Business Rule Solutions

Ronald G. Ross
Principal, Business Rule Solutions, LLC
Executive Editor, Business Rules Journal
www.BRCommunity.com
Author: *Business Rule Concepts: Getting to the Point of Knowledge*

Rross@brsolutions.com

Rules Are Not Actions
So Don't Implement Them That Way!

Some professionals have a difficult time distinguishing between rules and actions. Nonetheless, they are quite different. A rule is a rule, and an action is, well, an action. Understanding the difference between these primitive elements is fundamental to the business rule approach, as this discussion explains.

A rule is a bit of declarative logic that is required to be true. Here is an example: *A salesperson may sell a product only if that salesperson represents some manufacturer that produces that product*. The sequence in which rules are evaluated should never make any significant difference to the outcome of a decision. For a process, in contrast, outcomes *always* depend on sequence.

This distinction can be difficult to grasp if you have never been exposed to a rule engine. Traditionally, rules have often been *implemented* as actions – that is, as executable statements in some kind of programming language that executes statements.

The important thing to remember is that execution style depends on the class of platform. In a rule engine environment, rule statements in declarative form are *evaluated* (not executed) by the system (rule engine) so the *system* can take responsibility for the outcome. That is in no way like writing a transforms (actions) in some procedural 3GL or 4GL, and then deliberately executing them.

A counter-argument might be the observation that rules are often executed as actions in everyday human activity. Police patrol. Inspectors inspect. Courts sentence. Employees ask, "What are you doing?" Supervisor replies, "Just checking to see if you are wearing your safety glasses." Rule enforcement seems like an action.

To fully appreciate rules you need to separate the rule statement from the *enforcement regimen* chosen for the rule. The enforcement of non-automatable rule statements (e.g., *A hardhat must be worn by any worker working in a construction site*) can be implemented only through procedures (actions) and the job responsibilities of those participating (e.g., the supervisor) in the activity. But that doesn't in any way affect the rule statement.

For automatable rules (e.g., *A group must not include both union and non-union members*), you can take advantage of the logic processing potential of computers and hand over the "processing" of the rule to a rule engine (just as you hand over the management of data to a DBMS).

A current goal in the IT industry is to make business and system *models* transformable into executable components. Obviously you want such transforms to be trustworthy and consistent and manageable. As far as I can see, as complexity scales up, the only conceivable way to achieve these goals is to address decision logic *directly*. "Directly" here means expressed as rules, not actions – and supported by the right class of platform, *rule engines*.