Expert Querying and Redirection with Rule Responder

FEWS-2007, 12 Nov 2007

Harold Boley[1], Adrian Paschke[2]

[1] National Research Council of Canada
University of New Brunswick, Canada

[2] Bioinformatics, BIOTEC Center,
Technical University Dresden, Germany

Introduction

- Earlier expert finding (discovery) focus:
  Static expert **description** using
  agent-centric metadata profiles
- The current focus:
  Dynamic expert **interaction** using
  agent-communication infrastructure
- Rules employed for semi-automated
  - decision support
  - delegation/coordination
  - negotiation flow
  - reaction logic

Introduction

- Research builds on our previous work in
  - AgentMatcher: Similarity match-making of weighted trees describing (learning) objects
  - FindXpRT (Find an eXpert via Rules and Taxonomies): (RDF or RuleML) FOAF facts and (RuleML) FOAF rules as well as (RDFS) classes for expert finding
  - **Rule Responder: Extending Semantic Web towards a Pragmatic Web infrastructure for distributed rule-based human-computer collaboration**

RuleML FOAF

- Previous system – RDF FOAF:
  Person-centric metadata **facts** only
- FindXpRT contribution – RuleML FOAF:
  Person-centric metadata **rules** added
  - Enables XML-based
    - Rule Formalization
    - Rule Interchange
    - Rule Execution
  - Can derive new FOAF facts
  - Can often be ‘normalized’ into RDF FOAF

RuleML FOAF: Rules

- Make **implicit** properties and relationships **explicit**:

\

```
watchMovie(Peter,?X) :- recommendMovie(Firefly,?X).
```

- Constitute person-centric metadata properties **conditional** on the time, place, …:

\

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

Current Scenario

- An entity that seeks expertise in some area, hence called the ***(expertise) consumer***

consults

- Another entity that mediates, or directly offers, expertise, hence called the ***(expertise) provider***
- Both consumers and providers of expertise can be individuals, organizations, or software systems
- Interested in systems as expertise providers for individuals/organizations as expertise consumers
- Based on Semantic (FOAF) & Pragmatic (Agent) Web

Expertise Providers

Can be

- *Expert finders*
  - Systems that find experts who then solve problems

or

- *Problem solvers*
  - Systems that directly solve problems
- The same system can be an expert finder and a problem solver:

  Duality of expert finding and problem solving

Continuum Between Extremes of Pure Expert Finding and Pure Problem Solving (1)

- In its pure form, expert finding uses a description of human expertise, skills, and traits to search for a matching expert by
  - Broadcasting the description to candidates, comparing it to candidates' self-descriptive (FOAF or SIOC) profiles, or
  - Via a match-maker that mediates between both
- Once an expert is engaged by an organization, he or she starts with problem (analysis and) solving

Continuum Between Extremes of Pure Expert Finding and Pure Problem Solving (2)

- When a consumer performs expert finding, this is part of problem solving in broad sense:
- An expert is sought – e.g. by advertising a generalized problem description – to solve temporary or continuing problems of the consumer
- If the problems can be solved via ‘outsourcing’, the found expert(s) can perform problem solving remotely

Continuum Between Extremes of Pure Expert Finding and Pure Problem Solving (3)

- In some cases, not even the identity of individuals within a team of remote experts will need to be revealed.
  A remote team that solves problems of a consumer can become increasingly ‘virtual’:
- The focus can shift to communication protocols between the problem solving provider and the consumer
- The experts no longer need to form a fixed team or consist of human experts only:

  Problem-solving systems can perform some of the tasks
- The internal structure of the providing system need not be known to the consumer

Continuum Between Extremes of Pure Expert Finding and Pure Problem Solving (4)

- If, for all problems, the consumer has to deal with only one provider without being revealed the expert(s) or system(s) behind it,

  the provider turns into a pure problem solver

Interplay Between Expert Finding and Problem Solving

- Even if a consumer primarily plans to engage an expert, their decision on, say, hiring him or her for a job may be (partially) based on his or her performance when solving a trial problem as part of the selection process
- Conversely, even if a consumer primarily plans to get a problem solved, their trust in a proposed solution may be (partially) based on the reputation of the expert(s) involved in problem solving

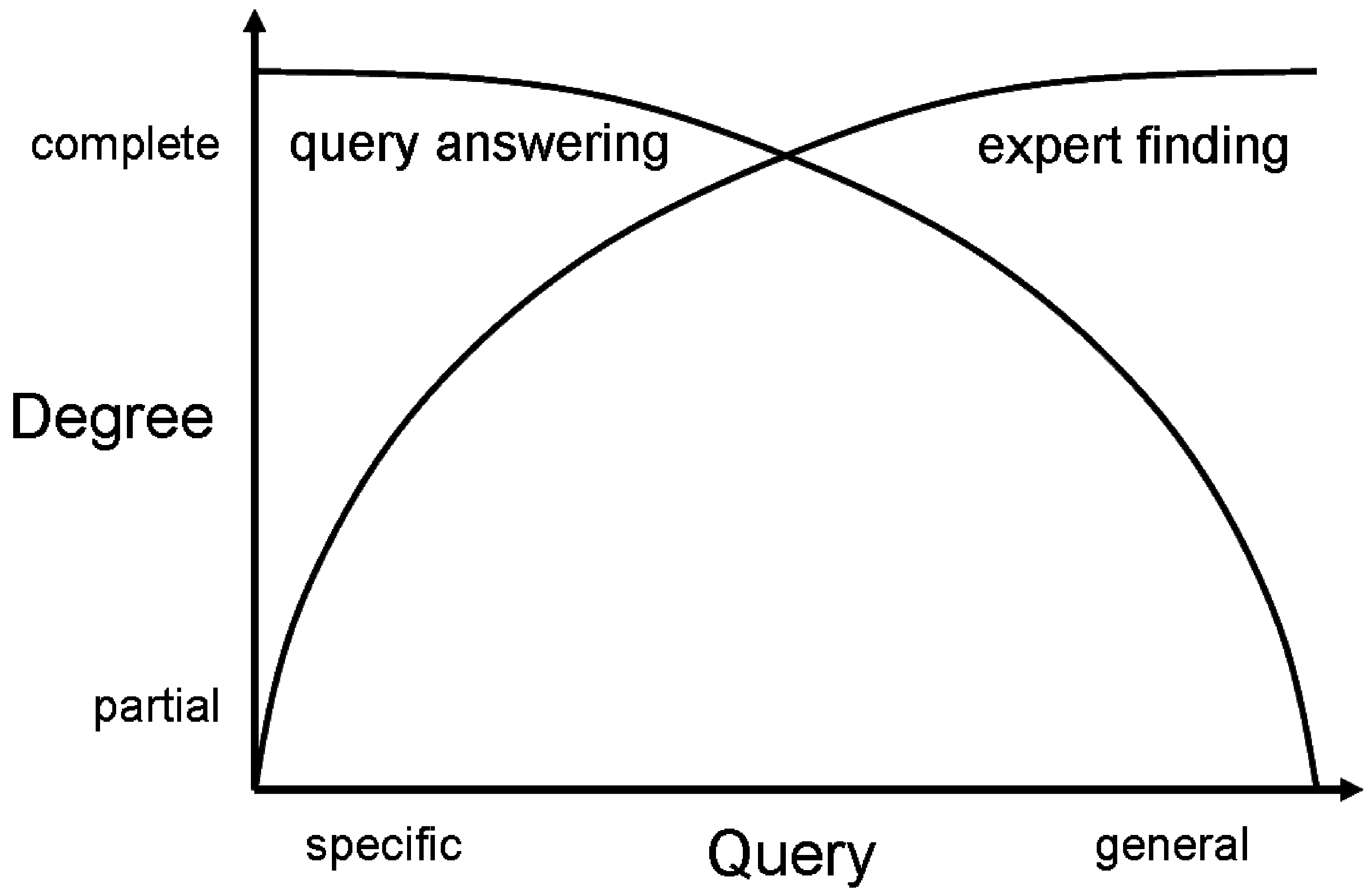
Problem Solving Specialized to Query Answering

- Some problems themselves consist in answering a query, and various other problems, X, are basically solved when the query “How to solve problem X?” is answered
- Therefore, and since query answering is more amenable to current software systems than, say, sensor/effector-intensive problem solving, we will focus on the important special case where problem solving equals query answering

Overlap of Expert Finding and Query Answering (1)

- A **specific query** is normally posed to directly receive an answer from a given expert
(it is too narrow to define an area of expertise)
- A **general query** is normally posed to find an expert in the whole area of expertise it defines
(it is too broad to lend itself to a direct answer)
- *A **typical query** can result in both a (partial) answer and a (partial) identification of the expert(s) involved in finding it*

Overlap of Expert Finding and Query Answering (2)



Horizontal axis: specificity/generality of queries

Vertical axis: degree to which queries are answered by a given expert
degree to which queries are used to find experts

Expert Redirection as
Expert Referral or Expert Delegation

- *Expert referral*: provider refers consumer to other providers inside or outside his/her organization
- *Expert delegation*: provider delegates query to such providers, maybe without telling the consumer

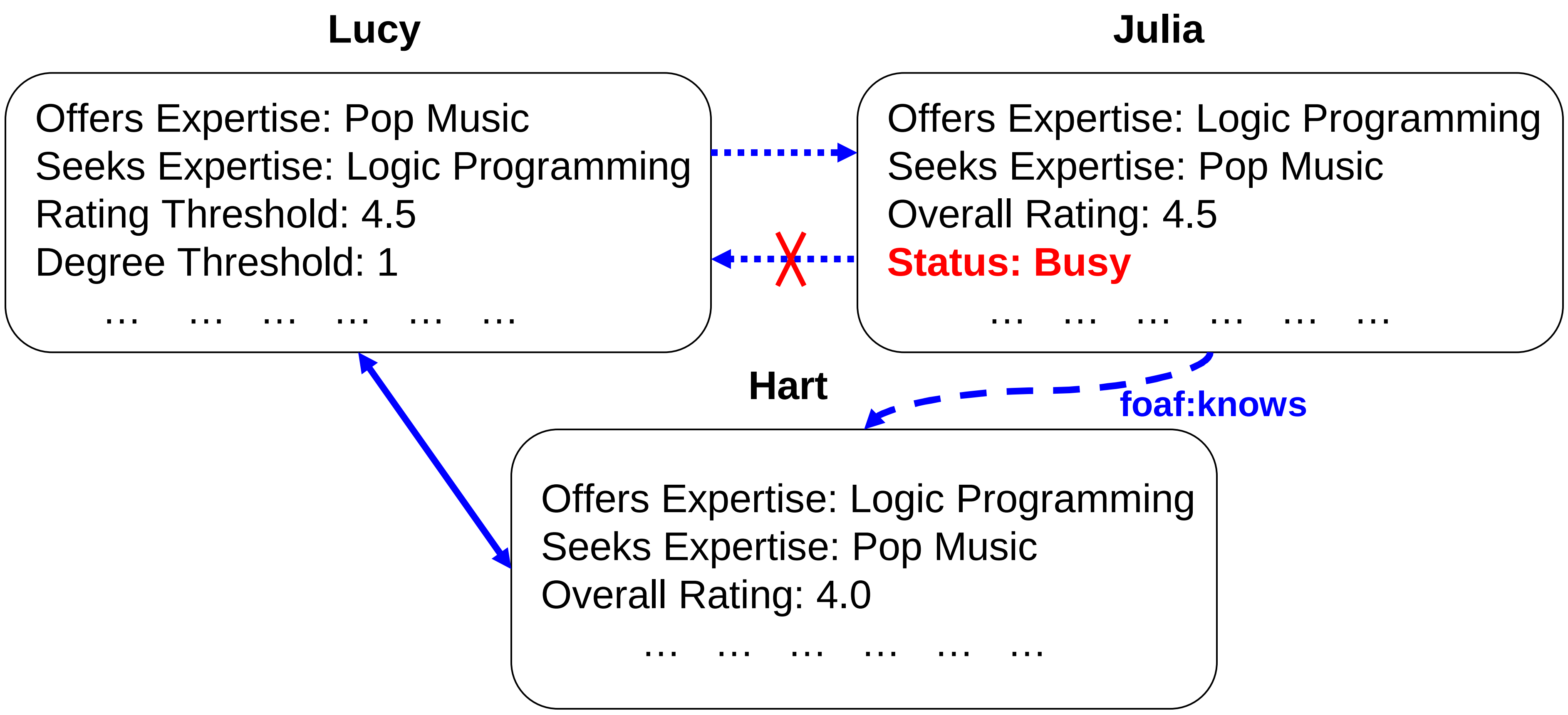
Expert Chaining (1)

- Redirection of an expertise consumer C from an expertise provider Pj to the ‘next best’ provider Pj+1 can be stored *statically* on the profile of Pj, e.g. using the renowned **`foaf:knows`** property
- Can also be computed *dynamically* by Pj, e.g. based on the knowledge Pj has gained so far about the query of consumer C and the partial answer Pj may have given to C
- *Combinations* of static and dynamic redirections are also possible

Expert Chaining (2)

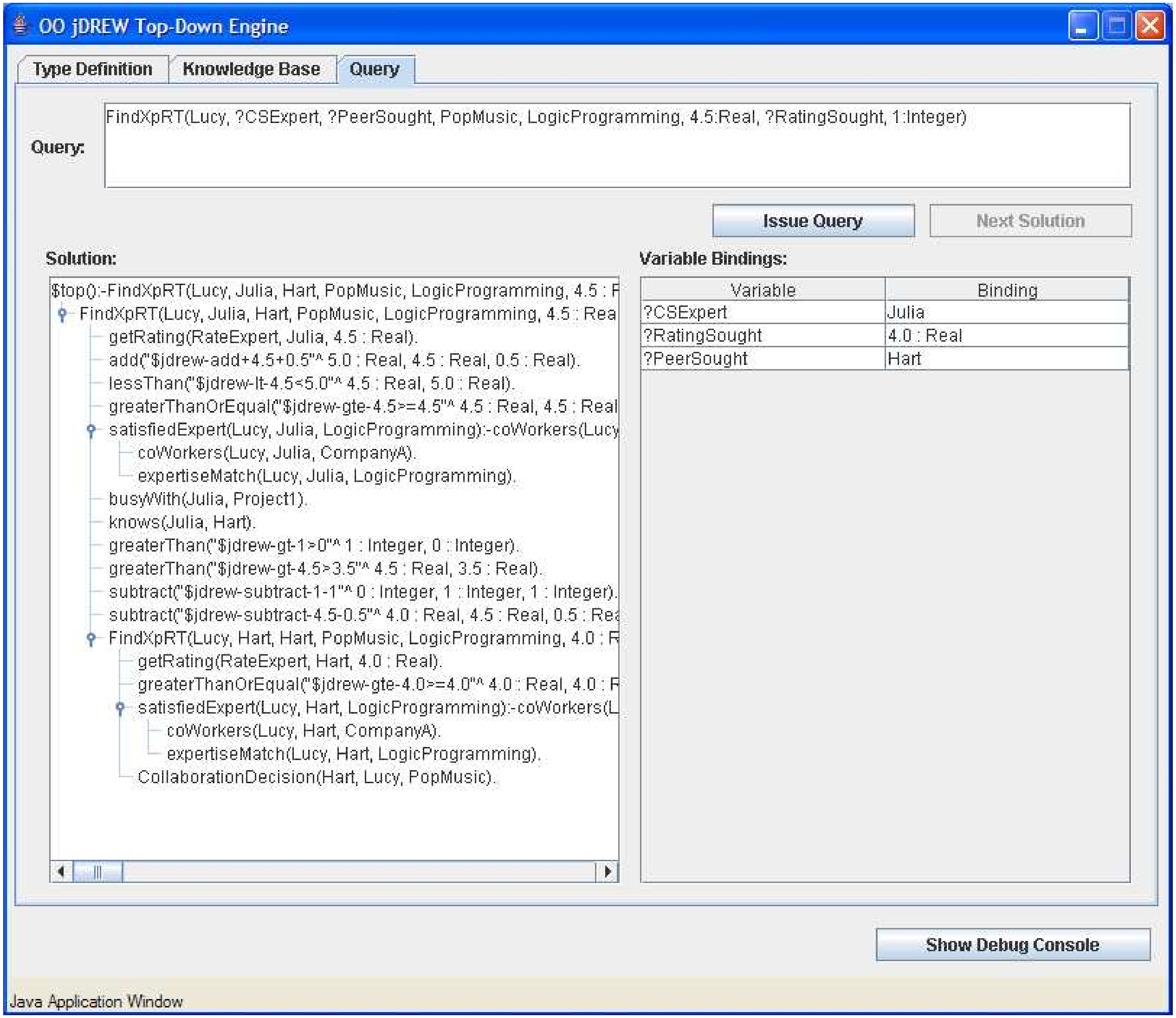
- Leads to chaining, e.g. a *redirection path* P1, P2, ..., Pj, Pj+1, ..., Pn
- In more general case becomes a *redirection tree*
  - with branches from a provider P to k other providers, who
    - besides performing own referrals and/or ‘downward’ delegations
  - give partial answers ‘upward’ to P for answer integration
- In most general case, the redirection tree becomes a general *redirection graph*, where providers perform networking

Find an Expert with Referrals



Lucy
Offers Expertise: Pop Music
Seeks Expertise: Logic Programming
Rating Threshold: 4.5
Degree Threshold: 1
… … … … … …
Julia
Offers Expertise: Logic Programming
Seeks Expertise: Pop Music
Overall Rating: 4.5
Status: Busy
… … … … … …
foaf:knows
Hart
Offers Expertise: Logic Programming
Seeks Expertise: Pop Music
Overall Rating: 4.0
… … … … … …

FindXpRT(Lucy, ?CSXpRT, ?ReferredXpRT,
PopMusic, LogicProgramming, 4.5:Real, ?RatingSought, 1:Integer)



Query:

```
FindXpRT(Lucy, ?CSExpert, ?PeerSought, PopMusic, LogicProgramming, 4.5:Real, ?RatingSought, 1:Integer)
```

Solution:

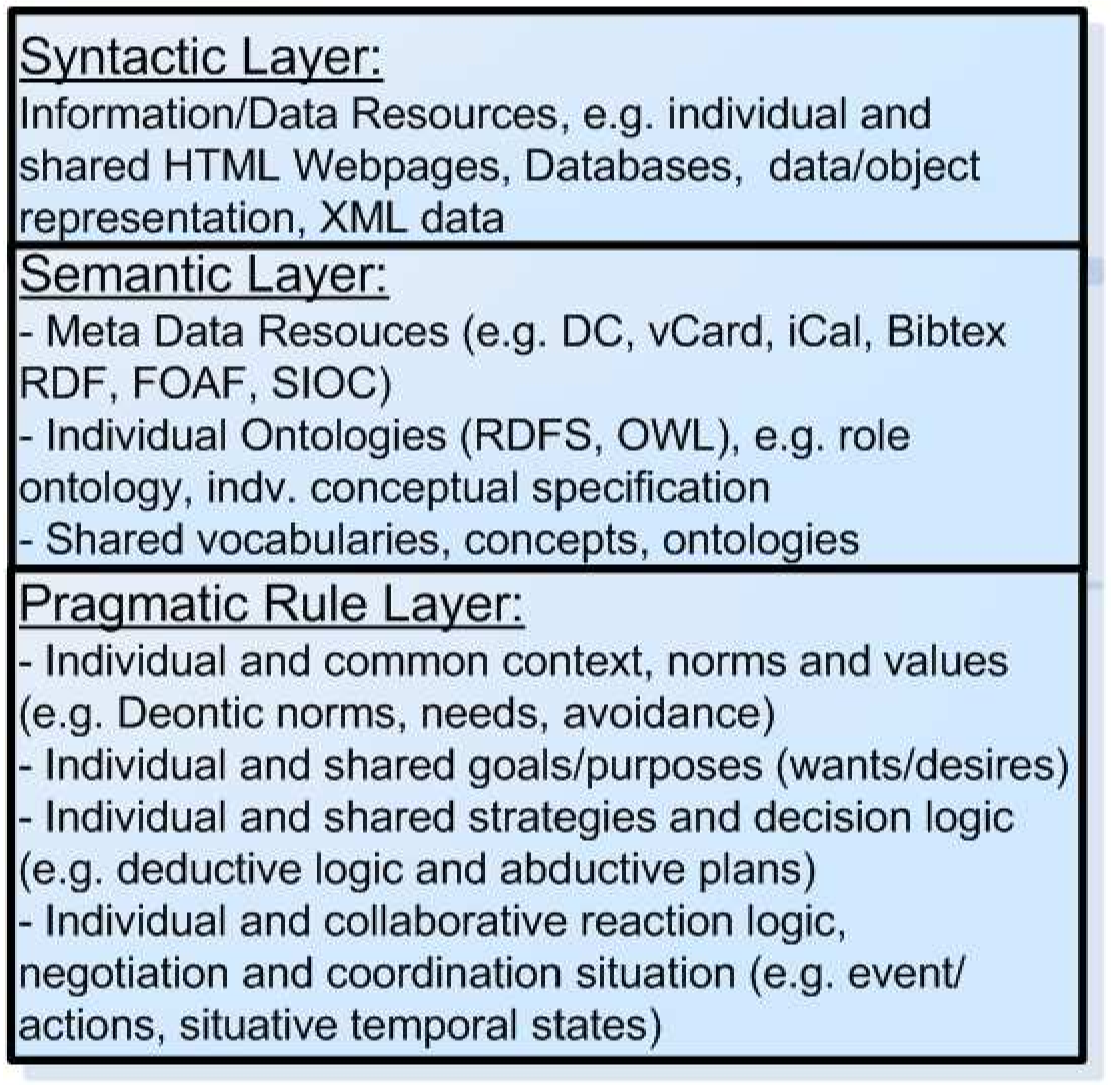
```
$top():-FindXpRT(Lucy, Julia, Hart, PopMusic, LogicProgramming, 4.5 : F
  FindXpRT(Lucy, Julia, Hart, PopMusic, LogicProgramming, 4.5 : Rea
    getRating(RateExpert, Julia, 4.5 : Real).
    add("$jdrew-add+4.5+0.5"^ 5.0 : Real, 4.5 : Real, 0.5 : Real).
    lessThan("$jdrew-lt-4.5<5.0"^ 4.5 : Real, 5.0 : Real).
    greaterThanOrEqual("$jdrew-gte-4.5>=4.5"^ 4.5 : Real, 4.5 : Real
    satisfiedExpert(Lucy, Julia, LogicProgramming):-coWorkers(Lucy
      coWorkers(Lucy, Julia, CompanyA).
      expertiseMatch(Lucy, Julia, LogicProgramming).
    busyWith(Julia, Project1).
    knows(Julia, Hart).
    greaterThan("$jdrew-gt-1>0"^ 1 : Integer, 0 : Integer).
    greaterThan("$jdrew-gt-4.5>3.5"^ 4.5 : Real, 3.5 : Real).
    subtract("$jdrew-subtract-1-1"^ 0 : Integer, 1 : Integer, 1 : Integer).
    subtract("$jdrew-subtract-4.5-0.5"^ 4.0 : Real, 4.5 : Real, 0.5 : Rea
    FindXpRT(Lucy, Hart, Hart, PopMusic, LogicProgramming, 4.0 : R
      getRating(RateExpert, Hart, 4.0 : Real).
      greaterThanOrEqual("$jdrew-gte-4.0>=4.0"^ 4.0 : Real, 4.0 : R
      satisfiedExpert(Lucy, Hart, LogicProgramming):-coWorkers(L
        coWorkers(Lucy, Hart, CompanyA).
        expertiseMatch(Lucy, Hart, LogicProgramming).
      CollaborationDecision(Hart, Lucy, PopMusic).
```

Variable Bindings:

Variable	Binding
?CSExpert	Julia
?RatingSought	4.0 : Real
?PeerSought	Hart

Towards a Pragmatic Web

1. Explicit Meta-data
   - vCard, PICS, Dublin Core, RDF, IEEE LOM (Learning Objects Metadata), Micro Formats, FOAF, SIOC …
2. Ontologies
   - RDFS, OWL Lite|DL|Full
3. Logic and Inference
   - e.g. Logic Programming Rule/Inference Engines
4. Software Agents and Web Services
   - FIPA, Semantic Web Services, RBSLA, …

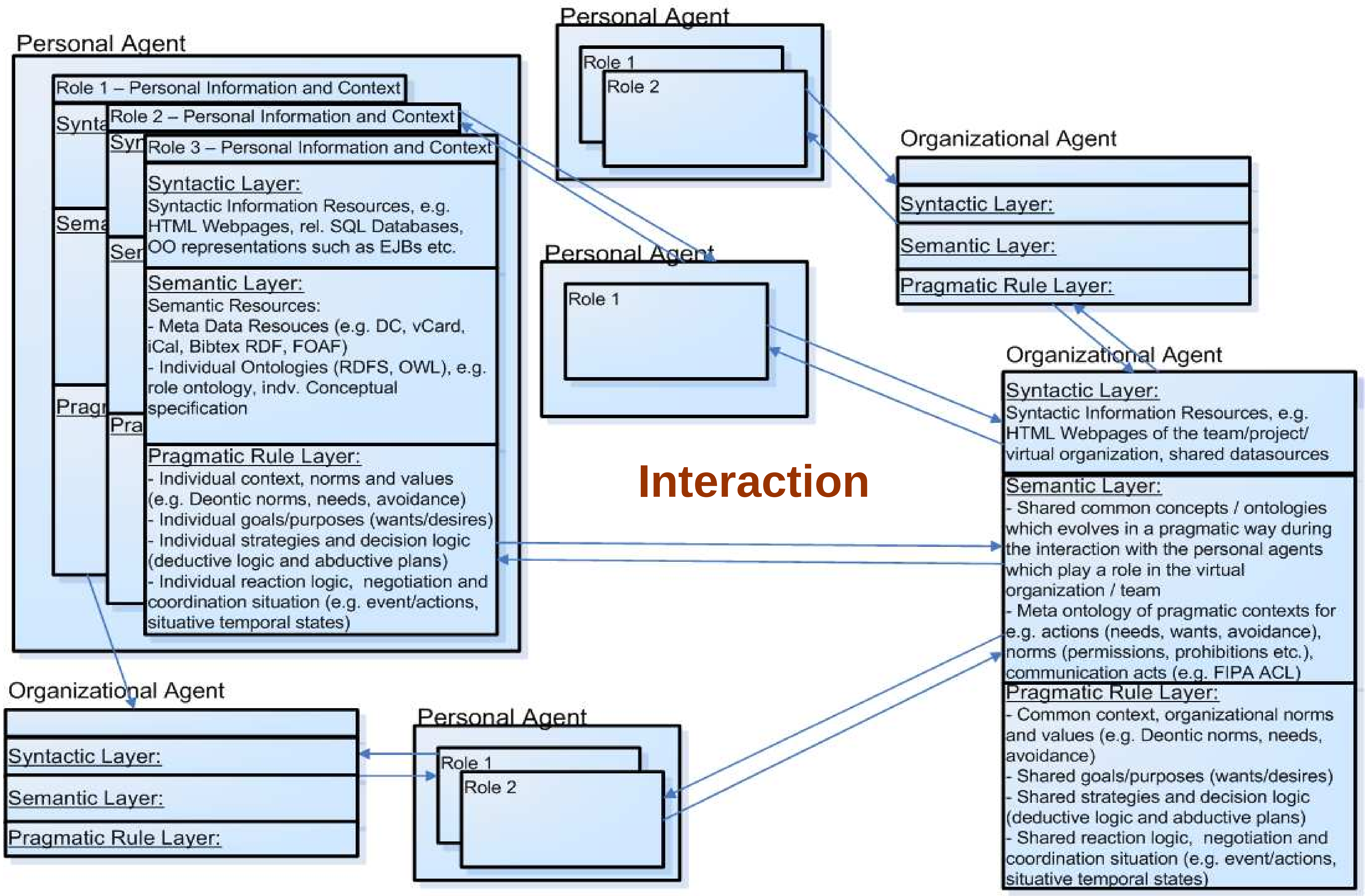


Syntactic Layer:
Information/Data Resources, e.g. individual and shared HTML Webpages, Databases, data/object representation, XML data

Semantic Layer:
- Meta Data Resouces (e.g. DC, vCard, iCal, Bibtex RDF, FOAF, SIOC)
- Individual Ontologies (RDFS, OWL), e.g. role ontology, indv. conceptual specification
- Shared vocabularies, concepts, ontologies

Pragmatic Rule Layer:
- Individual and common context, norms and values (e.g. Deontic norms, needs, avoidance)
- Individual and shared goals/purposes (wants/desires)
- Individual and shared strategies and decision logic (e.g. deductive logic and abductive plans)
- Individual and collaborative reaction logic, negotiation and coordination situation (e.g. event/ actions, situative temporal states)

Pragmatic Web with Two Kinds of Collaborative Agents



Syntactic Layer:
Syntactic Information Resources, e.g. HTML Webpages, rel. SQL Databases, OO representations such as EJBs etc.

Semantic Layer:
Semantic Resources:
- Meta Data Resouces (e.g. DC, vCard, iCal, Bibtex RDF, FOAF)
- Individual Ontologies (RDFS, OWL), e.g. role ontology, indv. Conceptual specification

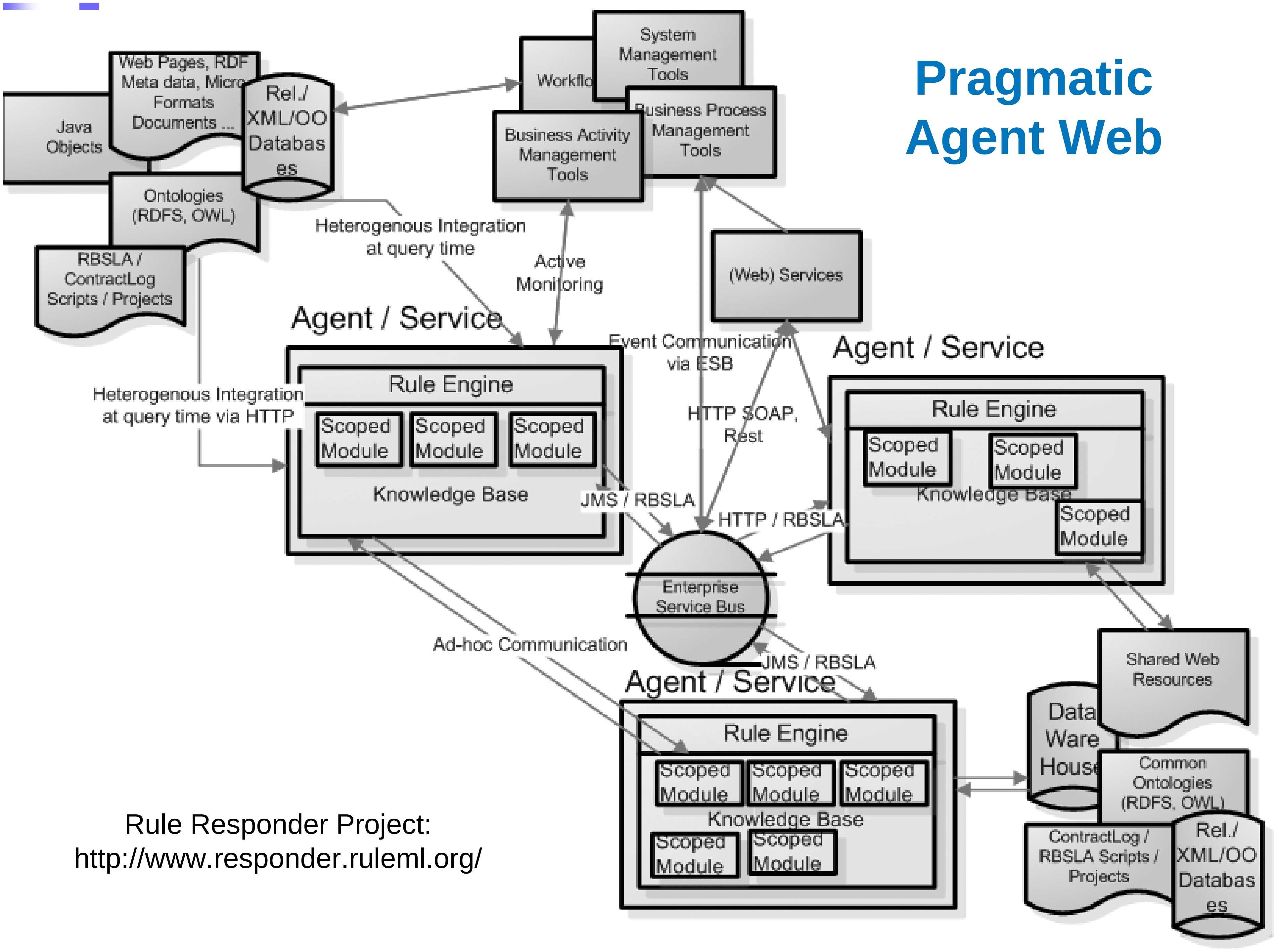
Pragmatic Rule Layer:
- Individual context, norms and values (e.g. Deontic norms, needs, avoidance)
- Individual goals/purposes (wants/desires)
- Individual strategies and decision logic (deductive logic and abductive plans)
- Individual reaction logic, negotiation and coordination situation (e.g. event/actions, situative temporal states)

Syntactic Layer:
Syntactic Information Resources, e.g. HTML Webpages of the team/project/ virtual organization, shared datasources

Semantic Layer:
- Shared common concepts / ontologies which evolves in a pragmatic way during the interaction with the personal agents which play a role in the virtual organization / team
- Meta ontology of pragmatic contexts for e.g. actions (needs, wants, avoidance), norms (permissions, prohibitions etc.), communication acts (e.g. FIPA ACL)

Pragmatic Rule Layer:
- Common context, organizational norms and values (e.g. Deontic norms, needs, avoidance)
- Shared goals/purposes (wants/desires)
- Shared strategies and decision logic (deductive logic and abductive plans)
- Shared reaction logic, negotiation and coordination situation (e.g. event/actions, situative temporal states)

Pragmatic Agent Web



Java Objects
Web Pages, RDF Meta data, Micro Formats Documents ...
Rel./ XML/OO Databases
Ontologies (RDFS, OWL)
RBSLA / ContractLog Scripts / Projects
Workflo
System Management Tools
Business Process Management Tools
Business Activity Management Tools
Heterogenous Integration at query time
Active Monitoring
(Web) Services
Agent / Service
Rule Engine
Scoped Module
Knowledge Base
Event Communication via ESB
HTTP SOAP, Rest
Heterogenous Integration at query time via HTTP
JMS / RBSLA
HTTP / RBSLA
Enterprise Service Bus
Ad-hoc Communication
Shared Web Resources
Data Ware House
Common Ontologies (RDFS, OWL)
ContractLog / RBSLA Scripts / Projects

Rule Responder Project:
http://www.responder.ruleml.org/

Benefits of Rule-Based Decision and Reaction Logic

1. Compact declarative representation of rules
   - Clear semantics
   - Global rules which might apply in several contexts (reusability)
   - Separation of contract rules from the application code
   - Extensibility of the rule base (without changing the interpreter)
2. Efficient, generic interpreters (rule engines) for automated rule chaining and execution of reaction rules
3. Automated conflict detection of rule conflicts
   - Traceable and verifiable results
   - Integrity constraints are possible
   - Automated conflict resolution by rule prioritization

Rules play an important role to automatically transform contextual data, derive new conclusions and decisions from existing knowledge and behaviourally act according to changed conditions or occurred events

Rule Responder Architecture (MDA)

1. **Computational independent model (CIM)** with rules, processes, conversational flows (e.g. in a natural or visual language)
2. **Platform independent model (PIM)** which represents the rules, events and ontologies in a common (standardized) interchange format (e.g. a markup language)
3. **Platform specific model (PSM)** which encodes the rule statements in the language of a specific execution environment (e.g. a rule engine / inference service or compiled code)

General Syntax for Reaction Rules (Reaction RuleML 0.2)

```
<Rule style="active" evaluation="strong">
    <label> <!-- metadata --> </label>
    <scope> <!-- scope --> </scope>
    <qualification> <!-- qualifications --> </qualification>
    <oid> <!-- object identifier --> </oid>
    <on> <!-- event --> </on>
    <if> <!-- condition --> </if>
    <then> <!-- conclusion --> </then>
    <do> <!-- action --> </do>
    <after> <!-- postcondition --> </after>
    <else> <!-- else conclusion --> </else>
    <elseDo> <!-- else/alternative action --> </elseDo>
    <elseAfter> <!-- else postcondition --> </elseAfter>
</Rule>
```

Messages in Reaction RuleML

```
<Message mode="outbound" directive="ACL:inform">
      <oid> <!-- conversation ID--> </oid>
      <protocol> <!-- transport protocol --> </protocol>
      <sender> <!-- sender agent/service --> </sender>
      <content> <!-- message payload --> </content>
</Message>
```

- **@mode** = inbound|outbound – attribute defining the type of a message
- **@directive** – attribute defining the pragmatic context of the message, e.g. one or more FIPA ACL performatives, KQML, OWL-QL, Standard Deontic Logic norms, …
- **< oid >** – the conversation id used to distinguish multiple conversations and conversation states
- **< protocol >** – a transport protocol such as HTTP, JMS, SOAP, Jade, Enterprise Service Bus (ESB) ...
- **< sender >< receiver >** – the sender/receiver agent/service of the message
- **< content >** – message payload transporting a RuleML / Reaction RuleML query, answer or rule base

Example: Request / Query

```
...
<Message mode="outbound" directive="ACL:query-ref">
    <oid> <Ind>RuleML-2007</Ind> </oid>
    <protocol> <Ind>esb</Ind> </protocol>
    <sender> <Ind>User</Ind> </sender>
    <content>
      <Atom>
            <Rel>getContact</Rel>
            <Ind>Website</Ind>
            <Var>Contact</Var>
      </Atom>
    </content>
</Message>
...
```

FIPA ACL directive

- Message is local to the conversation state (oid) and pragmatic context (directive)

Example: Response / Answer

```
<Message …>
...
  <content>
      <Atom>
            <Rel>getContact</Rel>
            <Ind>Website</Ind>
            <Expr>
               <Fun>person</Fun>
               <Ind>Adrian Paschke</Ind>
               <Ind>Program Co-Chair RuleML-2007</Ind>
               <Ind>Technical University Munich</Ind>
               <Ind>paschke@in.tum.de</Ind>
               <Ind>+49 89 289 17504</Ind>
            </Expr>
      </Atom>
  </content>
...
</Message>
```

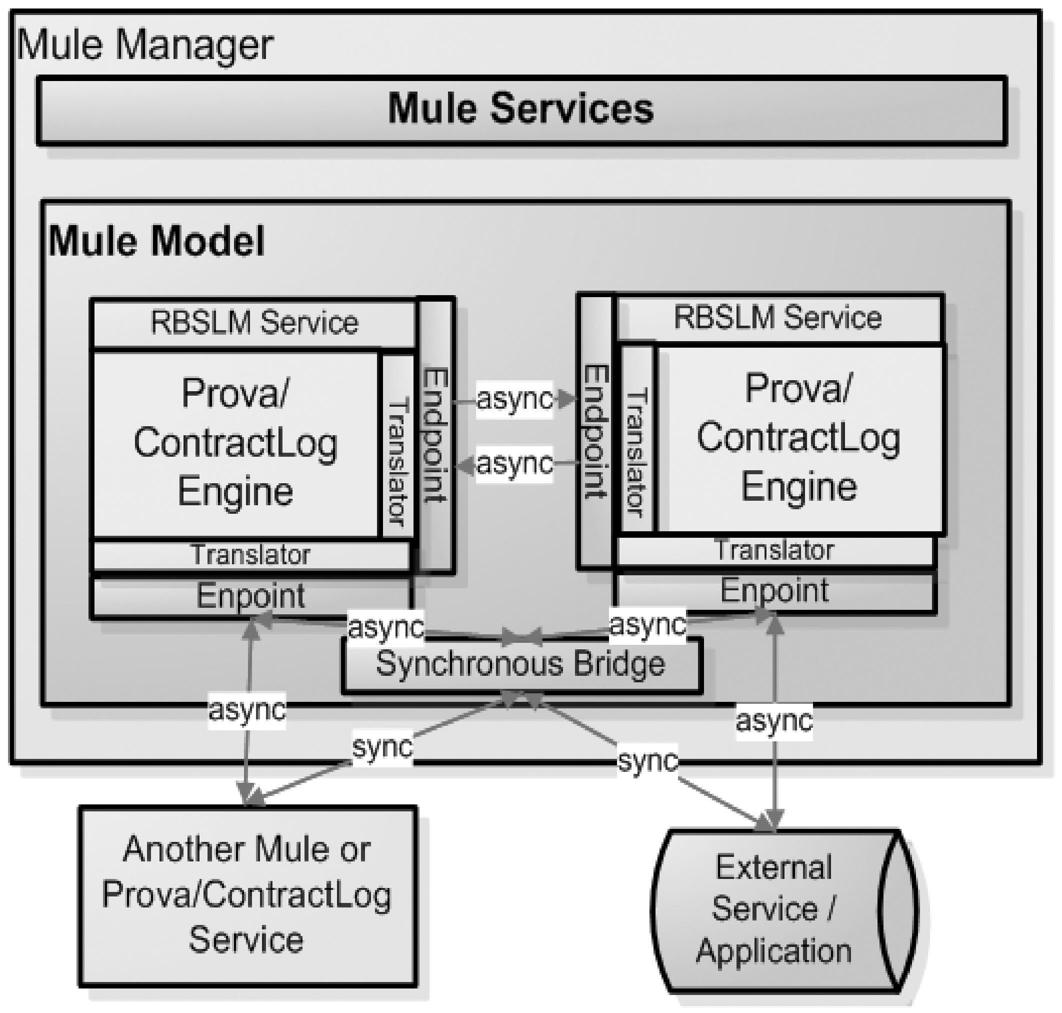
Enterprise Service Bus

Communication Middleware

+

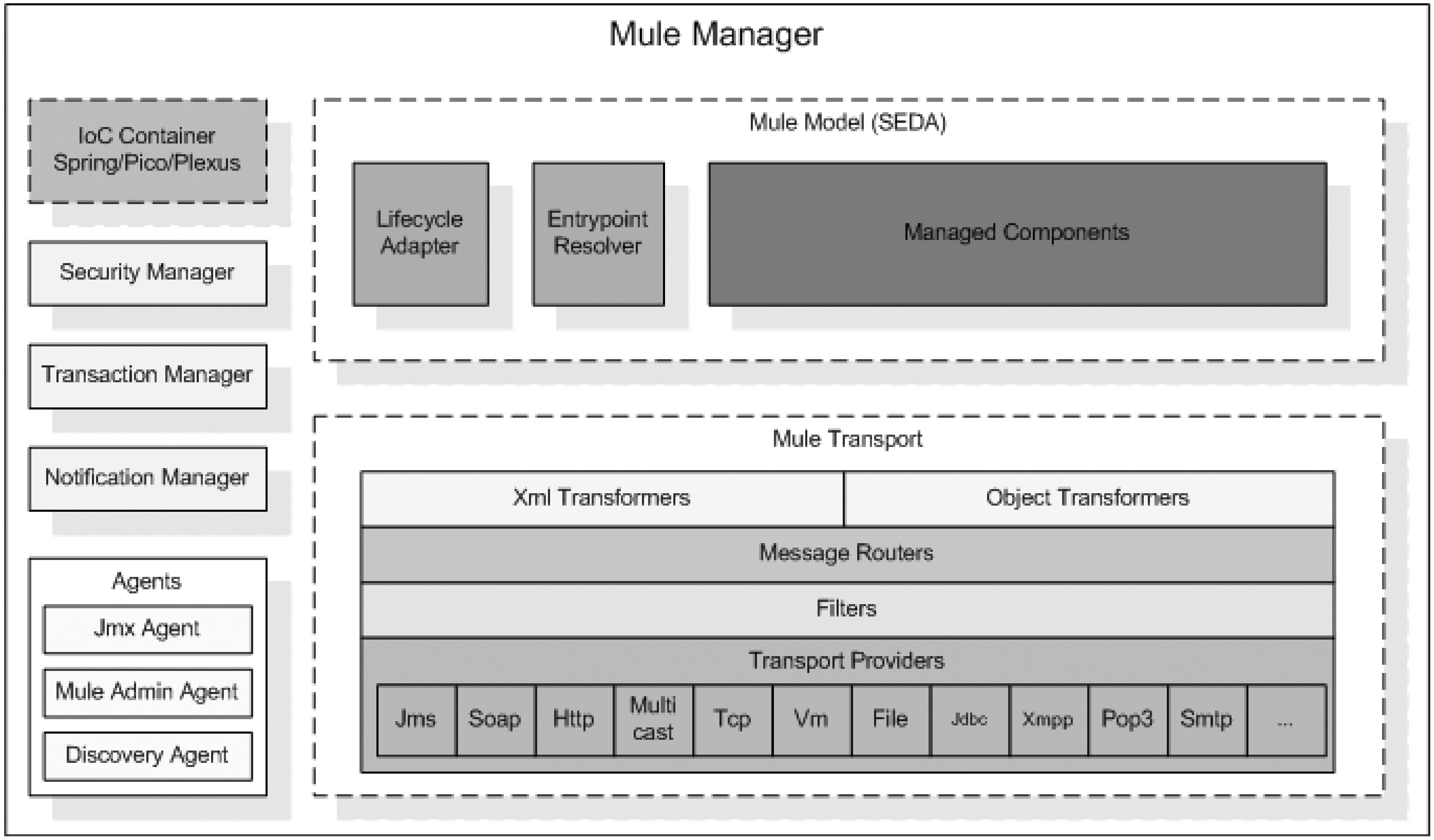
Service Object Broker

Mule Enterprise Service Bus

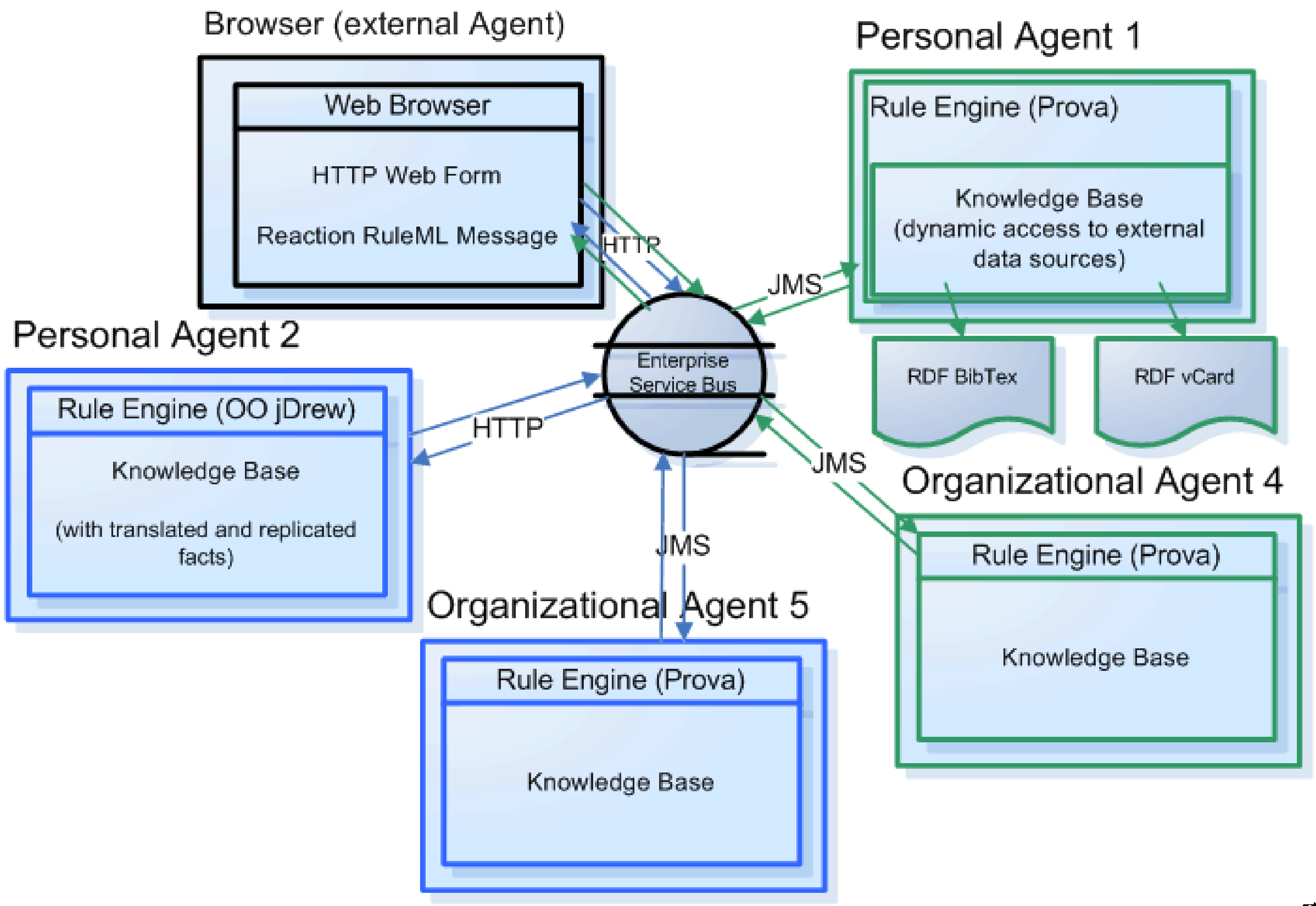


- Mule ESB Open Source
- Message Platform and distributed Object Broker
- Staged Event Driven Architecture (SEDA)
- > 30 Protocols (JMS, HTTP, SOAP …)
- Synchronous and Asynchronous Communication
- Complex Message-driven Event Processing (CEP)

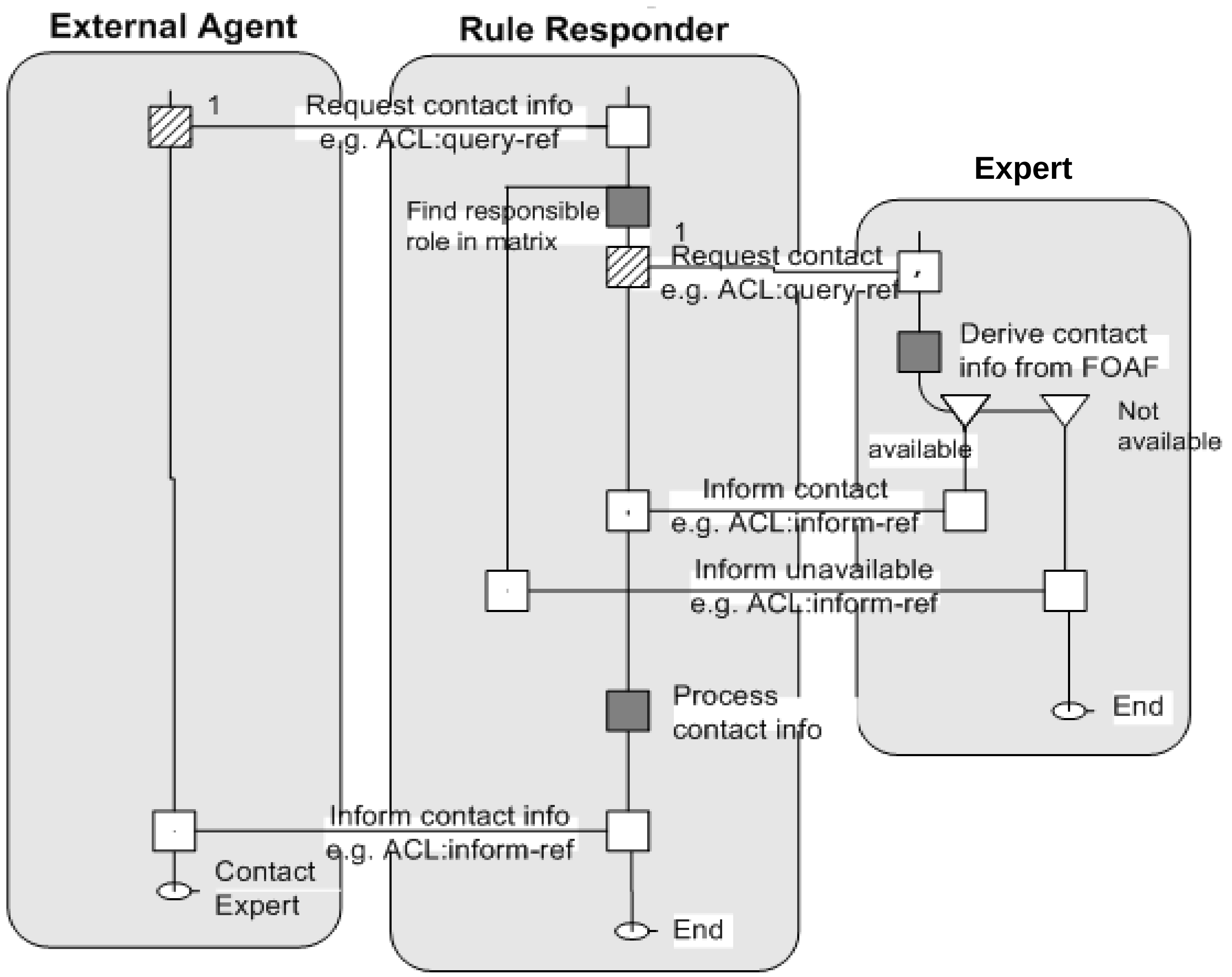
Mule Enterprise Service Bus



Use Case: Expert Finding (1)



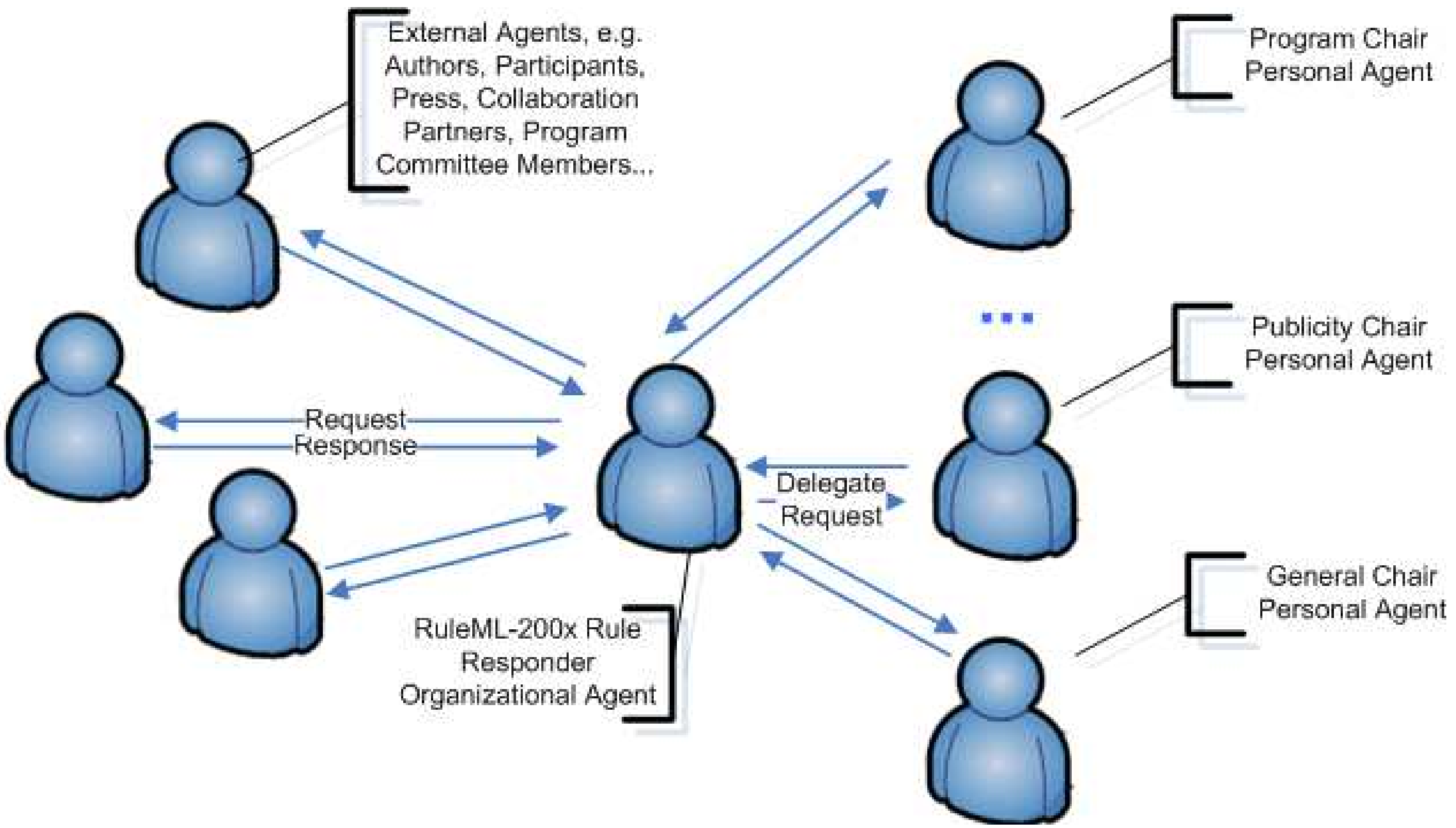
Use Case: Expert Finding (2)

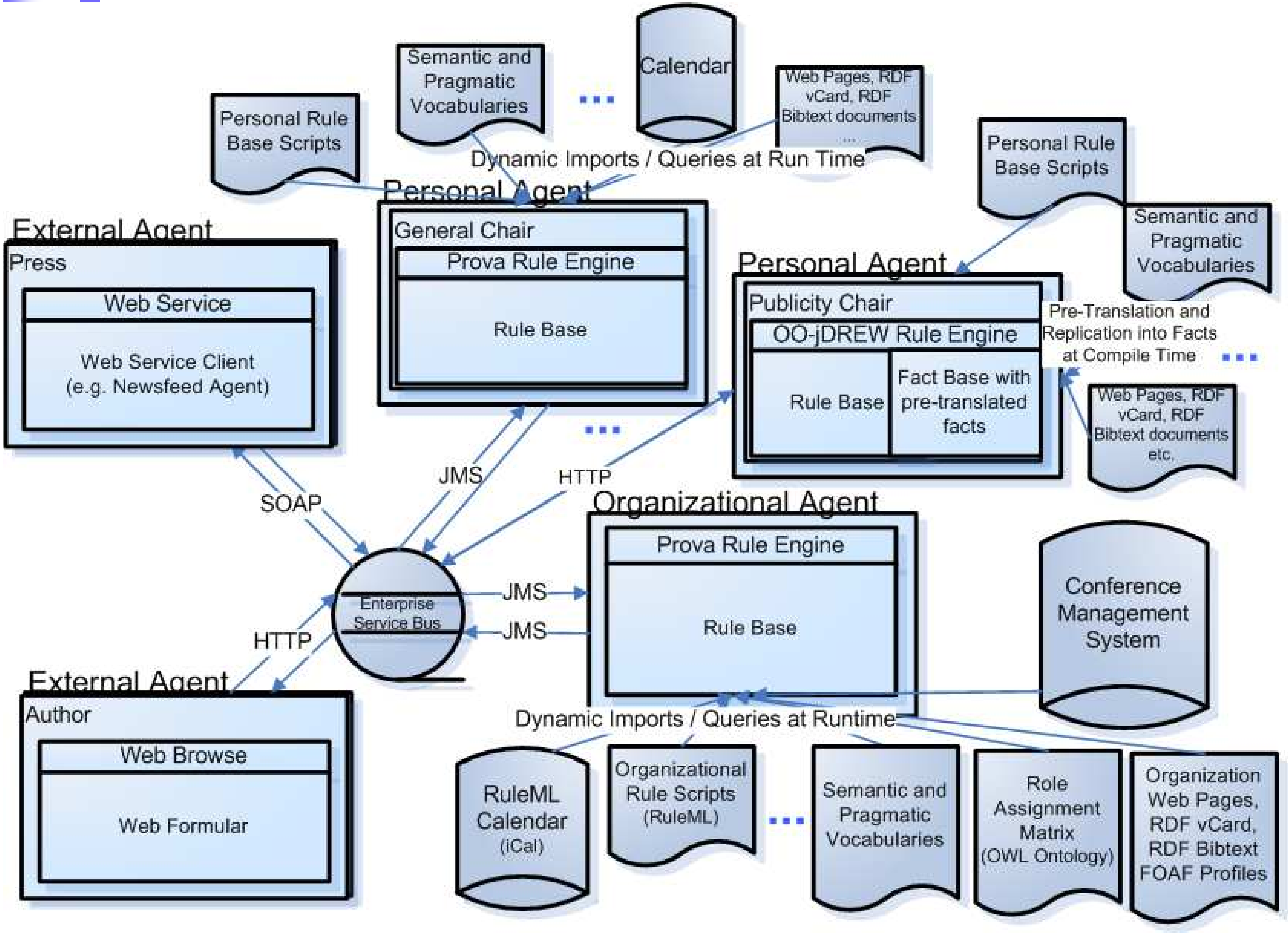


Advanced Performatives

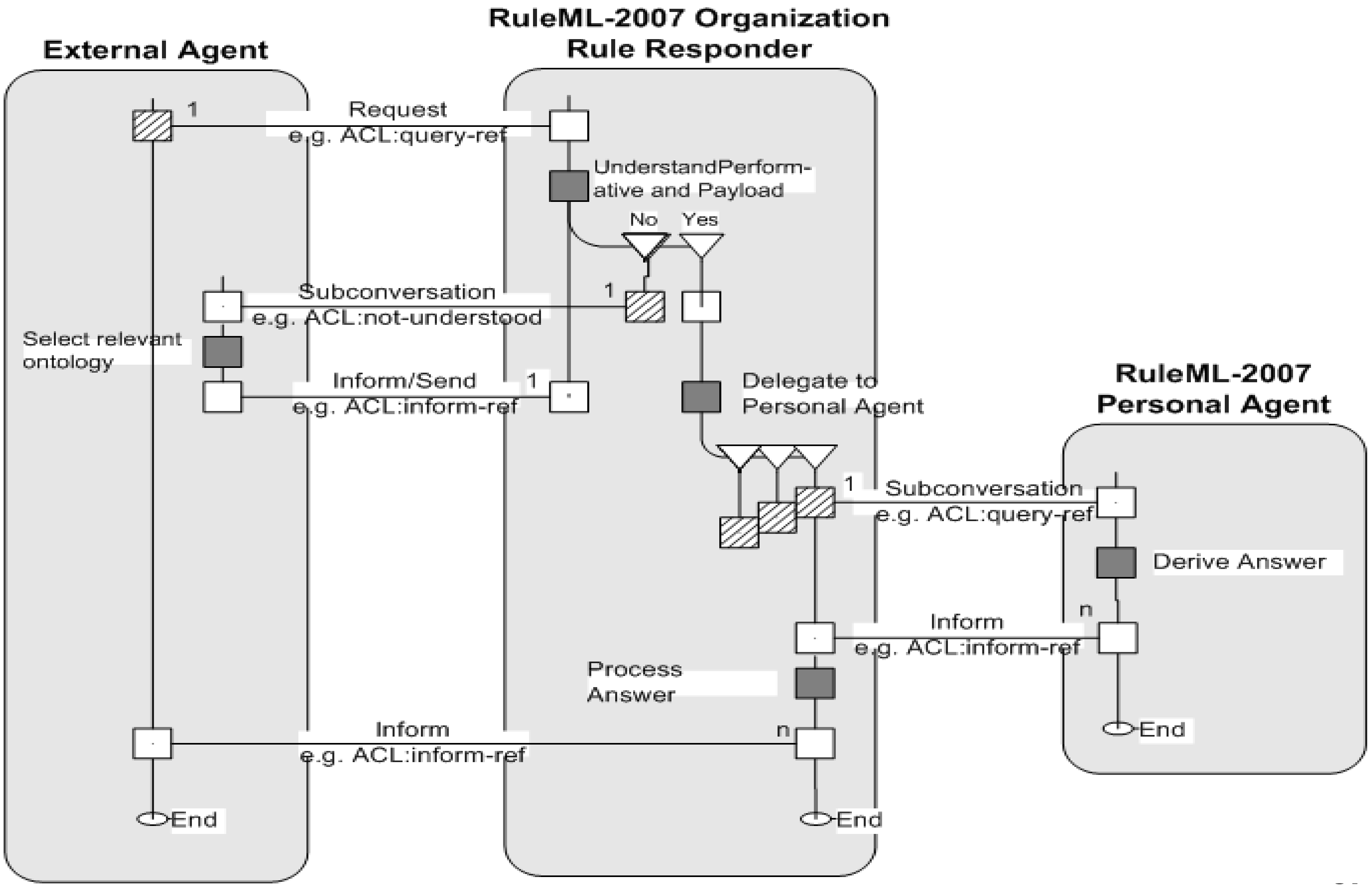
- **Query Answering**: Should the query be answered directly, only be used for expert finding, or both?
- **Expert Identification**: Do experts answering a query inside or outside the provider's organization need to be identified to the consumer?
- **Query Confidentiality**: Must the query be treated as confidential inside or outside the provider's organization?
- **Expert Referral**: Should the provider refer the consumer to other providers inside or outside his/her organization if no (complete) answer can be found (the consumer himself/herself would then need to contact that next provider, perhaps showing them the referral message from the previous provider)?
- **Expert Delegation**: Is the provider entitled to delegate the query (perhaps anonymized?) to other providers inside or outside his/her organization?
- **Delegation Chains**: Should delegation be stopped after a maximum length of delegation chains is reached (special cases: no delegation, 1-step delegation)?
- **Query Decomposition**: Should the provider decompose the query into subqueries if it cannot be answered or delegated as a whole?
- **Answer Returning**: Should providers to whom a (sub)query was delegated return answers to the delegating provider, to the consumer, or to both?
- **Answer Integration**: Should the provider integrate answers returned to it for subqueries or just collect them and hand them back unchanged to the consumer?

Use Case: Symposium Question-Answering (by its Virtual Organization)


External Agents, e.g. Authors, Participants, Press, Collaboration Partners, Program Committee Members...
Request
Response
RuleML-200x Rule Responder Organizational Agent
Delegate
Request
Program Chair Personal Agent
...
Publicity Chair Personal Agent
General Chair Personal Agent


Personal Rule Base Scripts
Semantic and Pragmatic Vocabularies
...
Calendar
Web Pages, RDF vCard, RDF Bibtext documents ...
Dynamic Imports / Queries at Run Time
Personal Agent
General Chair
Prova Rule Engine
Rule Base
External Agent
Press
Web Service
Web Service Client (e.g. Newsfeed Agent)
Personal Rule Base Scripts
Semantic and Pragmatic Vocabularies
Personal Agent
Publicity Chair
OO-jDREW Rule Engine
Rule Base
Fact Base with pre-translated facts
Pre-Translation and Replication into Facts at Compile Time
...
Web Pages, RDF vCard, RDF Bibtext documents etc.
...
SOAP
JMS
HTTP
Organizational Agent
Prova Rule Engine
Rule Base
Enterprise Service Bus
JMS
JMS
HTTP
External Agent
Author
Web Browse
Web Formular
Conference Management System
Dynamic Imports / Queries at Runtime
RuleML Calendar (iCal)
Organizational Rule Scripts (RuleML)
...
Semantic and Pragmatic Vocabularies
Role Assignment Matrix (OWL Ontology)
Organization Web Pages, RDF vCard, RDF Bibtext FOAF Profiles

Use Case: Symposium Question-Answering (by its Virtual Organization)



Example: Request Registration Fee (1)

Query from External Agent (e.g. Web form)

```
<RuleML ...>
   <Message mode="outbound" directive="query">
      ...
     <content>
           <Atom> <Rel>computeFee</Rel>
                        <Ind>Peter Pan</Ind>
                        <Var>Fee</Var>
           </Atom>
     </content>
   </Message>
</RuleML>
```

Delegation Logic of Organizational Agent (Prova)

```
fee(Phase,Fee):-
  % look-up responsible agent
  assigned(Agent,
           ruleml2007_Submission, ruleml2007_responsible),
  % query permission from responsible agent
  sendMsg(XID,esb,Agent, "query", fee(Phase,Fee)),
  % receive answers multiple times
  rcvMult(XID,esb,Agent,"answer", fee(Phase,Fee)).
```

Example: Request Registration Fee (2)

Rule of Program Chair Agent: (Prova)

```
computeFee(Participant,Fee,T):-
   holdsAt(open(RegistrationPhase),T),
   fee(RegistrationPhase,BasicFee),
   sendMsg(XID,esb,ruleml2007_PublicityChair,query,
   registrationDiscount(Participant,BasicFee,1,Discount,Cost)),
   rcvMsg(XID,esb,ruleml2007_PublicityChair,answer,
   registrationDiscount(Participant,BasicFee,1,Discount,Cost)),
   Fee = Cost - (Cost * Discount).

   fee(earlyRegistration,currency_Dollar:500).
   fee(regularRegistration,currency_Dollar:600).
```

Rule of Publicity Chair Agent: (OO jDREW)

```
%This rule is used to calculate the discount of registration if
   % the organization is a collaboration partner.
   registrationDiscount(?Organization:collaborationPartner,?Regi
   strationCost:real,?NumberOfRegistrations:integer,?IndividualD
   iscountCost:real,?TotalCost:real) :-
     multiply(?IndividualDiscountCost:real,
              ?RegistrationCost:real, 0.9:real),
     multiply(?TotalCost:real,
             ?NumberOfRegistrations:integer,
             ?IndividualDiscountCost:real).
```

Conclusions (1)

- Explained duality of expert finding and query answering, and introduced new approach to both
- Rule Responder was employed as the Web-based communication infrastructure for expert querying and redirection
- As our use case, we discussed the organization of the RuleML-2007 Symposium

Conclusions (2)

Future work includes

- Formal foundations of expert finding (e.g., expert and conversation id's as π-calculus channel names)
- Refinement of the system of performatives supporting Rule Responder expert finding
- Further development of Reaction RuleML for serializing the required event and rule types
- Continuation of the RuleML-2007 use case and exploration of similar ones