REASONING AND PROOFING SERVICES FOR SEMANTIC WEB AGENTS

Kalliopi Kravari[1], Konstantinos Papatheodorou[2],
Grigoris Antoniou[2] and Nick Bassiliades[1]

[1]Dept. of Informatics, Aristotle University of Thessaloniki, Greece
[2]Institute of Computer Science, FORTH, Greece
Department of Computer Science, University of Crete, Greece

Overview

- **Motivation**
- *Standard rule/logic language use*
- *Proposal*
- *Semantics*
- *Proofs*
- *Proof Validation*
- *Our appoach*
    - o ***EMERALD***
    - o ***Reasoners***
    - o ***DR-Prolog Rasoner***
- ***Conclusions – Future Work***

Motivation

SW brings interoperability for web information systems

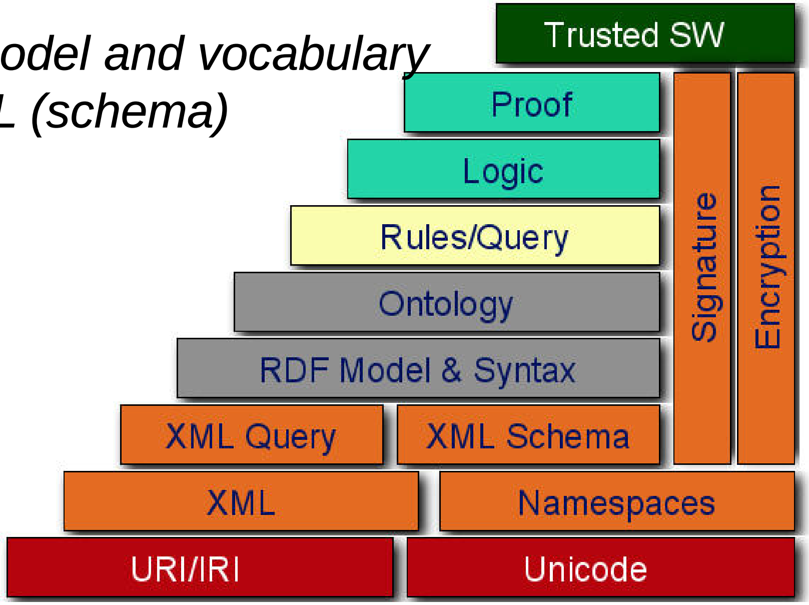
Agents need this interoperability

- *to work seamlessly in the web*
- *to achieve tasks on behalf of the users*

<u>Interoperability is in several levels:</u>

1. ***Syntactic – use** of common language formats*
   e.g. XML (parsers)

2. ***Semantic – use** of common data model and vocabulary*
   e.g. RDF (data), OWL (schema)



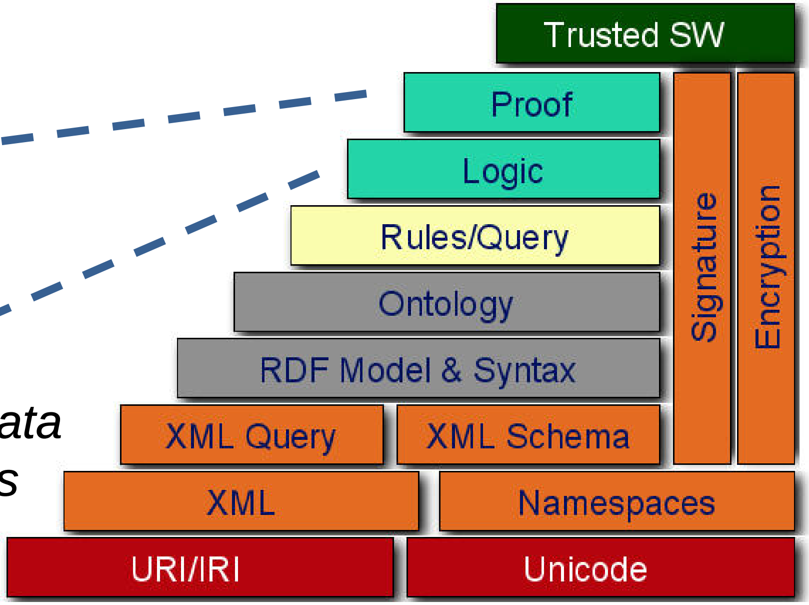
Motivation

These interoperability solutions are
→ more or less stable in the currents SW standards

The next level of interoperability needed is in Logic and in Proof

explanations why
certain conclusions were made

i.e. rules of inference that manipulate data
in order to infer new data or conclusions



Motivation

Logic / rules *have a new standard* ***(RIF)***

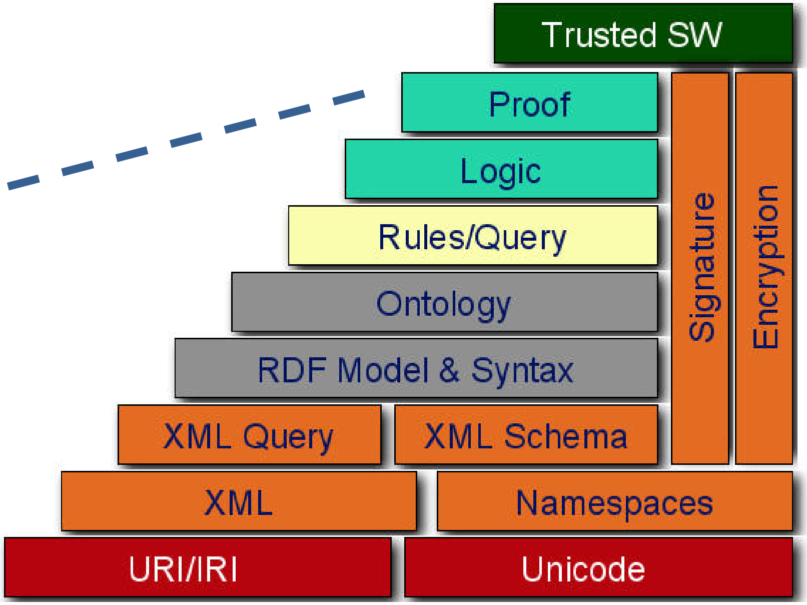
→ but its adoption is not as wide as OWL, RDF, XML

WHY?

Rules Standards DEFINE not a single language

BUT a family of languages

- *Share same syntax*
- *Differ in semantics (entirely/ partially)*

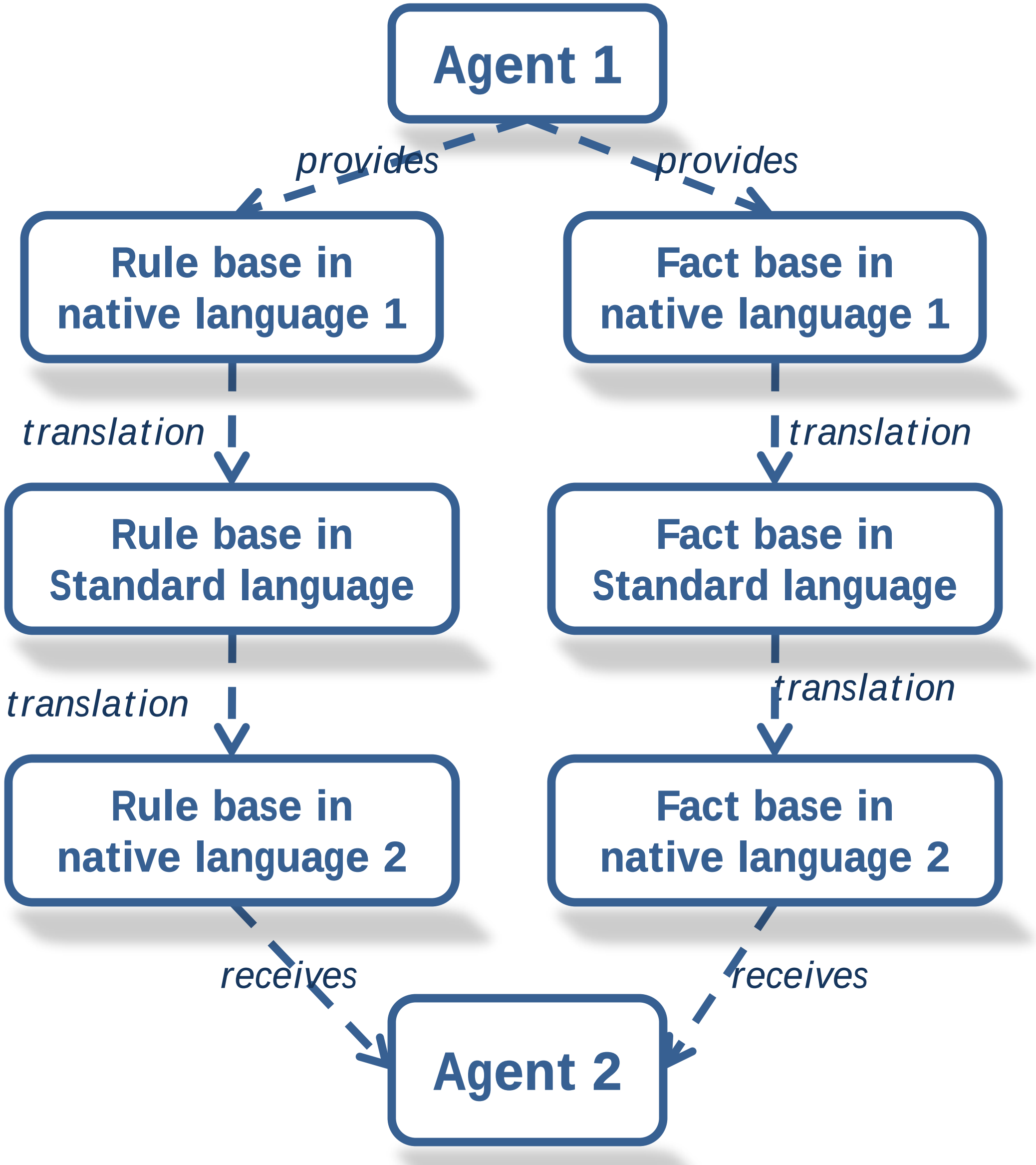


In the Proof layer things are even more primitive since there is no standard.

Overview

- Motivation
- Standard rule/logic language use
- Proposal
- Semantics
- Proofs
- Proof Validation
- Our appoach
  - EMERALD
  - Reasoners
  - DR-Prolog Rasoner
- Conclusions – Future Work

Standard rule/logic language use


Agent 1
provides
provides
Rule base in native language 1
Fact base in native language 1
translation
translation
Rule base in Standard language
Fact base in Standard language
translation
translation
Rule base in native language 2
Fact base in native language 2
receives
receives
Agent 2

FACT CHAIN:

1. *Usually entire chain in RDF, or*
2. *Translation from-to RDF is more easily achievable*

Overview

- ➢ **Motivation**
- ➢ *Standard rule/logic language use*
- ➢ *Proposal*
- ➢ *Semantics*
- ➢ *Proofs*
- ➢ *Proof Validation*
- ➢ *Our appoach*
  - o ***EMERALD***
  - o ***Reasoners***
  - o ***DR-Prolog Rasoner***
- ➢ ***Conclusions – Future Work***

Proposal

***complete translation** between rule languages is:*

- *hard to achieve*
- *sometimes not possible at all*

    → *because of the diversity in semantics between rule languages*

Our solution:

→***trusted third-party reasoning services** (in an agent framework)*

- *Responsible for executing the inference on the original rule base*
- *No transformation*

Overview

- ***Motivation***
- *Standard rule/logic language use*
- *Proposal*
- ***Semantics***
- *Proofs*
- *Proof Validation*
- *Our appoach*
  - ***EMERALD***
  - ***Reasoners***
  - ***DR-Prolog Rasoner***
- ***Conclusions – Future Work***

Semantics

But then how the semantics of the exchanged rule base are going to be understood by the receiving agent?

Semantics of a rule base ≡ conclusions that can be derived

↓

sending only the conclusions, the semantics are communicated

Why should agent2 trust that agent1 run the rule base completely?

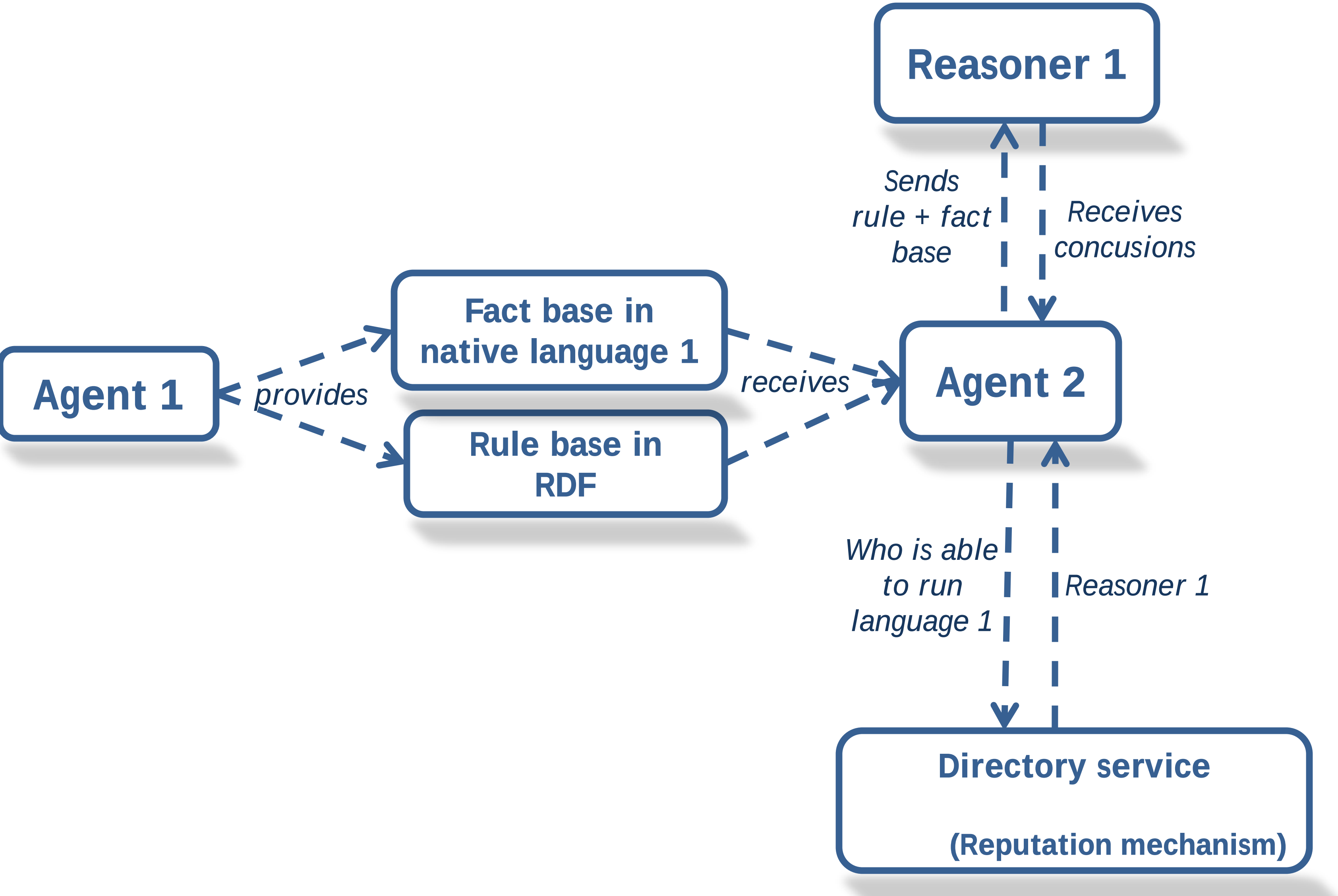
↓

It shouldn't!

it ***can trust***

- *an independent agent with good reputation*
- *supposing it's a "good" reasoning service provider*

Semantics



Reasoner 1

Sends rule + fact base

Receives concusions

Fact base in native language 1

Agent 1

provides

receives

Agent 2

Rule base in RDF

Who is able to run language 1

Reasoner 1

Directory service

(Reputation mechanism)

Overview

- ***Motivation***
- ***Standard rule/logic language use***
- ***Proposal***
- ***Semantics***
- ***Proofs***
- ***Proof Validation***
- ***Our appoach***
  - ***EMERALD***
  - ***Reasoners***
  - ***DR-Prolog Rasoner***
- ***Conclusions – Future Work***

Proofs

Sometimes it is not enough to trust the results

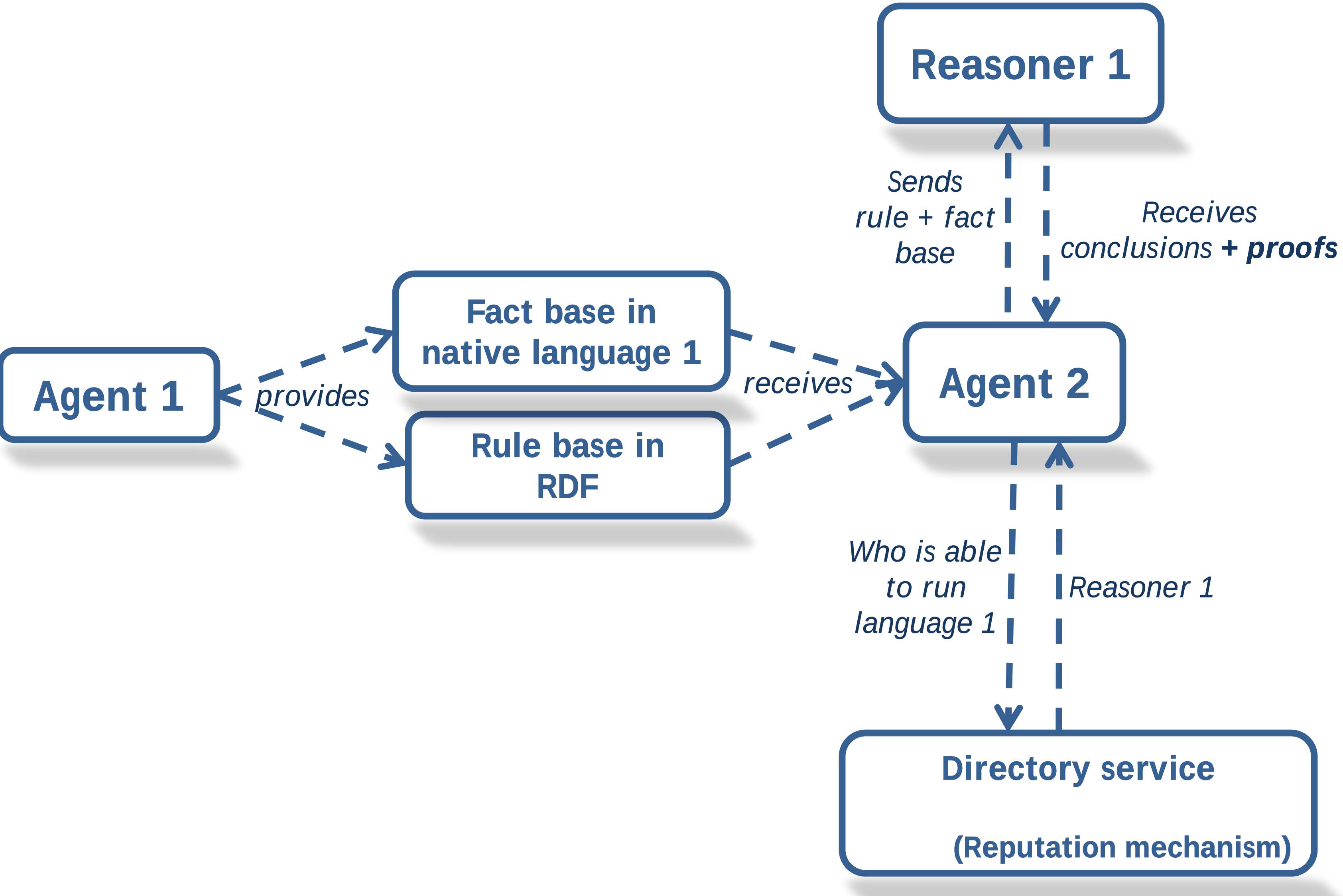
The receiving agent might want:

- *proof that some conclusion is correct*
- *an explanation of how the results was drawn*

- *Verify that the reasoner did a good job*
- *Present the explanations to its human user*

The reasoning service facility needs → a proof generation facility

Proofs



Reasoner 1

Sends rule + fact base

Receives conclusions + **proofs**

Agent 1

provides

Fact base in native language 1

Rule base in RDF

receives

Agent 2

Who is able to run language 1

Reasoner 1

Directory service

(Reputation mechanism)

Overview

- ***Motivation***
- ***Standard rule/logic language use***
- ***Proposal***
- ***Semantics***
- ***Proofs***
- ***Proof Validation***
- ***Our appoach***
  - ***EMERALD***
  - ***Reasoners***
  - ***DR-Prolog Rasoner***
- ***Conclusions – Future Work***

Proof Validation

assist agents in validating the proofs provided by other agents

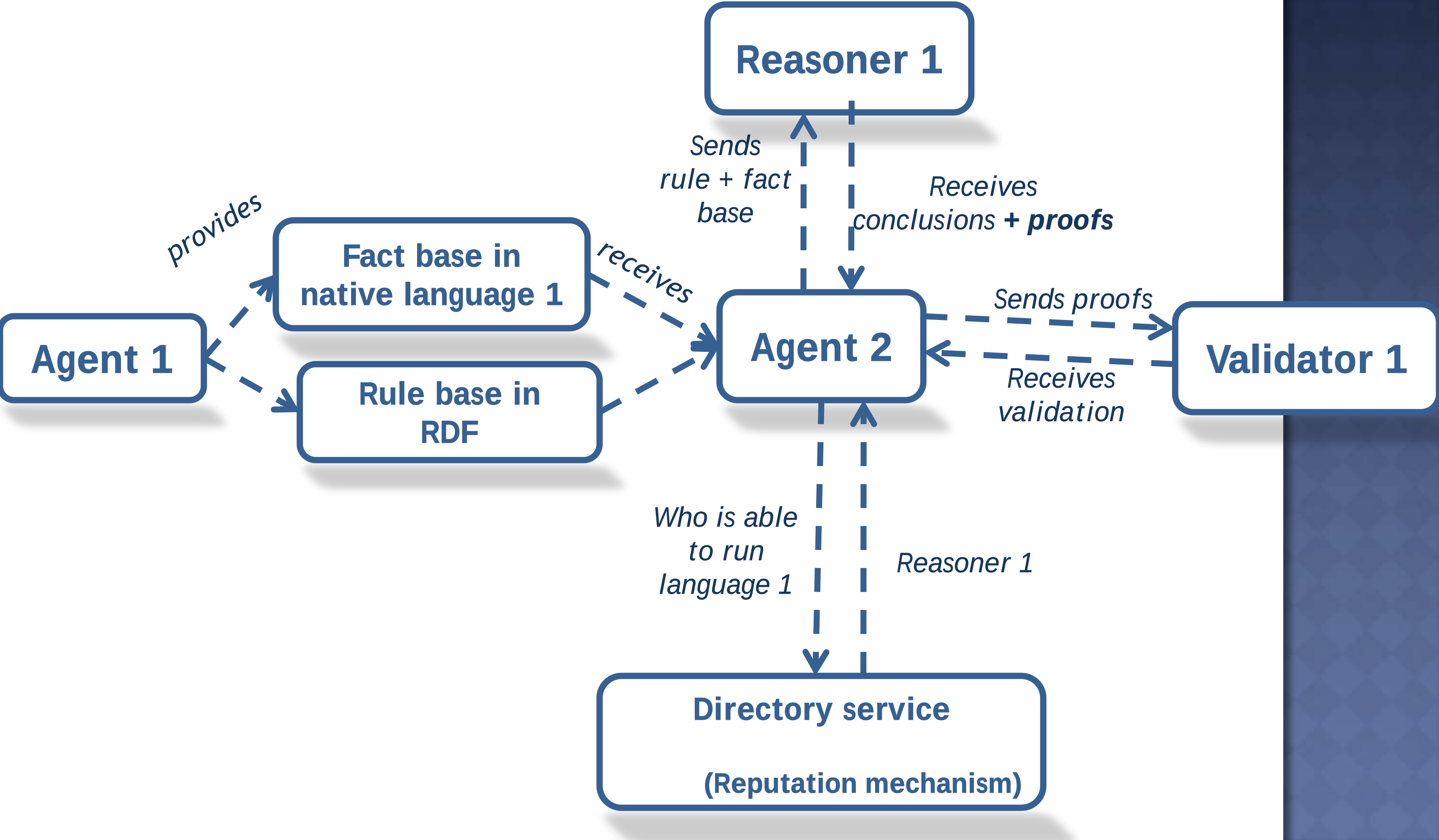
proof validation service

Input: rule base, fact base, set of conclusions, set of proofs about conclusions
Output: validation (or not) of the proofs

Implemented as:

- *third-party, independent and trusted service,*
- *preferably different from the reasoning service that provided the original conclusions + proofs*

Proof Validation



Reasoner 1

Sends rule + fact base

Receives conclusions + **proofs**

provides

Fact base in native language 1

receives

Agent 1

Agent 2

Sends proofs

Validator 1

Receives validation

Rule base in RDF

Who is able to run language 1

Reasoner 1

Directory service

(Reputation mechanism)

Overview

- *Motivation*
- *Standard rule/logic language use*
- *Proposal*
- *Semantics*
- *Proofs*
- *Proof Validation*
- *Our appoach*
  - ***EMERALD***
  - ***Reasoners***
  - ***DR-Prolog Rasoner***
- *Conclusions – Future Work*

Our approach

*Having all the above → **trust** in the exchanged rules, conclusions and proofs → **will be increased***

***key point:** reputation and trust in these reasoning/validation services*

***Our proposal:** a complete instantiation of such a framework*

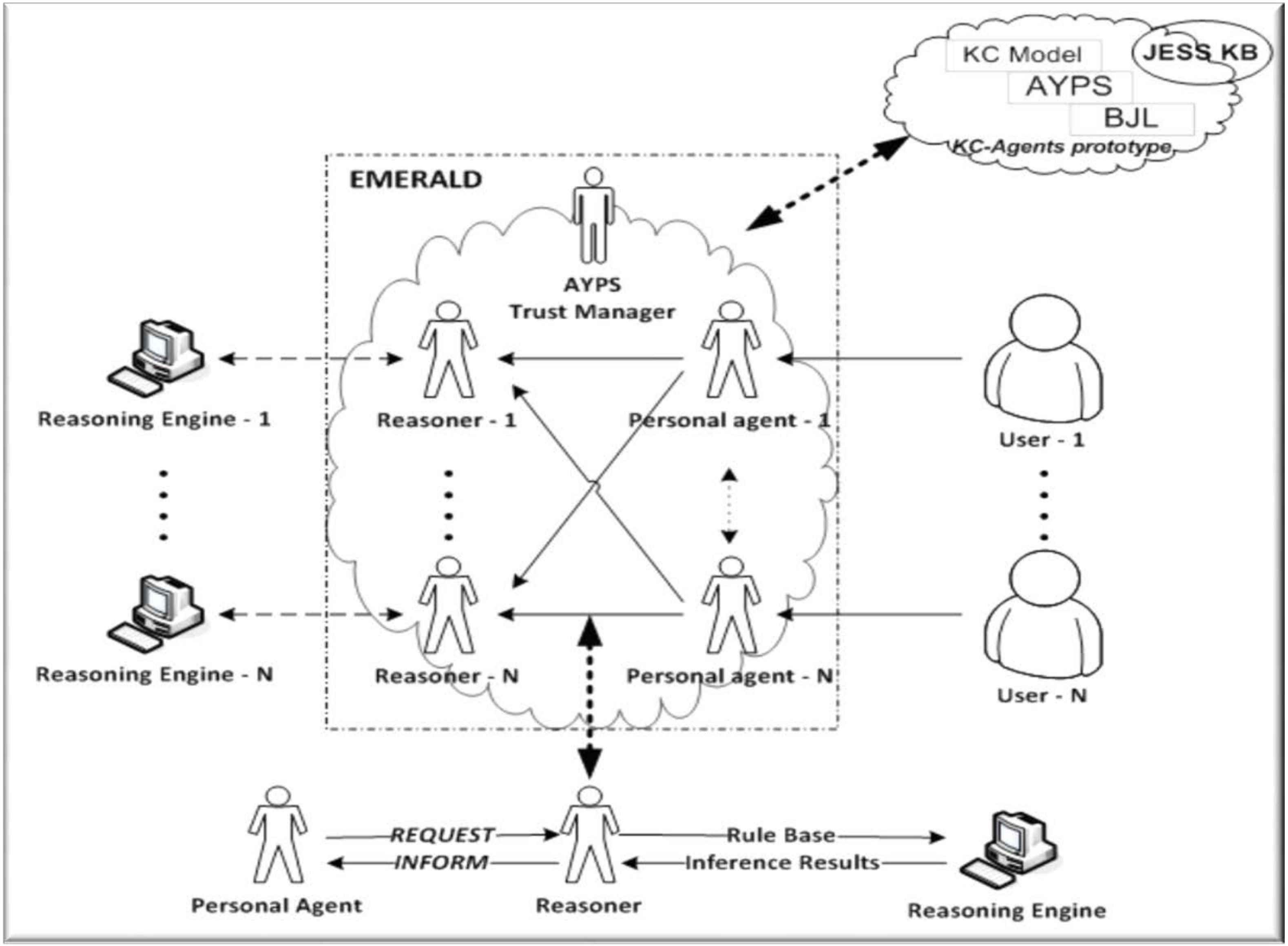
***EMERALD** (extended JADE MAS)*

implements many reasoning services (called Reasoners)

Here: DR-Prolog Reasoner (operating on defeasible rules)

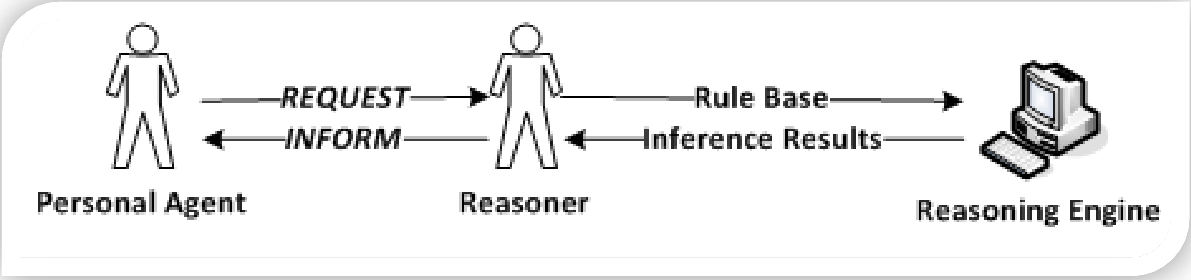
EMERALD

A Multi-Agent Knowledge-Based Framework



Reasoners

- *Built as agents*
- *Act as like web services*
- *Provide the reasoning services*
- *Launch an associated reasoning engine*



Reasoner:

- *stands* by for new requests
- *gets* a valid request → *launches* the reasoning service → *returns* the results

DR-Prolog Reasoner

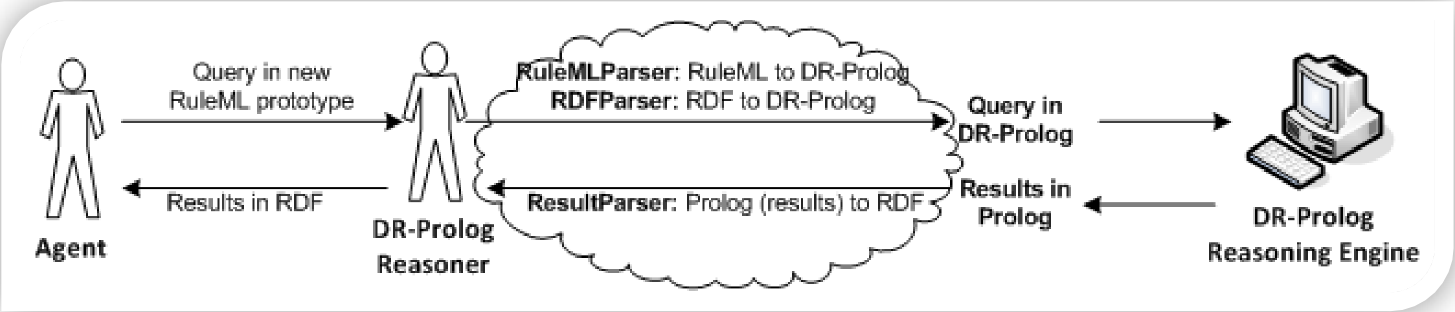
DR-Prolog[1]**:** *built on-top of Prolog*

DR-Prolog Reasoner:

- *follows the EMERALD Reasoners' general functionality*
- *With* some new intermediate steps that
  - process the receiving queries
  - send back the appropriate answer in RDF format

➢ ***extended:***

- *"Proof Generator" to explain its conclusions*
- *"Proof Validator" to validate proofs provided by other agents (in defeasible logic)*


Agent
Query in new RuleML prototype
Results in RDF
DR-Prolog Reasoner
RuleMLParser: RuleML to DR-Prolog
RDFParser: RDF to DR-Prolog
ResultParser: Prolog (results) to RDF
Query in DR-Prolog
Results in Prolog
DR-Prolog Reasoning Engine

[1] Antoniou, G., Bikakis, A.: DR-Prolog: A System for Defeasible Reasoning with Rules and Ontologies on the SW. IEEE TKDE 19,2

Overview

- ➢ Motivation
- ➢ Standard rule/logic language use
- ➢ Proposal
- ➢ Semantics
- ➢ Proofs
- ➢ Proof Validation
- ➢ Our appoach
  - o EMERALD
  - o Reasoners
  - o DR-Prolog Rasoner
- ➢ **Conclusions – Future Work**

Conclusions - Future Work

EMERALD

- a fully FIPA-compliant MAS developed on top of JADE
- proposes the use of trusted, independently-developed reasoning services

(REASONERS)

1. Can offer inferencing on a variety of logics
2. Can be used for related services such as
   a) proof explanations on the inference results
   b) Proof validations on exchanged proofs

In future:

- Integrate broader variety of reasoning and proof validation engines
- Integrate the generated proofs with trust mechanisms

EMERALD available at:

http://lpis.csd.auth.gr/systems/emerald

CS-566 Project available at:

http://www.csd.uoc.gr/~hy566/project2010.html

Thank you!

Any Questions?