

RuleML and Aristotle University Collaborate on Rule-Based Multi-Agent System Emerald RuleML Responder

**Fredericton, Canada, Thessaloniki, Greece, April 23,
2010**

[RuleML](#), the leading provider of standards for rule technologies on the Web, and the Department of Informatics of [AUTH](#) (Aristotle University of Thessaloniki), the largest university in Greece, announced a collaborative effort on integrating their platforms to enhance support for rule-based multi-agent systems. This collaboration explores the integration of the Rule Responder and Emerald platforms, with respect to their agent-connection topologies, their interchange principles, as well as their used subsets of RuleML and (bidirectional) gateways between them. Furthermore, the Prova and DR-Device reasoning engines will be supported for both Rule Responder and Emerald. RuleML is to be used for all serialization needs, in particular Reaction RuleML for event messaging with pragmatic primitives from e.g. the FIPA Agent Communication Language (ACL). The resulting Emerald RuleML Responder is to be tested both by extending the Rule Responder instantiation SymposiumPlanner to further agents with an Emerald bridge and, conversely, by extending an Emerald instantiation with a Rule Responder bridge.

The goal of the Rule Markup ([RuleML](#)) Initiative is to develop industry standards for rules on the Web using XML markup, formal semantics, and efficient implementations. RuleML covers the entire rule spectrum, from derivation rules to transformation rules to reaction rules. RuleML can thus specify inferences in Web ontologies, mappings between Web ontologies, and dynamic Web behaviors of workflows, services, and agents.

[Rule Responder](#) is a Web tool for rule-based enterprise service networks and virtual organizations, permitting rule-based collaboration between its distributed members and services. Human participants are assisted by semi-autonomous rule-based agents that proxy aspects of their owners' derivation and reaction logic. Each Rule Responder instantiation employs External Agents (EAs), an Organizational Agent (OA), and Personal Agents (PAs). An EA accepts Web queries from users and passes them to the OA. The OA represents the organization as a whole via a global rule base and assigns incoming queries to appropriate PAs. Each PA assists a single participant, (semi-autonomously) acting on his/her behalf by using a local rule base defined by the participant.

[Prova](#) is an open source rule language for creating distributed collaborating agents and rule inference services. Running in Java, it combines declarative ISO Prolog-style logic inferences with extensions for Java scripting, external data access via rich query built-ins for relational, XML and Semantic Web data, concurrent and scalable reactive messaging, workflows and event processing suitable for Enterprise Service Bus deployment. Rule Responder is closely linked to the Prova language and engine, as all

implementations of Rule Responder, use Prova as the OA's Reaction RuleML rule base.

[Emerald](#) is a JADE-based implementation framework for interoperable reasoning among agents on the Semantic Web, by using third-party trusted reasoning services. Every Emerald agent can exchange its rule base with any other agent, without the need for agents to conform to the same kind of rule paradigm or logic; the receiving agent can use an external reasoning service to interpret a rule base. Reasoning services are “wrapped” by an agent interface, called the Reasoner, allowing other agents to contact them via ACL messages. The Reasoner can launch an associated reasoning engine, in order to perform inferences and provide results. Emerald currently supports the following reasoning engines: DR-Device and Spindle, for defeasible reasoning, R-Device, for Datalog-like rules, and Prova, for logic programming.

[DR-Device](#) is a system for reasoning about RDF metadata over multiple Web sources using defeasible logic rules. Defeasible reasoning is a rule-based approach for efficient inferencing with incomplete and inconsistent information. DR-Device is implemented on top of the Clips production rule system and works by translating defeasible logic rules into production rules, which are run using the native Rete engine. Rules can be expressed in an extension of Object Oriented RuleML.