Expert Finding for eCollaboration Using FOAF with RuleML Rules

Li, J.; Boley, Harold; Bhavsar, Virenda; Mei, J.

Proceedings of the 2006 Conference on eTechnologies, 2006

National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de technologie de l'information

NRC-CNRC

Expert Finding for eCollaboration Using FOAF with RuleML Rules *

Li, J., Boley, H., Bhavsar, V.C., and Mei, J.
2006

* Proceedings of the 2006 Conference on eTechnologies. (EDs.) L. Logrippo, H. Mili and A. Salah. Montreal, QC, Canada. pp. 53-65. 2006. NRC 49309.

Canada

Expert Finding for eCollaboration Using FOAF with RuleML Rules

Jie Li[1,2], Harold Boley[1,2], Virendrakumar C. Bhavsar[1], Jing Mei[3]
[1]Faculty of Computer Science, University of New Brunswick
Fredericton, NB, E3B 5A3, Canada
{Jie.Li, bhavsar} AT unb.ca
[2]Institute for Information Technology - e-Business
National Research Council of Canada
{Jie.Li, Harold.Boley} AT nrc.gc.ca
[3]Department of Information Science, Peking University
Beijing 100871, China
mayyam AT is.pku.edu.cn

***Abstract*— Web-based collaboration (eCollaboration) is becoming increasingly popular. The crucial first step of a consulting collaboration is expert finding. This paper describes the FindXpRT project for finding experts via rules and taxonomies. We implemented rules for a client finding an expert to collaborate with, for an expert's decision making on whether to collaborate, and for specifying the collaboration mode. Our expertise taxonomy is taken from the earlier project Teclantic match-making portal. FindXpRT builds on the RDF-based FOAF project, which is attracting increasing attention of researchers as well as practitioners. However, FOAF only provides person-centric facts. We complement these with rules for deriving further FOAF facts, either before (RDF) FOAF publication or, on demand, from published (RuleML) FOAF page: we implemented, in OO jDREW, both a Fact-oriented Normal Form and a Rule-oriented Normal Form.**

***Index Terms*— Social networking, Semantic Web, eCollaboration, RDF, FOAF, RuleML, rules, OO jDREW, expert finding, FindXpRT.**

I. Introduction

The collaboration between people through the Web infrastructure is conceived here as eCollaboration. It facilitates knowledge interchange, processing and publication. Collaboration constitutes a symmetric relationship among people where two or more people are interested in certain knowledge of each other. The crucial first step of collaboration, focused in this paper, is expert finding, where a *client* is aided in finding an *expert* with the expertise he or she requires. While each client-expert consulting is asymmetric, in a working collaboration the client and expert roles are inverted for different subareas of expertise, so that an overall symmetric relationship emerges. Many projects on eCollaboration have been put forward (e.g., [5], [6], [11] and [12]), one of which being our project FindXpRT (Find an eXpert via Rules and Taxonomies), conducted jointly by the National Research Council and the University of New Brunswick. An earlier version of our expert finding technology has been delivered as Teclantic (Technology transfer portal for Atlantic Canada) [13], online at teclantic.ca. Teclantic, focusing on technology transfer in Atlantic Canada, provides its clients with the ability to seek for other projects, or offer projects to be matched with, and found by, other users [13]. Teclantic uses a technology taxonomy for the classification of Atlantic projects [35]. Benefitting from the refined taxonomy of technology provided by Teclantic, employers can find experts who best fit their projects, and a group of experts can collaborate through matching their similar interests.

Social networking plays a significant role in expert finding. Often we require a more general expert to suggest the specialized expert we need by referring to colleagues in his or her social network. Social networking also helps to find an expert by providing a group of people within a community and perhaps links of people outside of the community as well.

Web-based social networking is emerging as a major application area for Semantic Web metadata. "Social networking is built on the idea that there is a determinable structure to how people know each other, whether directly or indirectly" [22]. The well-known notion of "six degrees of separation" [22] means that a person can reach any other person with at most six intermediate personal relationships (because there is a connecting path containing seven nodes with five persons acting as mediators. This supports the idea that people, even directly not knowing each other, can be connected and thus develop a virtual community. This connectivity enables people to both keep in touch with friends and meet new friends. Many companies have been dedicating to developing expert finding through social networks for it improves the match-making expertise among business collaborators.

A number of portals have become popular to support Web-based social networking, including FOAF [20], Friendster [1] and Stumbleupon [3]. In particular, the RDF-based Friend-Of-A-Friend (FOAF) project, originated by Dan Brickley and Libby Miller, has gained momentum in the last few years, and is attracting increasing attention of researchers as well as practitioners.

FOAF allows the expression of personal information and relationships, and permits machine-readable homepages for grouping, categorization and linking of persons. In this sense, FOAF supports online social networks and more importantly, it has the potential to become an important tool for managing social networks. In addition to providing simple directory services, one can use information from FOAF in many ways. For example [23]:

1) Augment e-mail filtering by prioritizing mails from trusted colleagues
2) Provide assistance to new entrants in a community
3) Locate people with interests similar to yours.

The FOAF vocabulary does not, however, capture rule knowledge, and Web Rule Languages such as the Rule Markup Language (RuleML) [18] have only recently been applied to social networking. Rules can, e.g., be employed for dynamically deriving certain FOAF facts on demand. Using the Objected Oriented java Deductive Reasoning Engine for the Web (OO jDREW), one can find people with similar interests more quickly, and RuleML tools enable the XML-based elicitation (VDR-Device [17]), interchange (XSLT [32]), and execution (OO jDREW [15]) of such rules.

Therefore, this paper focuses on applying RuleML to FOAF, specifically on expert finding for eCollaboration using FOAF with RuleML rules. The rest of the paper is organized as follows. In section II, we describe basic concepts, such as expert finding and social networking. In section III, we propose an approach aiming at achieving the goal of this paper: the design of a FOAF rule vocabulary, which is the principal component in the realization; two normal forms, the FNF and RNF, are provided for FindXpRT users with different preference; finally, an XSLT translation from an FNF subset of RuleML to RDF is defined, producing RDF syntax as used by the FOAF community. In section IV we detail the use case focused in this paper: applying RuleML FOAF to expert finding in the domain of computer science and music preparing a possible client-expert collaboration. Sample facts and rules are provided in the human-oriented POSL of RuleML.

II. Basic Concepts

The following describes the main concepts that are relevant to this paper. In subsection II.A, we give a brief introduction to the most important and frequently used concept of this paper, expert finding. We then explain our notion of social networking in subsection II.B, which is the basis of the main topic in the following subsection, II.C: the FOAF vocabulary. Rule languages and engines, serving as the tools to accomplish our goals, are introduced in subsection II.D. Finally, in order to give an explicit view of profiles exemplifying RulML FOAF, in subsection II.E we provide a description of rules extending FOAF profiles for eCollaboration.

A. Expert finding

In the era of rapidly developing technology and economics, there emerge many multi-regional corporations. However, these large organizations often suffer from the fact that people in the same organization do not know each other well or what other members' expertise is. Effective collaboration among employees is thus hampered.

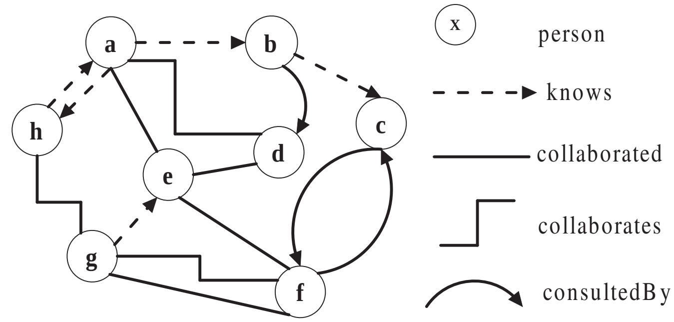

Fig. 1. A social network with 'knows', 'collaborates' (current collaborative relationship), 'collaborated' (former collaborative relationship) and 'consultedBy' links.

Portals such as ExpertWitness [9], Expertise Search [8] and Teclantic [13], providing services for finding an expert or an expertise-developing project, have become popular recently, as they provide an appropriate way to help solving the above problem. These portals enable users to query for a specific expertise and provide them with a list of experts, who have the relevant expertise, through match-making.

However, it is quite common that novices end users have difficulty in characterizing their request of specific expertise, and current systems are not user-friendly enough to help users to find an expert and collaborate with him or her if desired.

Therefore, a main application of our model is expert finding, where a taxonomy of expertise is needed. The technology taxonomy from Teclantic and its classification data are the main sources for the expertise taxonomy utilized in this paper. The expertise taxonomy can be implemented in RDFS [33], providing the order-sorted type system for OO jDREW [15]. With the help of the expertise taxonomy and rule specification, an expert finder can help people inside or outside a company to find an appropriate expert with whom they can collaborate and even provide 'proxy' suggestions, thus supporting the collaboration between people.

B. Social Networking

To support social networking it is helpful to represent various properties of, and relationships between, persons expressing a wide range of self-description and social connectivity. Persons who participate in such a networked (sub)society may be friends, relatives, work collaborators, employees, and so on. A diagram that illustrates a schematic example of a social network with four kinds of links is depicted in Fig. 1.

C. FOAF Vocabulary

FOAF (Friend Of A Friend) permits detailed description of profiles of persons and the relationships between them using a machine-readable syntax. It is realized as a Semantic Web vocabulary serialized in RDF/XML, from which browsable HTML can be generated.

Since RDF cannot express rules, the current FOAF project contains only facts so that rule-based deduction is not usually done in FOAF. However, rules have significant use in the Semantic Web, which can be brought to bear to FOAF rules. For example, profile 'facts' can be made conditional on the situation, context, time and/or location, like the time preference of and distance from an access.

D. Rule Languages and Engines

Here we give an overview of the tools relevant to this paper, including rule languages such as RuleML and POSL [19], rule engines such as OO jDREW, as well as the translating environments such as XSLT.

RuleML (Rule Markup Language) is a markup language that allows to publish and share rulebases on the World Wide Web. It is an evolving logic language with a current version 0.9. Object-Oriented RuleML (OO RuleML)[1] is an extension of RuleML that has developed from a "slotted" sublanguage of RuleML.

POSL (Positional-Slotted Language) is a human-readable format for Semantic Web Knowledge that combines Prolog's positional and F-logic's slotted syntaxes for representing knowledge (facts and rules) in the Semantic Web [19]. Moreover, it is interconvertible with RuleML.

Mandarax [2], SweetRules [4] and jDREW [34] have been developed as rule engines supporting various subset of RuleML. OO jDREW, an object oriented extension of jDREW, is a reasoning engine for executing RuleML rule markup. There are two kinds of deduction in OO jDREW: OO jDREW BU (Bottom Up) and OO jDREW TD (Top Down). In OO jDREW BU, rules are used to derive new facts from given facts until a fix point is reached. In OO jDREW TD, rules are used to answer queries by reducing them to subqueries until facts are reached.

XSLT, which can help transform one XML file to another, is used here to map RuleML facts back to RDF facts via XML Spy.

E. Rules Extending FOAF Profiles for Social Networking

Rules added to FOAF can extend a person's profile facts to make implicit properties and relationships with other persons explicit. For example,

```
IF A collaborated with B
THEN B collaborated with A
```

This symmetry of the 'collaborated' relation and the transitive 'knows' closure, 'knowsTrans', can help group together persons for consultation. For example,

```
IF A knows B THEN A knowsTrans B
IF A knows B AND B knowsTrans C
THEN A knowsTrans C
```

RuleML markup can also constitute properties conditional on other persons, the time, the location and so on. A time-dependent rule set such as for preferred phones to call a FOAF page owner for different time intervals can be expressed as follows. For normal workdays a person Peter Pan could specify the following rule set:

```
phonePreference(Peter_Pan,office) :-
               time(9-12) OR time(13-17).
phonePreference(Peter_Pan,cell) :-
               time(12-13) OR time(17-18).
phonePreference(Peter_Pan,home) :-
               time(18-21).
phonePreference(Peter_Pan,voicemail) :-
               time(21-9).
```

Such rulebases can alert both humans and machines about the preferences of homepage owners depending on their person-centric metadata.

III. Extended FOAF Vocabulary and Translation

In this paper we propose an approach to applying the RuleML language to metadata constituting FOAF profiles for expert finding. This section deals with the main parts of realizing our proposed approach and is divided into the following seven subsections.

The first two subsections provide us with the basis of the realization. In subsection III.A, we introduce the programming language we are using for implementation. We explain that POSL is used as a human-oriented language in our research on FindXpRT. We then describe the structure of the facts in our knowledge base in subsection III.B. In subsection III.C, we propose the major component of the approach, the design of a FOAF rule vocabulary in the domain of expert finding. Subsection III.D then describes the classification of facts and rules according to this rule vocabulary. After the design of a rule vocabulary, the fact-oriented and rule-oriented normal forms, for different uses of FindXpRT, are put forward in subsection III.E. In subsection III.F, we introduce rule execution for expert finding, i.e., computing derived FOAF properties with OO jDREW. Finally, the XSLT translation of RuleML facts to RDF, for the RDF publication of FNF facts is presented in subsection III.G. Scenarios of rule-extended FOAF profiles for expert finding will be given in Section IV.

A. POSL as an Human-Oriented Language

The declarative language POSL used in this paper for facts and rules is directly correspondant to mathematical specifications, so that their verification becomes easier. We use POSL as a human-oriented language, obtain the RuleML syntax of the FOAF document automatically. Two steps are thus involved:

- Development and maintenance of the knowledge base in the compact form of POSL.
- Translation of a POSL syntax into RuleML through the bidirectional RuleML ↔ POSL converter[2].

There are several advantages of this approach. First, with POSL, we are able to represent the knowledge base compactly and hence it is more human readable than RuleML/XML. Second, RuleML use of the XML syntax, permits XSLT translation to other XML formats, such as RDF. Third, POSL-to-RuleML translation can be easily done through the existing converter.

[1]http://www.ruleml.org/indoo/indoo.html

[2]http://www.jdrew.org/oojdrew/demo/translator.jnlp

B. Structure of Facts in Knowledge Base

We store the information of each expert as facts in our knowledge base. All of these facts are expressed by a tree data structure. The structures of these facts need not be uniform, i.e., we do not assume a fixed schema.

As illustrated in our scenario of a fictitious expert Peter Pan (see Fig 3), the information about the expert normally includes his or her expertise, publications, working location, telephone, email, and so on. When expressing the information of Peter Pan, we first take advantage of the current FOAF vocabulary, and only if the existing FOAF vocabulary cannot satisfy our needs, we propose some new vocabulary.

The detailed information on experts, down to the leaf nodes of these trees, can be extracted through queries using a top-down engine, e.g., OO jDREW TD [15].

C. Designing a FOAF Rule Vocabulary for Expert Finding

One purpose of implementing rules is to perform certain actions, such as a client querying a certain expert with several conditions; a client finding a specialized expert via consulting a more general expert; an expert making decisions on whether to collaborate according to his or her own criteria; designing different collaboration modes according to the convenience of people, the location, the time zone and so on, as can be developed into rule-based intelligent assistants ('proxies').

Rules are also able to 'harvest' metadata from other homepages, accumulating and filtering the results. An important special case is computing those subsets of the transitive closure of the foaf:knows property (relation) that contain one or more given persons. Like in the HTML Web, distributed RuleML FOAF homepages can permit everyone to copy and edit from other persons' published rulebases, and agents will be able to apply the rules. This transitive relationship can be applied to an expert's social networks so that he is able to recommend a specialized expert when a client consults with him.

Therefore, in this paper we specify a rule vocabulary augmenting the current fact-only FOAF vocabulary. Once rules are realized, various mathematical methods, such as graph-theoretic, algebraic, and logic, can be applied to RuleML FOAF. Moreover, we can provide foundations for making FOAF tools rule-aware by coupling them with rule engines such as OO jDREW for specifying and executing FOAF rules.

We have developed the current RDF FOAF vocabulary for both elementary and rule-derivable facts. Particularly, as we have done in the previous use cases, we extend the existing vocabulary according to the needs of the expert finder, such as seeksExpertise(Peter_Pan,ComputerScience) and publicationIn(Peter_Pan,IEEE).

We have also designed a FOAF rule specification, which does not exist in the current FOAF vocabulary. After the specification, rules can be implemented in RuleML FOAF. Principles for the FOAF rule vocabulary have been developed, e.g., relations should use the person as subject in the first argument position.

We have designed the rule vocabulary specification for music and computer science domain. In this paper we focus designing the rule specification on expert finding. For example, ?Person1 can seek advice of ?Person2 if ?Person1 seeks an expertise that ?Person2 offers.

```
seeksAdvice(?Person1,?Person2) :-
   seeksExpertise(?Person1,?X),
   offersExpertise(?Person2,?X).
```

D. Classification of Facts and Rules

We distinguish two different kinds of facts: elementary facts (properties) and rule-derivable facts (properties). We also distinguish different relation categories among people by identifying different relationships, namely 'knows', 'collaborates', 'collaborated' and 'consultedBy'. People's identifications, such as expert and specializedExpert, are distinguished as well. Different degrees of availability of people are also differentiated, such as atWork and onHoliday. Since the FOAF vocabulary only provides us with properties such as 'knows', we make use of our extended vocabularies, such as 'seeksAdvice' and 'atWork'.

Within the rule-derivable properties, we also distinguish two different kinds of properties[3]:

1) Properties that can be generated by taxonomic derivations (e.g., using RDF's subClassOf):

```
expert(?Person) :-
       rocketScientist(?Person).
```

2) Properties generated by general derivations:

```
expertise(?Person,?Area) :-
   publication(?Person,?Area,?Amount),
   greaterThan(?Amount,3).
```

We also extend rules to the FOAF vocabulary. Two categories of rules are involved here:

1) Rules describing a single person:

```
atWork(?Person,?Time) :-
    inInterval(?Time, 9, 17).
```

2) Rules describing information about two or more persons:

```
seeksAdvice(?Person1,?Person2) :-
   seeksExpertise(?Person1,?X),
   offersExpertise(?Person2,?X).
```

E. The Fact-oriented and Rule-oriented Normal Forms

Facts in the RuleML FOAF vocabulary using a subset of RuleML can be easily mapped back to RDF facts via an XSLT translator when necessary. In cases where only facts are needed in FOAF RuleML, their FOAF RDF form can be automatically generated using the XSLT stylesheet. Rules enriched for deriving new facts can infer, e.g., reflexive, symmetric, and transitive relations.

Two normal forms for RuleML FOAF rulebases are proposed for our FindXpRT: a Rule-oriented Normal Form (RNF) and a Fact-oriented Normal Form (FNF). While the RNF is more compact, the FNF directly corresponds to RDF FOAF facts.

[3]Variables are specified with a symbol '?'.

RNF: The RNF includes rules as well as the (elementary) facts that are needed by the premises of the rules. Those facts that are derivable from the rules by a bottom-up engine such as OO jDREW BU are removed from the rulebase. (Complementarily, corresponding queries could be posed to a top-down engine such as OO jDREW TD.)

FNF: The FNF includes elementary facts and derived facts. Rules are removed from the published rulebase after all possible facts are derived by running a bottom-up engine such as OO jDREW BU. (When new elementary facts are asserted, the rules need to be re-applied to them.)

A complete example of a non-normalized rulebase is given below:

```
(rule-1)
 expertise(?Person,?Area) :-
    rating(?Person,?Area,?Score),
    greaterThan(?Score,4),
    workDuration(?Person,?Area,?Year),
    greaterThanOrEqual(?Year,2).

(rule-2)
 expertise(?Person,?Area) :-
    publication(?Person,?Area,?Amount),
    greaterThan(?Amount,3:Integer).

(rule-3)
 expertise(?Person,?Area) :-
    RecordedCDs(?Person,?Area,?Amount),
    greaterThan(?Amount,6).


 (fact-0) expertise(Bill,AI).
 (fact-1) expertise(Peter_Pan,AI).
 (fact-2) rating(Peter_Pan,AI,5).
 (fact-3) workDuration(Peter_Pan,2).
 (fact-4) publication(Peter_Pan,AI,4).
 (fact-5) RecordedCDs(Lucy_Alm,Pop,6).
```

For transforming this to RNF, (fact-1) can be removed from the rulebase because it can be derived from either (rule-1) or (rule-2) .

Likewise, for transforming this to FNF, all these rules are removed from the published rulebase after a new fact expertise(Lucy_Alm,Pop) has been derived by running OO jDREW BU with (rule-1) and (rule-3).

F. Computing Derived FOAF Properties for Expert Finding

Computing derived FOAF properties involves two steps. First, we merge rules of different persons and eliminate duplicate facts and/or rules, if any, in the rulebases. Then we run the merged rulebases in OO jDREW to get the parsed rulebases with new facts.

We execute our knowledge base in OO jDREW BU and derive new facts which can be later added to the FNF. We then update the RNF by removing the derivable facts.

We query our knowledge base in OO jDREW TD to get the desired information for finding expert.

G. XSLT Translation of RuleML Facts to RDF

The knowledge base in RuleML syntax can be translated to RDF on demand via our existing XSLT translator. Obviously, only RuleML facts, not rules, obtained from the previous procedure can be mapped back to RDF syntax. Since FOAF pages are usually written in RDF syntax, it is important to enable RDF as the delivery format when there are no rules, as can be realized by applying FNF.

IV. Scenario of Expert Finding

We give a scenario here of FindXpRT's RuleML FOAF facts and rules, which includes information about fictitious persons, and rules that identify relationships between persons and preferred actions among persons. Inspired by the Robot Composer [10], where computer programs compose music using techniques from artificial intelligence (e.g., neural networks and 'genetic algorithms' [10]), we present a scenario of establishing a collaboration between an AI expert and a Pop musician. In subsection IV.A, we illustrates two sample facts in RuleML FOAF. Then we represents three kind of RuleML FOAF sample rules in subsection IV.B.

A. RuleML FOAF Sample Facts

Graphical FOAF tree representations about two fictitious persons are shown in Fig. 2 and Fig. 3 (without namespace prefixes).

The symbolic version for Fig. 3 in POSL can be written in the following way. Namespaces are represented as prefixes before a '.' symbol[4] for the purpose of implementation. The vocabularies with a 'foaf' namespace prefix indicates that they are borrowed from the existing FOAF vocabulary specification [21], while those with 'ex' prefixes are the vocabularies extended by, for the use in expert finding.

Profile of Lucy Alm

```
foaf.person(Lucy_Alm[
 foaf.membershipClass->Group[
  foaf.name->BestGroup;
  ex.location->Fredericton];
 ex.phone->Tel[ex.cell->0523];
 ex.expertise->Category[
  foaf.name->Pop[
   ex.rating->4.5;
   foaf.name->RecordedCDs[
    ex.amount->10];
   ex.workDuration->3]];
 foaf.mbox->"lalm@best.com";
 foaf.knows->Person[
  foaf.name->Eric;
  foaf.name->Hope]]).
```

[4]This is because the symbol ':', commonly used to express namespaces, is reserved in OO jDREW as a type infix for separating terms from their order-sorted types.

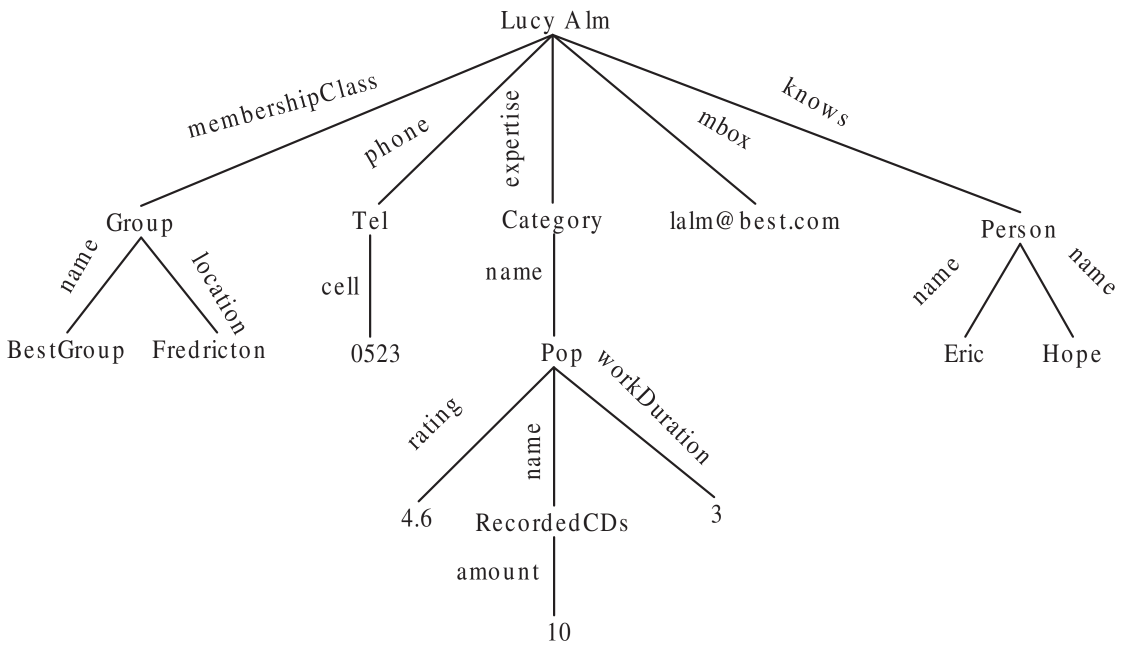

Fig. 2. Lucy Alm (fictitious person).

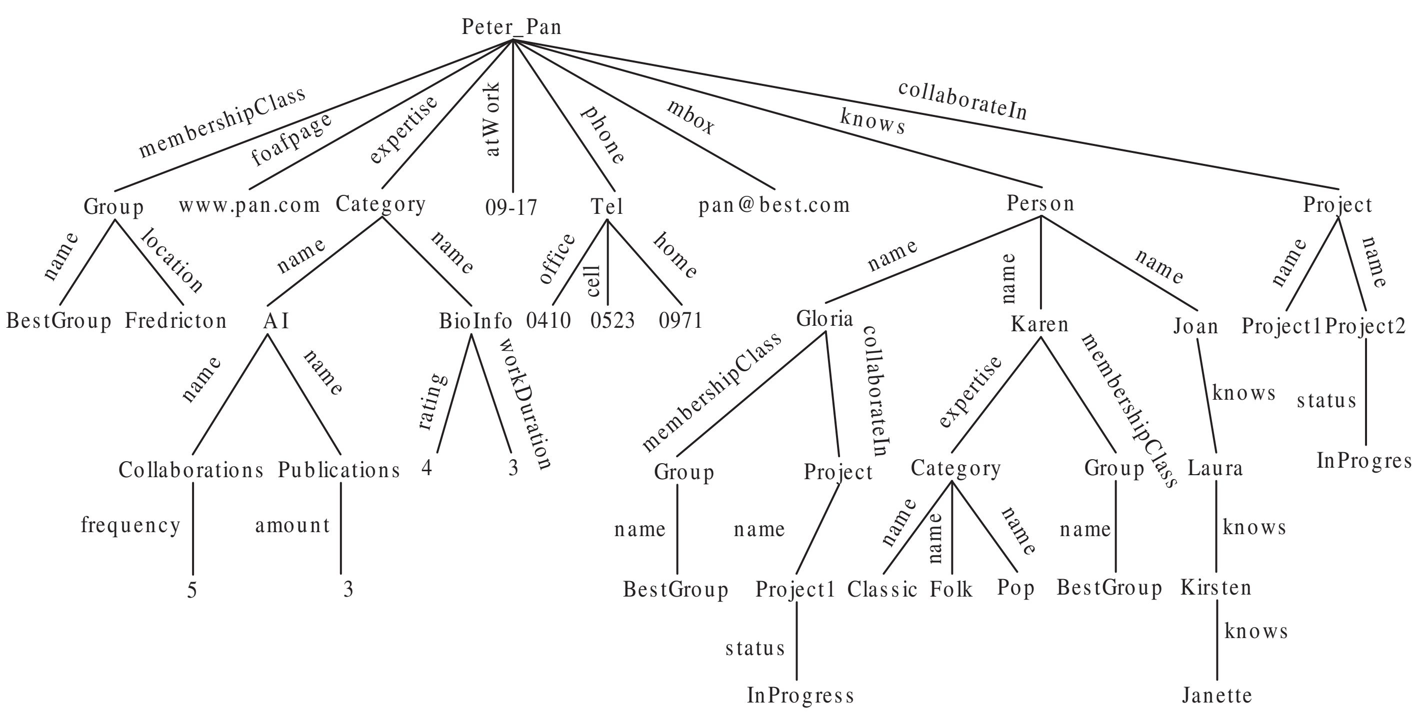

Fig. 3. Peter Pan (fictitious person).

Profile of Peter Pan

```
person(Peter_Pan[
 foaf.membershipClass->Group[
             foaf.name->BestGroup;
             ex.location->Fredericton];
 foaf.foafpage->"www.pan.com";
 ex.expertise->Category[
   foaf.name->AI[
     foaf.name->Collaboration[
      ex.frequency->5];
     foaf.name->Publication[
      ex.amount->3]];
   foaf.name->BioInfo[
    ex.rating->4;
    ex.workDuration->3]];
 ex.atWork->"09:00-17:00";
 ex.phone->Tel[
            ex.office->0410;
            ex.cell->0523;
            ex.home->0971];
 foaf.mbox->"pan@best.com";
 foaf.knows->Person[
   foaf.name->
    Gloria[
     foaf.membershipClass->Group[
      foaf.name->b]BestGroup];
     ex.collaborateIn->Project[
      foaf.name->Project1[
        ex.status->InProgress]];
   foaf.name->
    Karen[
     ex.expertise->Category[
       foaf.name->Classic;
       foaf.name->Folk;
       foaf.name->Pop];
     foaf.membershipClass->Group[
       foaf.name->BestGroup]];
   foaf.name->
    Joan[
     foaf.knows->Laura[
      foaf.knows->Kirsten]]];
 ex.collaborateIn->Project[
   foaf.name->Project1;
   foaf.name->Project2[
    ex.status->InProgress]]]).
```

B. *FindXpRT's Top-Level Rule Systems*

We provide rules systems to illustrate FindXpRT's method of expert finding for eCollaboration. We first represent rules for finding potential experts to collaborate with. Then we provide rules for "the selected" expert to make decisions on the collaboration. Next, we introduce the rules for preference to collaboration mode.

Fig. 4 and Fig. 5 are first given to illustrate two sample rule systems, which are followed by the symbolic versions of the same rule systems.

1) Rule system for expert finding: This rule system is client-centric and is used by clients to find an expert to collaborate with themselves. The flowchart showing these rules are shown in Fig. 4. The expertise ?X and ?Y represented in this paper ranges over the taxonomy for technology transfer of Tecalantic.ca.

When a client queries for an expert in an area, the rule system accesses the experts' profiles. It first checks, in the profile a candidate expert, if he/she meets the qualification of offering the required expertise and if he/she is a person different from the client. In Fig. 4, we focus on persons in that same cooperation (group). If the client and the expert have collaborated on the same project, which is still in progress, the process ends because they are already collaborating. Otherwise, if the expert's offered expertise does not match the client's sought expertise according to our taxonomic similarity measure [35] for a user threshold ?T, FindXpRT cannot pair them up for consultation. Otherwise, when this expert is not currently involved in any project, the FindXpRT calls another rule system as a subroutine, namely CollaborationDecision. If the result of the CollaborationDecision rule system is 'Accept', then the FindXpRT answers the client by providing this expert. However, when the expert turns out to be unavailable, or the result of the CollaborationDecision rule system is not 'Accept', then this expert may still refer the client to other potential experts in his/her social network[5]. According to the "six degrees" concept [22] discussed in section I, there would be at most six such rounds of recommendation. The variable ***degree*** in Fig. 4 is thus assigned the value ***6***.

The symbolic version of Fig. 4 is shown here[6]:

Input: client ?C, initial expert ?E, expertise ?X and ?Y, pre-assigned value 6 for degree (cf. subsection II.E).

Output: Assert consultedBy(?C,?E) when finding an appropriate ?E.

Step 1:

```
IF expertise(?E,?X)
{
  IF notEqual(?C,?E)
  {
      GOTO Step 2
  }
}
```

Step 2:

```
IF (group(?C,?Group), group(?E,?Group))
{
    GOTO Step 3
}
ELSE
{
   Process as Global Expert Finding
}
```

Step 3:

```
IF (collaborateIn(?C,?Project1),
    collaborateIn(?E,?Project1))
{
    GOTO Step 4
}
ELSE GOTO Step 5
```

[5]For simplicity (avoiding non-determinism on this level), we assume that each expert can refer at most to a single other expert.

[6]Negation as failure is written as naf(...).

FindXpRT

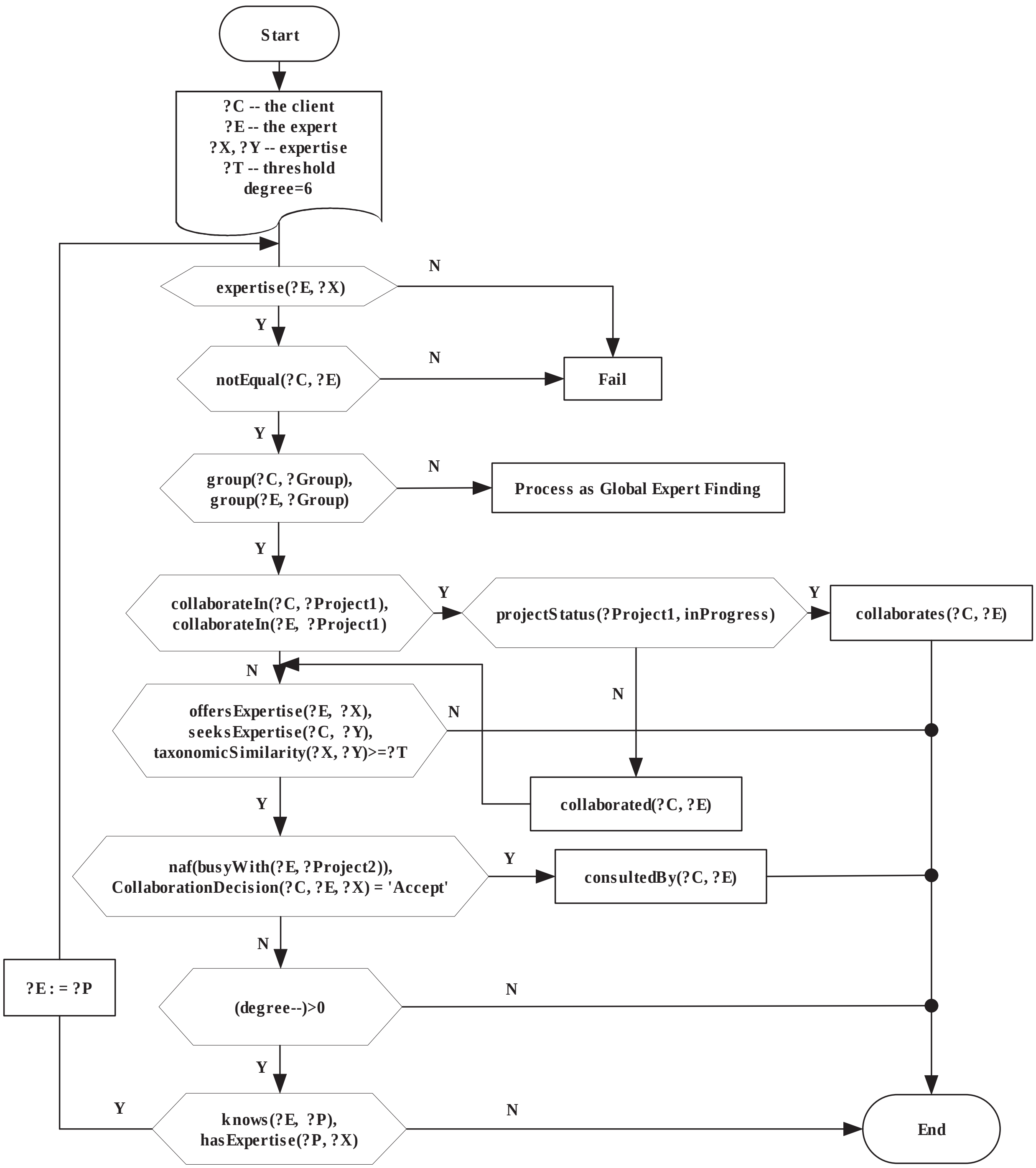

Fig. 4. Scenario of Expert Finding.

Step 4:

```
IF projectStatus(?Project1,InProgress)
{
   collaborates(?C,?E)
}
ELSE
{
   collaborated(?C,?E)
   GOTO Step 5
}
```

Step 5:

```
IF (offersExpertise(?E,?X),
    seeksExpertise(?C,?Y),
    taxonomicSimilarity(?X,?Y)>=?T)

{
   GOTO Step 6
}
```

Step 6:

```
IF (naf(busyWith(?E,?Project2)),
    CollaborationDecision(?C,?E,?X)
    ='Accept')
{
   Assert consultedBy(?C,?E)
}
ELSE GOTO Step 7
```

Step 7:

```
loop:
   while (degree--)>0
     IF (knows(?E,?P),
         offersExpertise(?P,?X))
     {
           ?E := ?P
           GOTO Step 1
     }
```

Pseudo-Code 1: Pseudo-Code for Fig. 4

We give here a match-making example on the basis of the two profiles, described in subsection IV.A, of persons Lucy Alm and Peter Pan. Suppose Lucy Alm is the client while Peter Pan is the expert. Lucy Alm, as a client, inputs AI as the required expertise, with a similarity threshold 0.8, from an expert. Peter Pan first satisfies the criteria that he has the expertise. Then he meets the requirement that he is from the same company, namely BestGroup, as Lucy Alm. The next rule checks if they are not collaborating with each other, succeeding in our case. Peter Pan then meets the condition that he offers AI as his expertise, with a taxonomic similarity of 1.0, greater or equal 0.8, as Lucy Alm required. The next step is to see if Peter Pan is currently busy with some project. Since Peter is not having any project at this time, the rule system calls another rule system CollaborationDecision, illustrated in the following subsection 2). When CollaborationDecision gives the result, in our case 'Accept', a new fact, consultedBy(Lucy_Alm, Peter_Pan), is asserted.

2) Rule system for decision making on collaboration: Rules in this rule system is expert-centric, and helps an expert who is "selected" to collaborate with an arbitrary person to make decisions on participation. Different persons can have different precondition for making a decision. Therefore, in Fig. 5, the previously open expert, expressed by a variable ?E, is here a constant, Peter_Pan, these rules are local and attached to this specific person.

Fig. 5 illustrates the scenario of how Peter Pan makes a decision on request by an arbitrary person for participating in a project. The rule system first gets the preferred phone number via the phonePreference rule set (cf. subsection II.E). When Peter Pan receives this request, he accesses this client, e.g. Lucy Alm's profile. He first checks if this person is from the BestGroup cooperation within the Fredericton region, as he declines all the request outside his company's Fredericton branch. Moreover, Peter Pan is only interested in collaborating with clients in Pop music. Peter Pan has criteria on the number of a collaborator's RecordedCDs, period of working and rating (ranked by colleagues with 5 as the best mark). Peter Pan only decides to collaborate when the client meets all of these criteria. After making his decision, Peter Pan contacts people in two ways, by phone or email, depending on this person being within his social network.

The symbolic version of Fig. 5 for a specific person Peter Pan is shown below:

Input: client ?C, expertise ?X.

Output: Accept or Decline the request.

Step 1:

```
phonePreference(Peter_Pan,?Tel1),
call(?C, ?Tel1)
```

Step 2:

```
IF location(?C,Fredericton)
{
   GOTO Step 3
}
ELSE
{
   Decline the request
}
```

Step 3:

```
IF offersExpertise(?C,?X)
{
   GOTO Step 4
}
ELSE
{
   Decline the request
}
```

Step 4:

```
IF (RecordedCDs(?C,?Amount),
    greaterThan(?Amount,8),
    workDuration(?C,?Year),
    greaterThan(?Year,2.0))
{
   GOTO Step 5
}
ELSE
{
   Decline the request
}
```

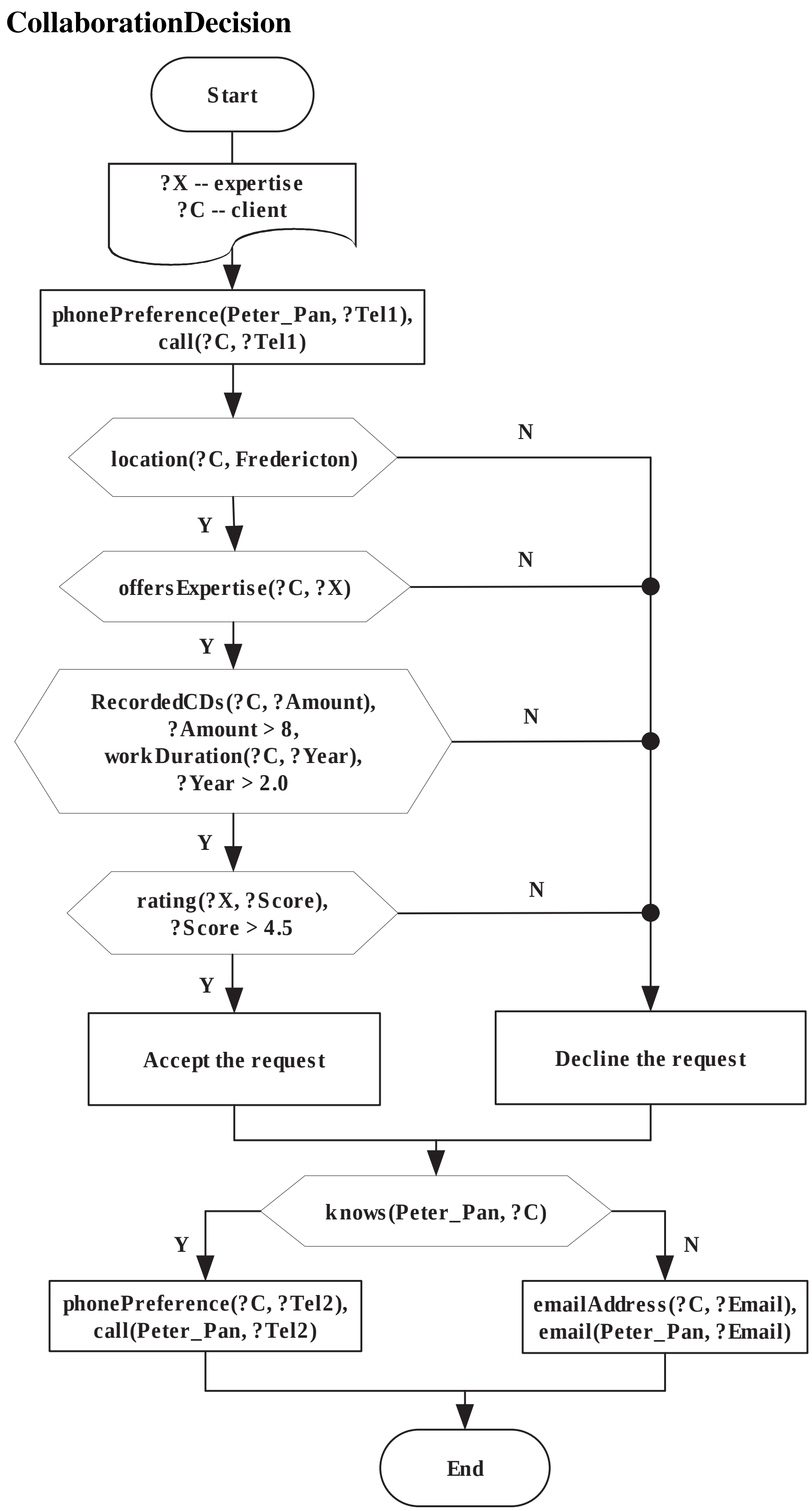

Fig. 5. Scenario of Decision Making on Possible Collaboration.

Step 5:

```
IF rating(?X,?Score) AND
   greaterThan(?Score,4.5)
{
   Accept the request
   GOTO Step 6
}
ELSE
{
   Decline the request
}
```

Step 6:

```
IF knows(Peter_Pan,?C)
{
   phonePreference(?C,?Tel2),
   call(Peter_Pan,?Tel2)
}
ELSE
{
   emailAddress(?C,?Email),
   email(Peter_Pan, ?Email)
}
```

Pseudo-Code 2: Pseudo-Code for Fig. 5

As described previously, in 1), Peter Pan is chosen by Lucy Alm as desired collaborator and it calls CollaborationDecision. Peter Pan receives the message in his preferred way providing by the rule set phonePreference. This rule system is for Peter Pan to decide whether he wants to accept this request. Peter Pan wants his prospective collaborator to be in Fredericton, where Lucy Alm satisfies. Lucy Alm also satisfies his criteria as she not only offers expertise in Pop music, but also her RecordedCDs exceed eight, and she has a three-year working experience, which exceeds the two-year's minimum, a 4.5 rating which also exceeds 4 as the minimum. Therefore, Peter Pan accepts the request. The means for him accepting the request is via e-mail since Peter Pan does not know Lucy Alm.

3) Rules for specifying the collaboration mode: This rule system represents preferences of clients regarding the collaboration mode. With these rules, a client can collaborate face to face with an expert, or by telephone, or through the Web, according to restrictions on the date, time and distance. We represent these rules first in POSL and then also serialize the first two rules in RuleML[7].

```
(POSL-1)
collaborationMode(F2F,?Date,?Time,?Distance):-
    holiday(?Date),
    greaterThanOrEqual(?Time,10:Integer),
    lessThanOrEqual(?Time,16:Integer),
    lessThan(?Distance,20:Integer).
(POSL-2)
collaborationMode(F2F,?Date,?Time,?Distance):-
    naf(holiday(?Date)),
    greaterThanOrEqual(?Time,16:Integer),
    lessThanOrEqual(?Time,20:Integer),
    lessThan(?Distance,20:Integer).
(POSL-3)
collaborationMode(Tel,?Date,?Time,?Distance):-
    holiday(?Date),
    greaterThanOrEqual(?Time,09:Integer),
    lessThanOrEqual(?Time,22:Integer),
    greaterThanOrEqual(?Distance,20:Integer),
    lessThan(?Distance,100:Integer).
(POSL-4)
collaborationMode(Tel,?Date,?Time,?Distance):-
    naf(holiday(?Date)),
    greaterThanOrEqual(?Time,17:Integer),
    lessThanOrEqual(?Time,22:Integer),
    greaterThanOrEqual(?Distance,20:Integer),
    lessThan(?Distance,100:Integer).
(POSL-5)
collaborationMode(Web,?Date,?Time,?Distance):-
    holiday(?Date),
    greaterThanOrEqual(?Time,09:Integer),
    lessThanOrEqual(?Time,22:Integer),
    greaterThanOrEqual(?Distance,100:Integer).
(POSL-6)
collaborationMode(Web,?Date,?Time,?Distance):-
    naf(holiday(?Date)),
    greaterThanOrEqual(?Time,17:Integer),
    lessThanOrEqual(?Time,22:Integer),
    greaterThanOrEqual(?Distance,100:Integer).
```

Rule (POSL-1) expresses that if it is a holiday, the time is between 10:00 and 16:00 o'clock, and the distance to the collaboration place is less than 20 miles, then the preferred collaboration mode is face to face.

Rule (POSL-2) represents that if it is not a holiday, the time is between 16:00 and 20:00, and the distance to the collaboration place is less than 20 miles, then the preferred collaboration mode is also face to face.

Rule (POSL-3) expresses that if it is a holiday, the time is between 09:00 and 22:00, and the distance to the collaboration place is between 20 miles and 100 miles, then the preferred collaboration mode is by telephone.

Rule (POSL-4) represents that if it is not a holiday, the time is between 17:00 and 22:00, and the distance to the collaboration place is between 20 miles and 100 miles, then the preferred collaboration mode is also by telephone.

Rule (POSL-5) expresses that if it is a holiday, the time is between 09:00 and 22:00, and the distance to the collaboration place is greater than 100 miles, then the preferred collaboration mode is the Web.

Rule (POSL-6) expresses that if it is not a holiday, the time is between 17:00 and 22:00, and the distance to the collaboration place is greater than 100 miles, then the preferred collaboration mode is also the Web.

To exemplify XML serialization, the rules (POSL-1) and (POSL-2) are marked up as two 'Implies' elements in RuleML 0.9[8] as follows.

[7]In order to run these rules in OO jDREW, a "`:Integer`" type will be given to all integer constants.

[8]http://www.ruleml.org/0.9/

```
<Assert>
   <And mapClosure="universal">
      <Implies>
         <And>
            <Atom>
               <Rel>holiday</Rel>
               <Var>Date</Var>
            </Atom>
            <Atom>
               <Rel>greaterThanOrEqual</Rel>
               <Var>Time</Var>
               <Ind type="Integer">10</Ind>
            </Atom>
            <Atom>
               <Rel>lessThanOrEqual</Rel>
               <Var>Time</Var>
               <Ind type="Integer">16</Ind>
            </Atom>
            <Atom>
               <Rel>lessThan</Rel>
               <Var>Distance</Var>
               <Ind type="Integer">20</Ind>
            </Atom>
         </And>
         <Atom>
            <Rel>collaborationMode</Rel>
            <Ind>F2F</Ind>
            <Var>Date</Var>
            <Var>Time</Var>
            <Var>Distance</Var>
         </Atom>
      </Implies>
      <Implies>
         <And>
            <Naf>
               <Atom>
                  <Rel>holiday</Rel>
                  <Var>Date</Var>
               </Atom>
            </Naf>
            <Atom>
               <Rel>greaterThanOrEqual</Rel>
               <Var>Time</Var>
               <Ind type="Integer">16</Ind>
            </Atom>
            <Atom>
               <Rel>lessThanOrEqual</Rel>
               <Var>Time</Var>
               <Ind type="Integer">20</Ind>
            </Atom>
            <Atom>
               <Rel>lessThan</Rel>
               <Var>Distance</Var>
               <Ind type="Integer">20</Ind>
            </Atom>
         </And>
         <Atom>
            <Rel>collaborationMode</Rel>
            <Ind>F2F</Ind>
            <Var>Date</Var>
            <Var>Time</Var>
            <Var>Distance</Var>
         </Atom>
      </Implies>
   </And>
</Assert>
```

V. CONCLUSION

We have proposed a combination of RDF FOAF facts and RuleML FOAF rules. We have extended the current FOAF vocabulary to RuleML FOAF via the human-oriented syntax for facts and rules, POSL. We have then designed the rule vocabulary specification for RuleML FOAF. Next, we have provided scenarios for expert finding, and applied the rules in the rule engine OO jDREW to compute the result for expert finding. Bottom-up execution provides us with the all the newly derived facts as required for the Fact-oriented Normal Form (FNF). Top-down execution enables users intending to find a collaboration expert to query specific information on demand, as requested by the Rule-oriented Normal Form (RNF).

Users can thus derive FOAF data by employing person-centric rules, either before (RDF) FOAF publication or, on demand, from published (RuleML FOAF) pages.

We provide two normal forms, the FNF and RNF. All the facts provided by FNF, original and derived, can be mapped back to RDF syntax on demand via our existing XSLT translator. This translation can benefit the RDF FOAF community in developing enriched FOAF vocabulary specifications as called for by concrete use cases.

Currently, the FNF and RNF are derived interactively, essentially by running OO jDREW BU for the FNF, and checking fact drivability by running OO jDREW TD for the RNF. A control loop for automatically generating the FNF and RNF is planned for the future. More use cases drawn from Teclantic constitute another area of future work.

ACKNOWLEDGMENT

The authors would like to thank for the partial funding provided by the National Research Council of Canada (NRC) and by Mathematics of Information Technology and Complex Systems (MITACS).

REFERENCES

[1] Friendster. http://www.friendster.com/, October 2005.
[2] Mandarax. http://java-source.net/open-source/rule-engines/mandarax, September 2005.
[3] StumbleUpon. http://www.stumbleupon.com/, September 2005.
[4] SweetRules. http://sweetrules.projects.semwebcentral.org/, September 2005.
[5] All Experts. http://www.allexperts.com/, January 2006.
[6] AskanExpert. http://www.askanexpert.com/, January 2006.
[7] Expert Finder Home Page. http://web.media.mit.edu/ lieber/Lieberary/Expert-Finder/ Expert-Finder-Intro.html, January 2006.
[8] Expertise Search. http://www.expertisesearch.com/, January 2006.
[9] ExpertWitness. http://www.expertwitness.com/, January 2006.
[10] Music. http://www.aaai.org/AITopics/html/music.html, March 2006.
[11] Purdue Experts. http://purduenews.uns.purdue.edu/UNS/expertlists/ expert.current.html, January 2006.
[12] Referralweb. http://www.cs.washington.edu/homes/kautz/referralweb/, March 2006.
[13] Teclantic. http://teclantic.cs.unb.ca/index.jsp, January 2006.
[14] Michelle Anderson, Marcel Ball, Harold Boley, Stephen Greene, Nancy Howse, Daniel Lemire, and Sean McGrath. RACOFI: A Rule-Applying Collaborative Filtering System. In *Proceedings of COLA'03*, pages 13–23. IEEE/WIC, October 2003.
[15] Marcel Ball. OO jDREW. http://www.jdrew.org/oojdrew/, September 2005.

[16] Marcel Ball, Harold Boley, David Hirtle, Jing Mei, and Bruce Spencer. The OO jDREW Reference Implementation of RuleML. In *Proc. Rules and Rule Markup Languages for the Semantic Web (RuleML-2005)*, pages 218–223. LNCS 3791, Springer-Verlag, November 2005.

[17] Nick Bassiliades. DR-DEVICE. http://iskp.csd.auth.gr/systems/dr-device.html, January 2006.

[18] Harold Boley. The Rule Markup Language: RDF-XML Data Model, XML Schema Hierarchy, and XSL Transformations. In *Proc. 14th International Conference on Applications of Prolog (INAP2001)*. The University of Tokyo, LNAI 2543, Springer-Verlag, October 2002.

[19] Harold Boley. Integrating Positional and Slotted Knowledge on the Semantic Web. http://www.ruleml.org/posl/poslintweb-talk.pdf, March 2005.

[20] Dan Brickley. The Friend of a Friend (FOAF) project. http://www.foaf-project.org/, November 2005.

[21] Dan Brickley and Libby Miller. FOAF Vocabulary Specification. http://xmlns.com/foaf/0.1/, October 2005.

[22] Elizabeth F. Churchill and Christine A. Halverson. *IEEE Internet Computing*, volume 9, chapter Social Networks and Social Networking, pages 14–19. IEEE Computer Society, 2005.

[23] Edd Dumbill. XML Watch: Finding friends with XML and RDF. http://www-128.ibm.com/developerworks/xml/library/x-foaf.html, June 2002.

[24] Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu. Semantically Rich Recommendations in Social Networks for Sharing and Exchanging Semantic Context. In *Proc.4th International Semantic Web Conference (ISWC2005)*, pages 293–307, Galway, Ireland, November 2005.

[25] Gunnar AAstrand Grimnes, Pete Edwards, and Alun Preece. Learning Meta-Descriptions of the FOAF Network. In *ISWC 2004, LNCS 3298*, pages 152–165. Springer-Verlag, Berlin, 2004.

[26] RSS-DEV Working Group. RDF Site Summary 1.0. http://web.resource.org/rss/1.0/spec, October 2005.

[27] Masahiro Hamasaki, Junichiro Mori, Hideaki Takeda, and Koiti Hasida. Ontological Consideration on Human Relationship Vocabulary for FOAF. 1st Workshop on Friend of a Friend, Social Networking and the Semantic Web, 2004.

[28] Elliotte Rusty Harold. *The XML Bible, 2nd Edition*, chapter 17: XSL Transformations. John Wiley & Sons, 2001.

[29] Luke McDowell, Oren Etzioni, Steven D. Gribble, Alon Halevy, Hank Levy, William Pentney, Deepak Verma, and Stani Vlasseva. Mangrove: Enticing Ordinary People onto the Semantic Web via Instant Gratification. In *Proc. 2nd International Semantic Web Conference (ISWC2003)*, pages 754–770, Sanibel Island, Florida, USA, October 2003. Springer.

[30] Luke McDowell, Oren Etzioni, Steven D. Gribble, Alon Halevy, Henry Levy, William Pentney, Deepak Verma, and Stani Vlasseva. Evolving the Semantic Web with Mangrove. Technical Report UWCSE030201, Department of Computer Science and Engineering, University of Washington, Seattle, WA, February 2003.

[31] Ikki Ohmukai1, Hideaki Takeda, Masahiro Hamasaki, Kosuke Numa, and Shin Adachi. Metadata-Driven Personal Knowledge Publishing. In *ISWC 2004, LNCS 3298*, pages 591–604. Springer-Verlag, Berlin, 2004.

[32] W3C Candidate Recommandation. XSL Transformations (XSLT) Version 2.0. http://www.w3.org/TR/xslt, November 2005.

[33] W3C Recommendation. RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR/rdf-schema/, February 2004.

[34] Bruce Spencer. jDREW. http://www.jdrew.org/jDREWebsite/jDREW.html, October 2005.

[35] Lu Yang, Marcel Ball, Virendrakumar C. Bhavsar, and Harold Boley. Weighted Partonomy-Taxonomy Trees with Local Similarity Measures for Semantic Buyer-Seller Match-Making. Business Agents and the Semantic Web (BASeWEB) Workshop, 2005.