Chapter 2.6

Rule Engines and Agent-Based Systems

Agostino Poggi
Università di Parma, Italy

Michele Tomaiuolo
Università di Parma, Italy

INTRODUCTION

Expert systems are successfully applied to a number of domains. Often built on generic rule-based systems, they can also exploit optimized algorithms.

On the other side, being based on loosely coupled components and peer to peer infrastructures for asynchronous messaging, multi-agent systems allow code mobility, adaptability, easy of deployment and reconfiguration, thus fitting distributed and dynamic environments. Also, they have good support for domain specific ontologies, an important feature when modelling human experts’ knowledge.

The possibility of obtaining the best features of both technologies is concretely demonstrated by the integration of JBoss Rules, a rule engine efficiently implementing the Rete-OO algorithm, into JADE, a FIPA-compliant multi-agent system. (Figure 1)

BACKGROUND

Rule Engines

The advantages of rule-based systems over procedural programming environments are well recognized and widely exploited, above all in the context of business applications. Working with rules helps keeping the logic separated from
Rule Engines and Agent-Based Systems

the application code: it can be modified by non-developers and, being centralized in one point, it can be analyzed and validated. Rule engines are often well optimized, being able to efficiently reduce the number of rules to match against the updated knowledge base.

Rule-based systems can also be augmented with ideas and techniques developed in other research fields, leading for example to fuzzy rule-based systems, which exploit fuzzy logic to deal with imprecision and uncertainty about the knowledge base. Moreover, sometimes these systems are coupled with genetic algorithms and evolutionary programming to generate complex classifiers.

One of the most notable application of rule-based systems are expert systems, where the rule set is a representation of an expert’s knowledge. In such systems, the AI (Artificial Intelligence) is supposed to perform in a similar manner to the expert, when exposed to the same data.

Among the different mechanisms to implement a rule-engine, Rete algorithm (Forgy, 1982) has gained more and more popularity, mainly thanks to the high degree of optimization that can be obtained. At NASA Johnson Space Center, Rete algorithm was implemented in a whole generation of rule engines. OPS5 was soon replaced by its descendant, ART, and in 1984 by the more famous CLIPS.

Nowadays, one of the most widespread engines implementing Rete is Jess (Friedman-Hill, 2000), at first developed as a Java port of CLIPS at Sandia National Laboratories in late 1990s. Jess has also been widely adopted by the agent community to realize rule-based agent systems (Cardoso, 2007).

A different yet promising rule-engine is JBoss Rules (Proctor, Neale, Frandsen, Griffith & Tillery, 2007), formerly Drools. It is a quite new, but already well known, freeware tool implementing so-called Rete-OO algorithm.

Its open-source availability is a clear advantage over Jess, but an even greater advantage is due to the implementation of a particular adaptation of the Rete algorithm for the object-oriented world, rather than a literal one. This way, the burden of integrating the rule-engine and application rules with existing external objects is greatly reduced. In fact, JBoss Rules uses plain Java objects to represent rules and facts, which can be modified through their public methods and properties. Rules can be specified through an appropriate syntax, or through xml structures, and their conditions

Figure 1. Basic example from the JBoss Rules handbook

```
package org.drools.examples

import org.drools.examples.HelloWorldExample.Message;

rule “Hello World”
when
   m : Message( status == Message.HELLO, message : message )
then
   System.out.println( message );
   m.setMessage( “Goodbye cruel world” );
   m.setStatus( Message.GOODBYE );
   update( m );
end

rule “GoodBye”
no-loop true
when
   m : Message( status == Message.GOODBYE, message : message )
then
   System.out.println( message );
end
```
Related Content

Combining Ontology with Intelligent Agent to Provide Negotiation Service
Qiumei Pu, Yongcun Cao, Xiuqin Pan, Siyao Fu and Zengguang Hou (2010). *International Journal of Software Science and Computational Intelligence* (pp. 52-61).
[www.igi-global.com/article/combining-ontology-intelligent-agent-provide/46146?camid=4v1a](www.igi-global.com/article/combining-ontology-intelligent-agent-provide/46146?camid=4v1a)

A Simulation of Temporally Variant Agent Interaction via Belief Promulgation
[www.igi-global.com/chapter/simulation-temporally-variant-agent-interaction/56181?camid=4v1a](www.igi-global.com/chapter/simulation-temporally-variant-agent-interaction/56181?camid=4v1a)

A Primer on Reinforcement Learning in the Brain: Psychological, Computational, and Neural Perspectives
[www.igi-global.com/chapter/primer-reinforcement-learning-brain/49232?camid=4v1a](www.igi-global.com/chapter/primer-reinforcement-learning-brain/49232?camid=4v1a)

Role-Based Autonomic Systems
[www.igi-global.com/article/role-based-autonomic-systems/46145?camid=4v1a](www.igi-global.com/article/role-based-autonomic-systems/46145?camid=4v1a)