PatientSupporter: A Social Semantic Web Portal for Virtual Support Groups

Omair Shafiq
Department of Computer Science
University of Calgary
Calgary, AB, Canada
moshafiq[AT]ucalgary.ca

Harold Boley, Derek Smith[2], Taylor Osmun
Institute of Information Technology
National Research Council of Canada
Fredericton, NB, Canada
firstname.lastname[2][AT]nrc.gc.ca

ABSTRACT

Patients are increasingly seeking interaction in support groups, which provide shared information and experience about diagnoses, treatments, etc. We present a Social Semantic Web prototype, PatientSupporter, that will enable such networking between patients within a virtual organization. PatientSupporter is an instantiation of Rule Responder that permits each patient to query other patients' profiles for finding or initiating a matching group. Rule Responder's External Agent (EA) is a Web-based patient-organization interface that passes queries to the Organizational Agent (OA). The OA represents the common knowledge of the virtual patient organization, delegates queries to relevant Personal Agents (PAs), and hands validated PA answers back to the EA. Each PA represents the medical subarea of primary interest to an associated patient group. The PA assists its patients by advertising their interest profiles, employing rules about diagnoses and treatments as well as interaction constraints such as time, location, age range, gender, and number of participants. Profiles can be distributed across different rule engines using different rule languages (e.g., Prolog and N3), where rules, queries, and answers are interchanged via translation to and from RuleML/XML. We discuss the implementation of PatientSupporter in a use case where the PA's medical subareas are defined through sports injuries structured by a partonomy of affected body parts.

1. INTRODUCTION

Social Web (Web 2.0) techniques have been explored in recent years for applications in healthcare [13, 12]. Web 2.0 systems have been developed that help patients to network with other patients having similar ailments to discuss and exchange information and experiences [18]. Complementing these approaches with Semantic Web techniques, the current paper introduces Social Semantic Web (Web 3.0) ontologies and rules for organizing geographically distributed patients – here, suffering from sports injuries – into virtual support groups around classes of an ontology of injuries – here, a sports-injury partonomy such as [23].

Described here is our prototype, PatientSupporter, which is designed to help patients with a similar sports injury to interact with a virtual support group having that common interest. Patients in an online PatientSupporter virtual organization create their semantic profile referring to classes in a disease ontology – here a partonomy of body parts affected by sports injuries. This body partonomy allows patients to base the description of their injuries on a `subPartOf` hierarchy leading to affected body parts, which is actually implemented as a corresponding `subClassOf` taxonomy of injury classes for those body parts. Profiles contain rules about diagnoses and treatments as well as interaction constraints such as time, location, age range, gender, and number of participants. A patient can pose queries against the semantic profiles of other patients in his or her virtual organization to find or initiate a matching group. PatientSupporter is built upon Rule Responder [19, 20] which has also been used in the related Social Semantic Web instantiation WellnessRules [9] as well as in SymposiumPlanner [11].

PatientSupporter allows patients to have their profiles expressed in either Pure Prolog [22] (Logic Programming rules) or N3 [3] (Semantic Web rules). Providing these quite different rule language paradigms permit patients to choose the language that best suits them. Rule Responder handles the interoperation between the rule languages of different patients using translators to and from RuleML/XML as the interchange format [4].

As an example, let us consider a patient, Paul, who has injured part of his left leg during a rugby game. He has questions about his injury and precautions for its recovery that others with similar injuries may be able to answer or help with. Using PatientSupporter, Paul poses a query through the External Agent (EA), focussing on leg injuries. The EA submits the query to the Organizational Agent (OA), which delegates it to the relevant Personal Agents (PAs) and collects answers from their local knowledge bases. When the OA returns the validated answers to the EA, Paul discovers that his query was too broad as it resulted in far too many answers. Paul, who actually hurt his left knee, thus proceeds downward the partonomy by querying PatientSupporter just for patients with knee injuries. Now, the answers returned are less in numbers and more relevant. Paul picks a subset of the returned profiles and queries them with his interaction constraints, proposing a knee-injury discussion in a Skype-based conference call on the upcoming Saturday

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

or Sunday any time between 10AM and 6PM EST. Paul's queries return the Skype IDs of patients interested in the discussion with the time narrowed down to Sunday between 3PM and 6PM EST. Hence Paul invites them for a first call Sunday, 4PM to 5PM, effectively initiating an ad hoc virtual support group.

It should be noted that Paul by using the PatientSupporter Social Semantic Web portal is able to initiate the virtual support group about his sports injury on a global scale. He also benefits from PatientSupporter's interoperation facility in the background – to transform patient profiles between Pure Prolog and N3 through RuleML/XML. The system employs a partonomy of sports-injury-affected body parts, which makes it easy for Paul to navigate hierarchically up or down, increasing recall or precision, respectively. Paul's queries invoke other patients' interaction rules, allowing him to narrow down his search in a step-wise fashion. All of this saves him from browsing through a large set of irrelevant patient profiles and permits him to efficiently converge on a first Skype call.

The rest of the paper describes the design and implementation of PatientSupporter and is organized as follows. Section 2 goes over the design goals of PatientSupporter and its instantiation based on Rule Responder. Section 3 discusses the global knowledge base used for the OA. Section 4 describes the use of local knowledge bases to represent the profiles of individual patients underneath the PAs. Section 5 expands upon the RuleML-based interoperation between Pure Prolog and N3 rules. Section 6 explains and demonstrates the use of RuleML-based querying for patients in a distributed setting. Section 7 evaluates the overall effectiveness and usefulness of PatientSupporter as a use case of the Social Semantic Web. Section 8 concludes the paper. Appendix A provides information about rule signatures and the body partonomy.

2. PATIENTSUPPORTER AS AN INSTANTIATION OF RULE RESPONDER

PatientSupporter is based on Rule Responder, where this reference architecture with each of its main agent types (i.e., EA, OA, and PAs) is instantiated as follows. It performs virtual support group matchmaking by querying patients organized in a disease ontology – here, a body partonomy. The following design goals have been pursued while developing the PatientSupporter prototype:

1. Identify a language of appropriate expressiveness to model patient profiles from the family of RuleML languages [6], Pure Prolog [22], and N3 [3].

2. Identify a language for light-weight ontologies of sports injuries, internal medicine, etc., such as `subPartOf` partonomies from the layered family of RDFS and OWL 2. The ontology language is to be combined with the rule language.

3. Allow eliciting rule and ontology definitions in human-oriented syntaxes, while translating the resulting rulebases and ontologies to and from RuleML/XML for interchange.

4. Allow different rule engines (e.g., OO jDREW [2], Prova [16], and Euler[1]) to execute global and local rulebases.

5. Allow rules as well as queries and their answers to be transmitted over an Enterprise Service Bus (ESB) – e.g., Mule [15] or IBM Service Bus [14].

6. Elicit exemplary patient profiles and abstract them to generally usable profile templates for increased usability and reusability.

7. Teach Kinesiology students to support each other regarding similar injuries through the formation and evolution of virtual support groups based on profile authoring and querying.

8. Investigate the appropriateness of languages and engines for rules as well as ontologies, to express, process, and transform the knowledge required in patient profiles.

9. Evaluate the effectiveness and usefulness of the distributed PatientSupporter architecture, based on its ESB-interconnected engines using different languages for the dynamic formation of virtual patient support groups.

10. Adapt PatientSupporter from the sports-injury domain to an internal-medicine domain, and – based on the experience with both – recommend future PatientSupporter domains and Rule Responder instantiations.

The current implementation of PatientSupporter models patient profiles in POSL [5] and N3 [3]. The profiles are interoperated through RuleML/XML as the intermediate format. Users are aided by a menu-based form, which makes it easier to query patient profiles. Knowlege about patients and their injuries is organized using rules combined with light-weight ontologies in sorted (typed) Horn logic or N3. The `subPartOf` partonomy is written in RDFS (cf. appendix A). Human-oriented syntaxes have been used while modeling the patient profiles. The overall communication and coordination of the rule engines (e.g., OO jDREW [2], Prova [16] and Euler [21] has been organized through Mule[2], an open-source ESB. The use of an ESB allows architectural flexibility by decoupling the functional components of Rule Responder from the communication components [10].

Rule Responder's instantiation to PatientSupporter in the sports-injury domain allows to create virtual patient organizations consisting of virtual support groups defined through sports injuries structured by a partonomy of affected body parts (further explained in section 3). The OA becomes an assistant to an entire virtual patient organization. Each PA becomes an assistant to a group of patients having the same class of injuries from the partonomy, and helps them as *profile users* to get organized as a support group. The EA is utilized by patients as *enquiry users* to (register with its virtual organization and) query the profiles of the virtual organization's other patients.

Rule Responder employs the following sequence of steps: An enquiry user interacts with the EA to author and submit queries to an OA. The OA assigns (maps and delegates) each query topic to the PA(s) most knowledgeable about it. Each PA poses the query to its own local rulebase, and returns the derived answer(s) to the OA. The OA collects and validates answers and gives the overall answer(s) back to the EA, hence to the enquiry user.

[1]http://eulersharp.sourceforge.net/

[2]http://www.mulesoft.org/

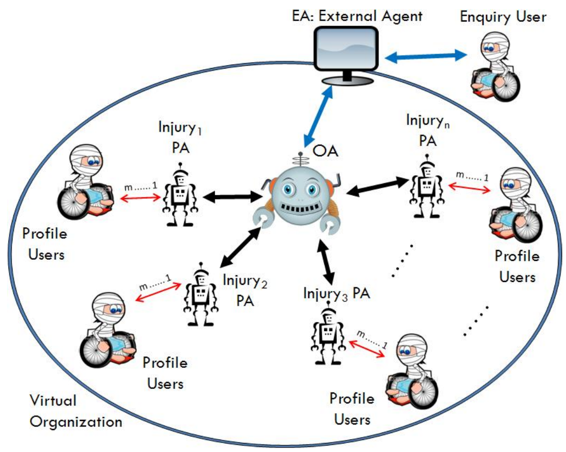

Figure 1: Overall architecture of PatientSupporter

Rule Responder's earlier instantiations include SymposiumPlanner [11] and WellnessRules [9]. However, PatientSupporter further extends the functionality of Rule Responder by making Social Semantic use of partonomies: Patient injuries are classified in the hierarchical part-whole manner of affected body parts. For example, injuries related to Foot are subordinated as `subPartOf` of Leg. Similarly, Heel and Toe injuries are subordinated as `subPartOf` under Foot. Employing partonomies as light-weight ontologies in this way allows the 'refinement' of a virtual support group (e.g., about Leg injuries) into subgroups (e.g., about Foot injuries and further about Heel and Toe injuries) along the transitive $\texttt{subPartOf}^{-1}$ paths.

PatientSupporter, as an extension of WellnessRules, allows users to be supported by PAs as follows: Each patient (as a profile user) publishes a profile employed by the responsible PAs to respond to queries (from enquiry users) about his or her preferences, constraints, etc. This dynamic profile association is implemented via the Profile Responsibility Matrix (PRM), and is not possible in SymposiumPlanner, where only chairs (as profile users) are supported by PAs.

The main agent types of PatientSupporter are described in the following subsections. Figure 1 depicts the interaction between the EA, OA, and PAs.

2.1 External Agent

The External Agent (EA) is the point-of-contact that allows a patient to query the Organizational Agent (OA) of a virtual patient organization. It is based on a Web interface that allows him or her as an enquiry user to pose queries employing a menu-based form, which gets translated to equivalent RuleML/XML. A sports-injury patient primarily selects the injury class from the partonomy. He or she can also fill in property values about diagnoses and treatments as well as interaction constraints. The finished RuleML/XML query is submitted to the OA with a "Send Message" button. Finally, the EA presents the OA's answer(s) to the patient.

2.2 Organizational Agent

The Organizational Agent (OA) is at the center of PatientSupporter, representing a virtual patient organization as a whole. The knowledge base of the OA is *global* across the virtual organization, and is written in Prova-the-language and run in Prova-the-engine [16]. The OA also employs two matrices, on the basis of which incoming queries are mapped and delegated: The *Group Responsibility Matrix*, written as an OWL-Lite ontology, defines which group headed by a PA is best for which kind of query. The *Profile Responsibility Matrix*, written as an XML document, defines which patient profiles exist in a PA's group, and in which formats (e.g., POSL or N3).

2.3 Personal Agent

The Personal Agents (PAs) contain disease-oriented groups of patient profiles, where diseases are here restricted to sports injuries. Each PA heads the group of patient profiles listed in the Profile Responsibility Matrix (cf. section 2.2). The knowledge base of each profile under a PA is *local* to that profile, and is either written in POSL and run in OO jDREW [2] or is written in N3 and run in Euler[3].

3. GLOBAL KNOWLEDGE BASE FOR PATIENTS

The ontologies and a subset of the rules are globally shared via the OA to benefit all the PAs. Another subset of rules is distributed amongst the PAs, where it is kept local (cf. section 4). The shared ontology and the shared subset of rules are referred to as *global knowledge base*.

Global knowledge in PatientSupporter is thus modeled as a combination of ontologies and rules. The ontologies include a light-weight ontology realizing the Group Responsibility Matrix (cf. section 2.2) and a body partonomy. The global rules include general constraints and preferences of a virtual organization. PatientSupporter makes use of the standard rule format RuleML/XML and the Rule Responder reference implementation to transform to and from other rule languages.

The body partonomy was elicited as a common-sense ontology to reflect the patient-centric perspective of support groups. It is drawing, among others, on the Digital Anatomist Foundational Model of Anatomy (FMA) [17], the online Sports Injury Clinic [23], and a medically trained NRC-IIT colleague. It is referred to as a partonomy because it represents the logical hierarchy of body parts; however, it is implemented as a taxonomy of injuries affecting the body parts. I.e., *A* `subPartOf` *B* implies *A Injury* `subClassOf` *B Injury*. Note that in this patient-centric representation we do not express *what* injury it is, but only *where* it is.

Under `Body`, PatientSupporter uses the partonomy classes `Head`, `Neck`, `Shoulder`, `Arm`, `Torso`, `Back`, and `Leg`. All of `Thigh`, `Lower Leg`, `Knee`, and `Foot` are regarded as direct parts of `Leg`. `Toe` and `Heel` are likewise part of `Foot`. The complete partonomy is shown in figure 2. Appendix A gives its `subPartOf` definition in RDFS.

The rule component in PatientSupporter employs Naf Hornlog POSL and N3 with scoped Naf. The use of Naf Hornlog POSL has been restricted to atoms with positional arguments, leaving F-logic-like frames with property-value slots to N3. This demonstrates the range of our approach through complementary rule styles. For that reason, the POSL syntax corresponds to pure-Prolog syntax except that POSL variables are prefixed by a question mark while Prolog variables are upper-cased.

This Hornlog POSL sublanguage uses (positional) n-ary relations (or, predicates) as its central modeling paradigm.

[3]http://eulersharp.sourceforge.net/

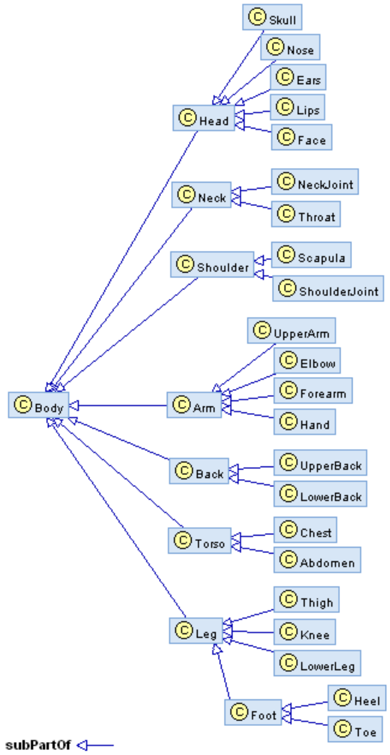

Figure 2: Patient-centric body partonomy for localizing sports injuries

N3 instead uses (unordered) sets of binary relations (or, properties) centered around object identifiers (OIDs, called 'subjects' in RDF and N3).

The following POSL example gives the *positional signature* of the 16-ary predicate `myDiscussion`:

```
myDiscussion(?ProfileID,?Injury,?MinAge,?MaxAge,?MinRSVP,
?MaxRSVP,?Category,?Treatment,?HealingStage,?StartTime,
?EndTime,?Duration,?Channel,?Contact,?Gender,?TimeZone).
```

In N3 this becomes a *slotted signature* with subject `_:myDiscussion`, an `rdf:type` of `:MyDiscussion`, and the 16 arguments as the remaining slots:

```
_:myDiscussion
    rdf:type        :MyDiscussion;
    :profileID      ?ProfileID
    :injury         ?Injury;
    :minAge         ?MinAge;
    :maxAge         ?MaxAge;
    :minRSVP        ?MinRSVP;
    :maxRSVP        ?MaxRSVP;
    :category       ?Category;
    :treatment      ?Treatment;
    :healingStage   ?HealingStage;
    :startTime      ?StartTime;
    :endTime        ?EndTime;
    :duration       ?Duration;
    :channel        ?Channel;
    :contact        ?Contact;
    :gender         ?Gender;
    :timeZone       ?TimeZone.
```

Shared rules (including underlying facts) defining these predicates in the PatientSupporter system have been collected for the rulebase of the OA. They, together with the ontologies, formalize the global knowledge of the PatientSupporter system.

For example, this is a local `myDiscussion` fact according to the above positional signature. It was generated by a local rule which used global rules and facts to satisfy its predicates:

```
myDiscussion(
     p0001,
     Injury:Leg,
     20:integer,
     50:intger,
     5:integer,
     10:integer,
     Out,
     Bandage,
     Medium,
     dateTime[2010:integer,
              6:integer,
              1:integer,
              10:integer,
              15:integer],
     dateTime[2010:integer,
              6:integer,
              1:integer,
              11:integer,
              20:integer],
     dateTime[0:integer,
              0:integer,
              0:integer,
              0:integer,
              30:integer],
     Skype,
     skypeUserName,
     Female,
     -400).
```

Similarly, given below is its slotted counterpart:

```
:myDiscussion_1
    rdf:type        :MyDiscussion;
    :profileID      :p0001;
    :injury         :Leg;
    :minAge         :20;
    :maxAge         :50;
    :minRSVP        :5;
    :maxRSVP        :10;
    :category       :Out;
    :treatment      :Bandage;
    :healingStage   :Medium;
    :startTime      [:year 2010; :month 6;
                     :day 1;     :hour 10;
                     :minute 15];
    :endTime        [:year 2010; :month 6;
                     :day 1;     :hour 11;
                     :minute 20];
    :duration       [:year 0; :month 0;
                     :day 0;  :hour 0;
                     :minute 30];
    :channel        :skype,
    :contact        :skypeUserName,
    :gender         :Female,
    :timeZone       :-400.
```

Both express that a `myDiscussion` could take place about `Leg` injuries, with `Medium` stage of healing, `Bandage` level for treatment, and with category as `Out` patient. It will be on June 1st, 2010, between 10:15 AM and 11:20 AM (GMT -4:00 Atlantic Time) for a duration of 30 minutes. It will have the form of a Skype call for 5 to 10 people. The skype user name of the person advertising this time is skypeUserName.

An example of a global POSL rule defines a `participation` in a virtual support group as follows:

```
participation(?ProfileID,?Injury,
                    ?Min,?Max) :-
      groupSize(?ProfileID,?Injury,
              ?MinRSVP,?MaxRSVP),
      greaterThanOrEqual(?MinRSVP,?Min),
      lessThanOrEqual(?MaxRSVP,?Max).
```

The first argument of the conclusion predicate `participation` as always is the person the rule is about. Similar to Prolog, the rule succeeds for its four positional arguments if the acceptable `groupSize` of the patient with `?ProfileID`, for an `?Injury`, is between `?Min` and `?Max`, and the emerging support group's size is between `?MinRSVP` $\geq$ `?Min` and `?MaxRSVP` $\leq$ `?Max`.

The corresponding global N3 rule for deriving `_:participation` facts is as follows:

```
{
 ?rsvpQuery
  rdf:type    :RSVPQuery;
  :profileID  ?ProfileID;
  :minRSVP    ?Min;
  :maxRSVP    ?Max.

 ?groupSize
  rdf:type    :GroupSize;
  :profileID  ?ProfileID;
  :injury     ?Injury;
  :min        ?MinRSVP;
  :max        ?MaxRSVP.
  http://ruleml.org/posl/converter.jnlp
 ?Min math:notLessThan ?MinRSVP.

 ?Max math:notGreaterThan ?MaxRSVP.
}
=>
{
 _:participation
  rdf:type    :Participation;
  :profileID  ?ProfileID;
  :injury     ?Injury;
  :min        ?Min;
  :max        ?Max.
}.
```

The global OA knowledge base is being maintained in both language paradigms, i.e. POSL[4] and N3[5].

4. LOCALLY DISTRIBUTED KNOWLEDGE BASES FOR INDIVIDUAL PATIENTS

Locally distributed knowledge bases are grouped as profiles underneath the PAs. Each PA group has its own kind of knowledge base, according to the medical subarea associated with the body partonomy (cf. section 3). For example, the profiles created by patients with `Leg` injuries are kept with the `Leg` PA.

The local knowledge bases have information about patients to model their profiles using a vocabulary of properties, namely unique identifier of a profile as `ProfileID`, kind of injury of the patient as `Injury`, the age of the patient as `Age`, time zone of the the patient as `TimeZone`, treatment required as `Treatment`, stage of healing of the injury as `HealingStage`, and the category information as `Category`. The properties `Treatment`, `HealingStage`, and `Category` have the following allowed value ranges. The `Treatment` property currently has values `Bandage`, `MajorOperation`, `MediumOperation`, `MinorOperation`, `MajorMedication`, `MediumMedication`, `MinorMedication`, or `ChangeOfLifeStyle`. The `HealingStage` property has values `Fresh`, `Medium`, `Convalescent`, or `Healed`. The `Category` property has values `In` or `Out` patient.

Given below is an example of a local POSL rule from a PA knowledge base of patient `p0001`, defining the main predicate `myDiscussion` about `Leg` injuries, speciying his desired support-group discussion:

```
myDiscussion(p0001,Injury:Leg,?MinAge,?MaxAge,
  ?MinRSVP,?MaxRSVP,Out,?Treatment,?HealingStage,
  dateTime[?StartYear,?StartMonth,?StartDay,?StartHour,?StartMinute],
  dateTime[?EndYear,?EndMonth,?EndDay,?EndHour,?EndMinute],
  dateTime[?DurYear,?DurMonth,?DurDay,?DurHour,?DurMinute],
  ?Channel,?Contact,?Gender,?TimeZone) :-
  event(p0001,Injury:Leg,Possible,
                      ?StartTime,?EndTime),
  participation(p0001,Injury:Leg,
                       ?MinRSVP,?MaxRSVP),
  communication(p0001,?Channel),
  healingStage(p0001,Injury:Leg,?HealingStage),
  treatment(p0001,?Treatment).
```

The rule conclusion's `myDiscussion` predicate starts with the person's profile ID, `p0001`, followed by the kind of injury, `Leg`, and its category as `Out` patient, followed by variables `?HealingStage`, the `?Treatment` that is required, as well as `?MinRSVP` and `?MaxRSVP` for the virtual support group limits.

The rule premises query `p0001`'s `?StartTime` and `?EndTime` of his or her `possible events`, the `participation` constraints, a `communication ?Channel`, as well as the `healingStage` and `treatment` constraints.

The corresponding local N3 rule is given abridged below:

```
{
 ?event
  rdf:type      :Event;
  :calendarID   ?ProfileID;
  :injury       :Leg;
  :tense        :Possible;
  :startTime    ?StartTime;
  :endTime      ?EndTime.

 ...
}
=>
{
 _:myDiscussion
  rdf:type       :MyDiscussion;
  :profileID     :p0001;
  :injury        :Knee;
  :minAge        ?MinAge;
  :maxAge        ?MaxAge;
  :minRSVP       ?MinRSVP;
  :maxRSVP       ?MaxRSVP;
  :category      :Out;
  :treatment     ?Treatment;
  :healingStage  ?Stage;
  :startDateTime ?StartTime;
  :endDateTime   ?EndTime;
  :duration      ?Duration;
  :channel       ?Channel;
  :contact       ?Contact;
  :gender        ?Gender;
  :timeZone      ?TimeZone.
}.
```

The POSL and N3 rules are available online[6].

5. INTEROPERATION BETWEEN POSL AND N3 RULES VIA RULEML/XML

The PatientSupporter use case includes a testbed for the interoperation (i.e., alignment and translation) of information in knowledge bases in the main two rule paradigms: Prolog-style (positional) relations and N3-style (slotted) frames. PatientSupporter inherits the interoperation mechanisms from Rule Responder [1]. The interoperation methodology makes iterative use of alignment and translation: An initial alignment permits the translation of parts of a hybrid knowledge base. This then leads to more precise alignments, which in turn lead to better translations, etc. Using this methodology, PatientSupporter can maintain relational (Pure Prolog)

[4]http://ruleml.org/PatientSupporter/PS-Global.posl
[5]http://ruleml.org/PatientSupporter/PS-Global.n3
[6]http://ruleml.org/PatientSupporter/PA/

as well as frame (N3) versions of rules, both accessing the same, independently maintained, body partonomy.

The PAs of PatientSupporter can thus use either of these rule paradigms, while interoperation is carried out through the intermediate rule language RuleML/XML, which has sublanguages for both of them, so that the cross-paradigm translations can use the common XML syntax of RuleML. A pair of online converters[7] is used for rulebase conversion between the human-oriented POSL syntax and its XML serialization in RuleML.

For rulebase translation, the signatures of PatientSupporter relations and frames are aligned in a shared signature document (cf. appendix A), discussed in section 3, which specifies the argument positions of relations and slot names of frames. The alignment of sample relations and frames in sections 3 and 4 suggests translations between both of the rule paradigms.

Translations that are considered to be 'static' or 'at compile-time', take an entire rulebase as input and return its entire transformed version in RuleML/XML. Thus, an assumption of 'closed-arguments' of fixed signatures for relations and frames is made [9].

The translation facility is available as an XSLT implementation which is also available online[8].

For example, the POSL `myDiscussion` fact of section 3 is serialized in positional RuleML as follows: where `Ind`ividual constants are distinguished from `Data` literals:

```
<Atom>
  <Rel>myDiscussion</Rel>
  <Var>ProfileID</Var>
  <Ind type="Leg">Injury</Ind>
  <Data type="integer">20</Data>
  <Data type="integer">50</Data>
  <Data type="integer">5</Data>
  <Data type="integer">10</Data>
  <Data>Out</Data>
  <Ind>Bandage</Ind>
  <Data>Medium</Data>
  <Var>StartTime</Var>
  <Var>EndTime</Var>
  <Var>Duration</Var>
  <Var>Channel</Var>
  <Var>Contact</Var>
  <Var>Gender</Var>
  <Var>TimeZone</Var>
</Atom>
```

Extending the mappings in OO RuleML[9], the N3 `myDiscussion` fact of section 3 is serialized in slotted RuleML as follows, where RuleML's `Rel` represents N3's `rdf:type`:

```
<Atom>
  <oid><Ind iri=":myDiscussion_1"/></oid>
  <Rel iri=":MyDiscussion"/>
  <slot>
    <Ind iri=":profileID"/>
    <Var>ProfileID</Var>
  </slot>
  <slot>
    <Ind iri=":injury"/>
    <Ind>Leg</Ind>
  </slot>
  <slot>
    <Ind iri=":minAge"/>
    <Data type="integer">20</Data>
  </slot>
  <slot>
    <Ind iri=":maxAge"/>
    <Data type="integer">50</Data>
  </slot>
  <slot>
    <Ind iri=":minRSVP"/>
    <Data type="integer">5</Data>
  </slot>
  <slot>
    <Ind iri=":maxRSVP"/>
    <Data type="integer">10</Data>
  </slot>
  <slot>
    <Ind iri=":category"/>
    <Data>out</Data>
  </slot>
  <slot>
    <Ind iri=":treatment"/>
    <Ind>Bandage</Ind>
  </slot>
  <slot>
    <Ind iri=":healingStage"/>
    <Data>Medium</Data>
  </slot>
  <slot>
    <Ind iri=":startTime"/>
    <Var>StartTime</Var>
  </slot>
  <slot>
    <Ind iri=":endTime"/>
    <Var>EndTime</Var>
  </slot>
  <slot>
    <Ind iri=":duration"/>
    <Var>Duration</Var>
  </slot>
  <slot>
    <Ind iri=":channel"/>
    <Var>Channel</Var>
  </slot>
  <slot>
    <Ind iri=":contact"/>
    <Var>Contact</Var>
  </slot>
  <slot>
    <Ind iri=":gender"/>
    <Var>Gender</Var>
  </slot>
  <slot>
    <Ind iri=":timeZone"/>
    <Var>TimeZone</Var>
  </slot>
</Atom>
```

The translators are currently being extended to RIF [8], starting with the Dlex subset of RIF and RuleML [7].

6. DISTRIBUTED RULE RESPONDER QUERYING

PatientSupporter inherits the distributed query mechanism from Rule Responder. For querying different rule engines, transformations between queries and answers from N3 and Pure Prolog through RuleML/XML are done as described for rules in section 5. Both the global knowledge base, described in section 3, and locally distributed knowledge bases, described in section 4, can be queried.

Given below is an example of a POSL query for patient profiles, which will be executed by Rule Responder's OO jDREW TD (Top-Down) engine:

```
myDiscussion(?ProfileID,Injury:Leg,20:integer,50:integer,5:integer,
 10:integer,Out,Bandage,Medium,?StartTime,?EndTime,?Duration,
 ?Channel,?Contact,?Gender,?TimeZone)
```

It uses the rule from section 4 to check whether any patient `(?ProfileID)` has `Leg` injury, is `Out` patient, has `Medium` healing stage, has treatment `Bandage`, with any start time,

[7]http://ruleml.org/posl/converter.jnlp
[8]http://ruleml.org/ooruleml-xslt/oo2prml.html
[9]http://ruleml.org/indoo/n3ruleml.html

any end time, willing to join a group of minimuim `5:integer` to maximum `10:integer`, with people between ages of `20:integer` and `50:integer`, using any channel.

This is the corresponding N3 query, which will be executed by Rule Responder's EulerSharp EYE bottom-up engine:

```
@prefix : <patient_profiles#>.
@prefix rdf: <http://www.w3.org/1999/02/22-
             rdf-syntax-ns#>.

  _:myDiscussion
  rdf:type        :MyDiscussion;
  :profileID      ?ProfileID;
  :injury         :Leg;
  :minAge         ?MinAge;
  :maxAge         ?MaxAge;
  :minRSVP        ?MinRSVP;
  :maxRSVP        ?MaxRSVP;
  :category       :Out;
  :treatment      :Bandage;
  :healingStage   :Medium;
  :startTime      ?StartTime;
  :endTime        ?EndTime;
  :duration       ?Duration;
  :channel        ?Channel;
  :contact        ?Contact;
  :gender         ?Gender;
  :timeZone       ?TimeZone.
```

After declaring two prefixes, it builds an existential '(`_`)' node, `_:myDiscussion`, using slots for the fixed parameters and the fact-provided `?MinRSVP` (5) and `?MaxRSVP` (10) bindings to fill slots with the `?ProfileID`, `?StartTime`, `?EndTime`, and `?Channel` solutions.

The above sample query could be narrowed down to produce exactly one solution, e.g. by changing the parameter `Out` patient to `In` patient. However, it would fail for some other queries such as those with `?MaxRSVP` greater than 20. Using variations of queries, patients as enquiry users of PatientSupporter can explore profiles of patients as profile users to find or initiate a support group. The implemented Rule Responder instantiation for PatientSupporter, with test queries, is available online.[10]

7. EVALUATION

In our experiments, the overall execution times for the online-selectable `myDiscussion` PatientResponder test queries in Rule Responder on average were 2407ms, 2427ms, and 5371ms in the selection order, measured as the Java system time, in Java JRE6, Windows XP Professional SP3, on an Intel Core 2 Duo 2.80GHz processor.

The usefulness of PatientResponder has further been examined in three different perspectives. The user perspective examines PatientSupporter's use for enquiry users and profile users (cf. section 7.1). The semantics perspective examines PatientSupporter as a Social Semantic Web system (cf. section 7.2). The infrastructure perspective focuses on the Rule Responder infrastructure and its capabilities as instantiated to PatientSupporter (cf. section 7.3).

7.1 User Perspective

The enquiry user working with PatientSupporter's EA can alternatively fill out a RuleML/XML template or do selections in a menu-based form. As expected, most users prefer the menu-based form for queries for which no templates are available. When sample queries and their rendering in plain English are similar to those the user wants to pose him/herself, he or she typically prefers to modify the closest samples. The broadness or narrowness of a query in terms of its level in the partonomy and its number of constants determines the recall and precision in finding the right profiles, ranging from too many, just the right number, exactly one, or none.

For profile rules, as for queries, the level in the partonomy and the number of constants determine recall and precision in being found by the right queries. This would further be made easier by providing patients with templates having comments about what each section is about. The freedom to write one's profile in Pure Prolog or N3 permits the profile user to employ his or her favorite paradigm. Finally, the profile user may choose to include or not, any identifying information or specific sports injuries. By employing semantically enabled profiles and queries, both profile and enquiry users can get the benefits of tuning search recall and precision for patient-patient matchmaking.

7.2 Semantics Perspective

By using ontologies and rules, PatientSupporter can be modified or extended by domain experts in an easier way.

The body partonomy can be refined by medical experts in its graphic (cf. figure 2) or symbolic form, without having to know the underlying technical implementation. Similarly, the OWL-Lite ontology implementing the Group Responsibility Matrix (cf. section 2.2) can be changed declaratively.

The global rulebase of the OA can be changed with knowledge about the behavioral logic and some knowledge about Rule Responder's implementation in Java. The local rulebases of the PAs can be changed with knowledge needed just about the decision logic, where each PA profile is a separate module that can be changed independently from all the other ones.

7.3 Infrastructure Perspective

Rule Responders allows the interoperation of different rule engines, e.g. Prova, OO jDREW, Euler, and, most recently, DR-Device. The transformation is inherited from Rule Responder to PatientSupporter.

Rule Responder allows updating of any of the rule engines, through the use of an ESB. The modular approach enables scalability of Rule Responder by allowing to bring in new rule engines, without disrupting the existing system. System users further benefit from Rule Responder's use of RuleML/XML as an interchange language, which allows the uniform production and consumption of test queries, answers, as well as entire rulebases.

8. CONCLUSION

PatientSupporter demonstrates a Social Semantic Web portal for patients who want to collaborate and share information with each other about sports injuries, on a global scale. It allows efficient networking between patients by using ontologies and rules to allow precise search of patient profiles. Key features of PatientSupporter are: First it allows interoperation of patient profiles between Pure Prolog (Naf Hornlog) and N3 through RuleML/XML. Second, it enables scalability of distributed knowledge on the Social Semantic Web via its PA modules, starting with derivation rules and light-weight ontologies. Third, PatientSupporter uses the OO jDREW, Euler, and Prova engines, while its open Rule Responder architecture makes it easy to bring

[10]http://ruleml.org/PatientSupporter/RuleResponder.html

in new engines. Fourth, it makes use of a body partonomy for modeling sports injuries in a hierarchical manner (from the patients point-of-view). Fifth, it makes use of ontologies and rules to precisely search for patient profiles, and allows enquiry users to narrow down their search in a step-wise fashion. Hence, delivering the relevant profiles, PatientSupporter saves enquiry users from the hassle of browsing through a large set of patient profiles.

The next extension of PatientSupporter is planned to include medical professionals (e.g. doctors and nurses) along with the patients for an internal-medicine domain. This should assist in the formation of virtual support groups consisting of doctors and nurses as well as patients, based on the preferences and constraints of all three subgroups. PatientSupporter and its use cases in sports injuries and internal medicine will also provide new challenges and suggestions for improvements of RuleML, Rule Responder, and the involved engines.

9. ACKNOWLEDGMENTS

The authors would like to thank their colleagues at NRC-IIT Fredericton, especially Benjamin Craig and Julie Maitland, for various helpful discussions. Work by Omair Shafiq was done during a stay at NRC-IIT. NSERC is thanked for its support through a Discovery Grant for Harold Boley.

10. REFERENCES

[1] M. Ball, H. Boley, D. Hirtle, J. Mei, and B. Spencer. Implementing ruleml using schemas, translators, and bidirectional interpreters. In *Rule Languages for Interoperability*, 2005.

[2] M. Ball and B. Craig. Object Oriented java Deductive Reasoning Engine for the Web. http://www.jdrew.org/oojdrew/.

[3] T. Berners-Lee, D. Connolly, L. Kagal, Y. Scharf, and J. Hendler. N3Logic: A Logical Framework For the World Wide Web. *Theory and Practice of Logic Programming (TPLP)*, 8(3), May 2008.

[4] H. Boley. Design Rationale of RuleML: A Markup Language for Semantic Web Rules. In *Semantic Web Working Symposium (SWWS'01)*, pages 381–401, July/August 2001.

[5] H. Boley. POSL: An Integrated Positional-Slotted Language for Semantic Web Knowledge. http://www.ruleml.org/submission/ruleml-shortation.html, May 2004.

[6] H. Boley. Are Your Rules Online? Four Web Rule Essentials. In A. Paschke and Y. Biletskiy, editors, *Proc. Advances in Rule Interchange and Applications, International Symposium (RuleML-2007), Orlando, Florida*, volume 4824 of *LNCS*, pages 7–24. Springer, 2007.

[7] H. Boley. RIF RuleML Rosetta Ring: Round-Tripping the Dlex Subset of Datalog RuleML and RIF-Core. In G. Governatori, J. Hall, and A. Paschke, editors, *RuleML*, volume 5858 of *Lecture Notes in Computer Science*, pages 29–42. Springer, 2009.

[8] H. Boley and M. Kifer. A Guide to the Basic Logic Dialect for Rule Interchange on the Web. *IEEE Transactions on Knowledge and Data Engineering*, Forthcoming 2010.

[9] H. Boley, T. M. Osmun, and B. L. Craig. Social Semantic Rule Sharing and Querying in Wellness Communities. In *4th Asian Semantic Web Conference, ASWC 2009, Shanghai, China*, 2009.

[10] D. Chappell. Enterprise Service Bus. OâĂŹReilly: June 2004, ISBN 0-596-00675-6.

[11] B. L. Craig and H. Boley. Personal Agents in the Rule Responder Architecture. In N. Bassiliades, G. Governatori, and A. Paschke, editors, *RuleML*, volume 5321 of *Lecture Notes in Computer Science*, pages 150–165. Springer, 2008.

[12] G. Eysenbach. Medicine 2.0: Social Networking, Collaboration, Participation, Apomediation, and Openness. JulâĂŞ-Sep 2008.

[13] B. Hughes, I. Joshi, and J. Wareham. Health 2.0 and Medicine 2.0: Tensions and Controversies in the Field. 2008.

[14] IBM. WebSphere Enterprise Service Bus. http://www-01.ibm.com/software/integration/wsesb/.

[15] M. Inc. Mule: A lightweight Java-based Enterprise Service Bus (ESB). http://www.mulesoft.org/what-mule-esb.

[16] A. Kozlenkov, A. Paschke, and M. Schroeder. Prova: A Language for Rule Based Java Scripting, Information Integration, and Agent Programming. http://www.prova.ws/.

[17] J. L. V. Mejino, A. V. Agoncillo, K. L. Rickard, and C. Rosse. Representing Complexity in Part-Whole Relationships within the Foundational Model of Anatomy. http://sig.biostr.washington.edu/projects/fm/FME/.

[18] H. Molyneaux and S. O'Donnell. Patient Portals 2.0: The Potential for Online Videos. In *COACH Conference: e-Health: Leadership in Action*, May/June 2009.

[19] A. Paschke, H. Boley, A. Kozlenkov, and B. Craig. Rule Responder: A RuleML-Based Pragmatic Agent Web for Collaborative Teams and Virtual Organizations. http://www.responder.ruleml.org.

[20] A. Paschke, H. Boley, A. Kozlenkov, and B. Craig. Rule Responder: RuleML-Based Agents for Distributed Collaboration on the Pragmatic Web. In *2nd ACM Pragmatic Web Conference*. ACM, 2007.

[21] J. D. Roo. Euler Sharp (Beta) Inference Engine. http://eulersharp.sourceforge.net/.

[22] L. Sterling and E. Y. Shapiro. *The Art of Prolog - Advanced Programming Techniques, 2nd Ed.* MIT Press, 1994.

[23] M. Walden, H. Mills, and T. Dekkers. Sports Injury Clinic. http://www.sportsinjuryclinic.net/cybertherapist/injurylist.htm.

APPENDIX

A. SIGNATURES AND PARTONOMY

POSL and N3 signatures for the `myDiscussion` predicate to be used by patient support groups[11] as well as the RDFS definitions for the body partonomy[12] are available online.

[11] http://ruleml.org/PatientSupporter/files/PSSignatures.pdf

[12] http://ruleml.org/PatientSupporter/files/PS-Taxonomy.rdf