network as an Actor-Concept-Instance model with $A = \{a_1, a_2\}$, $C = \{c\}$ and $I = \{i\}$ and $H(T) = (\{a_1, a_2, c, i\}, \{\{a_1, c, i\}, \{a_2, c, i\}\})$.

In order to calculate the co-affiliation matrix we could add the following rule to the dynamic knowledge base:

$$?\text{actor}[\text{ co-affiliated with}\rightarrow ?\text{other actor }] \leftarrow ?\text{instance}[\text{ ?actor}\rightarrow ?\text{concept }] \\ \wedge\ ?\text{instance}[\text{ ?other actor}\rightarrow ?\text{concept }]$$

Now by issuing a query similar a_1[co-affiliated with→?c] we would get all actors which share the same concepts with a_1 (namely a_1 and a_2). In this way adequate rules and queries can be developed for all the other graph folding procedures. Thus, the formalism outlined here can be used to simulate the formalism of Mika, but the reverse relation does not necessarily hold. The Actor-Concept-Instance approach can handle only concepts used on a given instance (describing the domain) and infer social relations based on various situations in the tripartite graph. What it cannot handle are rules (with possible constraints) and customized queries. Thus the approach here is more expressive. On the other hand Mika has developed heuristics to couple with inconsistencies in user supplied data (spelling errors, inadequate descriptors etc.) which the formalism outlined here at the present moment cannot handle even if there were few attempts to solve them (Schatten et al, 2009a, 2010).

The presented formalism can be used to simulate the MetaMatrix model as well. Consider the semantic social network on figure 13. For sake of simplicity, only tasks and actors are depicted.

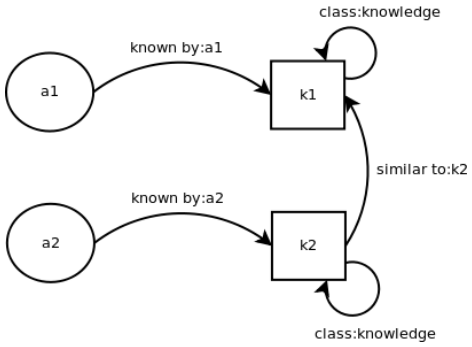

Fig. 13 MetaMatrix semantic social network

In order to compute the KIK' matrix of workers' similar knowledge we could append the following rule to the knowledge base:

$$?\text{person}_1[\text{ knows similar}\rightarrow ?\text{person}_2\text{ }] \leftarrow ?\text{k}_1 : \text{knowledge}[\text{ known by}\rightarrow ?\text{person}_1\text{ }] \\ \wedge\ ?\text{k}_2[\text{ similar to}\rightarrow ?\text{k}_1\text{ }] \\ \wedge\ ?\text{k}_2 : \text{knowledge}[\text{ known by}\rightarrow ?\text{person}_2\text{ }]$$

Now by issuing a query similar to ?person[knows similar→a_2] we would get the persons that have similar knowledge like person a_2 (namely person a_1). In this

way by implementing the necessary rules and queries the MetaMatrix model could be simulated using the presented formalism. We could even go further and add additional attributes to the semantic social network elements (for example we could add an expected learning duration attribute to knowledge instances and then query for only those knowledge artifacts which can be learned in a given time). Thus the presented approach is more expressive then the MetaMatrix model. On the other hand, the MetaMatrix is computationally less expensive then our approach. As with all deductive systems, $\mathcal{F}$LORA-*2* and likewise the presented approach suffers from sensitivity to combinatoric explosion, which is why it isn't suitable for the analysis of extremely large and complex systems.

11 Conclusion and Future Research

The paper presented two types of semantic social networks: (1) basic typed semantic social networks and (2) trust annotated semantic social networks, which are an extension of the former. Both types were implemented into the open-source ŦAOPīs system and leveraged using the niKłaś language for dynamic queries as well as $\mathcal{F}$LORA-*2* for intelligent agent definition. It was shown that this formalism can provide a backdrop for organizing the fishnet structure, a dynamic contemporary organizational form and is a generalization of a previously defined formalism (Schatten and Žugaj, 2007).

The knowledge based role management problem was presented and a solution to the problem was given in form of two algorithms that find optimal solutions, or best suboptimal if no optimal solution exists. Additionally the knowledge based team management problem was presented and solved as well, using a relatively simple algorithm. These problems can be generalized by changing the term 'knows' to any property an object/person can have, and thus represent a class of similar organizational problems that are subject to future research.

Semantic social networks have been put into a knowledge management perspective and it has been shown how they can help organizations to tackle various problems like department skill overview, identification of needed knowledge, identification of workers with to many obligations, or the construction of a minimal team by an intelligent agent. Thus, the envisaged application domain of the model, as put forward herien, are small and medium sized communities (mostly organizations and enterprises) which can make use of the model for their KM processes. On the other hand this application domain is tight to the acutal semantics annotated to given networks. Thus, it might be possible to find addtional application domains not necesarilly dealing with KM or (humane) social networks. An example might be mobile networks in which agents query the network for services and resources. Such a multi-agent domain would subsume that the querying algorithms are distributed, which currently isn't the case and could be subject to future research. Other domains might include smart houses, smart energy grids and other dynamic multi-agent environments.

The formalism was, furthermore compared to existing models and its expressiveness was identified as its main advantage. On the other hand the handling of inconsistent user-supplied metadata as well as computational expensiveness have been pointed out as its main deficiencies. At its current state, the model can handle only the current state of the social network (the valid time). In order to handle

historic states more efforts towards the formalization of a temporal model have to be made. There are, of course, other problems that could be solved using semantic social networks and/or the ТАОРĪS system, and these are subject to future research.

References

Aleman-Meza B, Nagarajan M, Ramakrishnan C, Ding L, Kolari P, Sheth AP, Arpinar IB, Joshi A, Finin T (2006) Semantic analytics on social networks: experiences in addressing the problem of conflict of interest detection. In: Proceedings of the 15th international conference on World Wide Web, pp 407–416

Berners-Lee T, Hendler J, Lassila O (2001) The semantic web. Scientific American Magazine

Bonacich P (1972) Factoring and weighting approaches to clique identification. Journal of Mathematical Sociology (2):113–120

Breslin JG, Decker S (2007) The future of social networks on the internet: The need for semantics. Internet Computing 11(6):86–90

Brickley D, Miller L (2000) Foaf, available at `http://www.foaf-project.org/`, accessed: 20^{th} July 2010.

Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. In: Computer Networks and ISDN Systems, pp 107–117

Cantador I, Castells P (2006) Multilayered semantic social network modeling by ontology-based user profiles clustering: Application to collaborative filtering. In: Managing Knowledge in a World of Networks, Springer, pp 334–349

Carley K (2002) Smart agents and organizations of the future. In: Lievrouw L, Livingstone S (eds) The Handbook of New Media, Sage, Thousand Oaks, CA, pp 206–220

Carley K (2003) Dynamic network analysis. In: Breiger R, Carley K, Pattison P (eds) Summary of the NRC workshop on social network modeling and analysis, Committee on Human Factors, National Research Council, pp 133–145

Downes S (2005) Semantic networks and social networks. The Learning Organization 12(5):411–417

Finin T, Ding L, Zhou L, Joshi A (418-435) Social networking on the semantic web. The Learning Organization 12(5)

Gloor P, Zhao Y (2006) Analyzing actors and their discussion topics by semantic social network analysis. In: Proceedings of the Tenth International Conference on Information Visualization, pp 130–135

Golbeck J, Parsia B, Hendler J (2003) Trust networks on the semantic web. In: Cooperative Information Agents VII, Springer, pp 238–249

Johansen R, Swigart R (2000) Upsizing The Individual In The Downsized Corporation Managing In The Wake Of Reengineering, Globalization, And Overwhelming Technological Change. Perseus Publishing

Jones PM (2001) Collaborative knowledge management, social networks, and organizational learning. In: Smith MJ, Salvendy G (eds) Systems, Social and Internationalization Design Aspects of Human-Computer Interaction, Lawrence Erlbaum Associates, pp 306–309

Jung JJ, Euzenat J (2007) Towards semantic social networks. Lecture Notes In Computer Science 4519:267–280

Kifer M, Lausen G, Wu J (1995) Logical foundations of object-oriented and frame-based languages. Journal of the Association for Computing Machinery 42:741–843

Krackhardt D, Carley KM (1998) A pcans model of structure in organization. In: Proceedings of the 1998 International Symposium on Command and Control Research and Technology, Evidence Based Research, Vienna, VA

Maleković M, Schatten M (2008) Leadership in team based knowledge management - an autopoietic information system's perspective. In: Aurer B, Bača M (eds) 19th Central European Conference on Information and Intelligent Systems – CECIIS2008 Conference Proceedings, Faculty of Organization and Informatics, pp 47–52

Martin J, Odell JJ (1998) Object-Oriented Methods: A Foundation, uml edition edn. Prentice Hall PTR, New Jersey

Mika P (2005) Flink: Semantic web technology for the extraction and analysis of social networks. Web Semantics: Science, Services and Agents on the World Wide Web 3(2-3):211–223

Mika P (2007) Ontologies are us: A unified model of social networks and semantics. Web Semantics: Science, Services and Agents on the World Wide Web 5(1):5–15

Mueller-Prothmann T, Finke I (2004) Selakt - social network analysis as a method for expert localisation and sustainable knowledge transfer. Journal of Universal Computer Science 10(6):691–701

Nietzsche F (1873) Über wahrheit und lüge im außermoralischen sinn, available at `http://www.textlog.de/455.html`, accessed: 21^{st} April 2008.

Nonaka I, Takeuchi H (1995) The Knowledge-Creating Company, How Japanese Companies Create the Dynamics of Innovation. Oxford University Press

Oversity (2004) Citeulike, available at `http://www.citeulike.org`, accessed: 9^{th} October 2010.

Page L, Brin S, Motwani R, Winograd T (1999) The pagerank citation ranking: Bringing order to the web

Reihlen M, Rohde A (2002) Das heterarchische unternehmen: ein flexibiles organisationsmodell im wissensbasierten wettbewerb (the heterarchic business: a flexible organizational model in a knowledge-based competition environment). Zeitschrift Lernende Organisation 8:30–34.

Schachter J (2003) Delicious, available at `http://www.delicious.com`, accessed: 9^{th} October 2010.

Schatten M, Žugaj M (2007) Organizing a fishnet structure. In: 29th International Conference Information Technology Interfaces Proceedings, Cavtat-Dubrovnik, Croatia, pp 81–86

Schatten M, Čubrilo M, Ševa J (2008) A semantic wiki system based on f-logic. In: Aurer B, Bača M (eds) 19th Central European Conference on Information and Intelligent Systems – CECIIS2008 Conference Proceedings, Faculty of Organization and Informatics, pp 57–61

Schatten M, Maleković M, Rabuzin K (2009a) Inconsistencies in semantic social web applications. In: Aurer B, Bača M, Rabuzin K (eds) Proceedings of the 20th Central European Conference on Information and Intelligent Systems, Faculty of Organization and Informatics

Schatten M, Čubrilo M, Ševa J (2009b) Dynamic queries in semantic wiki systems. In: Aurer B, Bača M (eds) Proceedings of the 20th Central European Conference on Information and Intelligent Systems, Faculty of Organization and Informat-

ics, pp 13–20
Schatten M, Kakulapati V, Čubrilo M (2010) Reasoning about social semantic web applications using string similarity and frame logic. In: Aurer B, Bača M, Schatten M (eds) Proceedings of the 21st Central European Conference on Information and Intelligent Systems, Faculty of Organization and Informatics, pp 25–32
Tredinnick L (2006) Web 2.0 and business. Business Information Review 23(4):228–234
Von Kortzfleisch HFO, Mergel I, Proll C (2007) Potentials of social networks for knowledge management with regard to the development of stable competences and dynamic capabilities–conceptualization and case study results. In: Proceedings of the 40th Hawaii International Conference on System Sciences - 2007, IEEE Computer Society, pp 201–209
Štritof T, Schatten M (2010) Visual knowledge discovery on semantic wikis. In: Aurer B, Bača M, Schatten M (eds) Proceedings of the 21st Central European Conference on Information and Intelligent Systems (CECIIS 2010), Faculty of Organization and Informatics, pp 41–45
Žugaj M, Schatten M (2005) Arhitektura suvremenih organizacija. Tonimir and Faculty of Organization and Informatics, Varaždinske Toplice, Croatia
Žugaj M, Schatten M (2007) Otvorena ontologija organizacijske arhitekture u funkciji upravljanja znanjem. Ekonomski vjesnik XX(1 – 2):39–45
Žugaj M, Schatten M (2008) Informacijski sustav za upravljanje znanjem u hipertekst organizaciji. Ekonomski vjesnik 21(1-2):19–30
Žugaj M, Schatten M (2009) Poduzeće 2.0 kao temelj za pramac/krma organizaciju i upravljanje znanjem. Ekonomski vjesnik 22(1):103–114
Wasserman S, Faust K (1994) Social Network Analysis ; Methods and Applications. Structural analysis in the social sciences, Cambridge University Press
Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature (393):440–442
Yang G, Kifer M, Zhao C (2003) Flora-2: A rule-based knowledge representation and inference infrastructure for the semantic web. In: Second International Conference on Ontologies, Databases and Applications of Semantics (ODBASE) Proceedings, Catania, Sicily, Italy