Expressing Business Rules

Ronald G. Ross
Business Rules Solutions, LLC
http://www.BRSolutions.com
rross@brsolutions.com

Business rules are formal statements about the data and processes of an enterprise. My overall approach to business rules is described in [1, 2]. Here, I will briefly discuss things we have learned about the expression of business rules in the last several years. This will shed light on where we stand in understanding business rules today.

First, it is clearly important to separate analysis-level expression of business rules from their design-level expression. Most of what I will say here is aimed toward the design level, but let me start with the analysis level.

Effective expression of business rules at the analysis level requires formative guidelines or Business Rule Statement Templates. Such language templates are now offered commercially (by my company and others). Think of these language templates as text or sentence patterns, to ensure higher clarity and consistency. These templates are important for making the business rule approach practical.

At the design level, how business rules are expressed to users must be cleanly separated from how they are represented inside the system. What is good for one is not good for the other.

For the *external* representation, at least several capture techniques are probably needed, each suited to different categories of rules. For example, each of the following techniques is probably well-suited to certain types of rules:

- Decision Tables for value thresholds, and perhaps certain types of computations.
- Point-and-Click Expression Builders, for instance limits and type consistency.
- Structured English, for more complex restrictions and logical inferences.
- Entity Life History or State Transition Diagrams, for both basic and more advanced state transition rules.
- Data Model or Class Model extensions, for basic property rules.

No matter how the rules are captured, there should be a single, unified *conceptual* representation "inside" of the man-machine boundary. "Inside" here means transparent to the specifiers, but visible to analysis tools (e.g., for conflict analysis) and to rule engines or business logic servers (for run-time processing).

Inside, there may be still other representations. For processing and performance reasons, there might be many physical representations of the rules, optimized for particular tools or hardware/software environments.

The result is actually three layers of representation: external, conceptual, and internal. This is strongly reminiscent of the old ANSI/SPARC three-schema architecture for data. This should not be surprising since rules simply build on terms and facts, which can be ultimately represented by data.

Which technique is the best for each representation layer is a matter of great debate. All three layers are important, but clearly the ultimate power lies in the middle or conceptual layer. The important thing for this language layer is that the rules must be represented in a sufficiently rigorous form for automated (even if not very efficient) execution [1].

Alternative candidate representations for this level of language include the following.

- Predicate Logic (the baseline – any other representation must be at least this powerful.)
- Ross Method, featuring strong rule typing [1].
- Terry Halpin's work on Object Role Modeling (ORM) [2]
- The Object Constraint Language (OCL), from the OMG [3].
- Tutorial D in C.J. Date & H. Darwen's Third Manifesto [4]

These languages are not for the faint of heart, but point toward the technological future of business rules -- supporting higher-level automation schemes for user requirements.

In retrospect, I believe the main contribution of [1] is its highly organized scheme for the *conceptual* representation of all rules. This representation is based on rule typing (patterns), which I believe is a level above other approaches. The graphic notation might be useful for capturing *certain* types of rules at the external layer – especially using a point-and-click environment. However, this would certainly not be optimal for all rules.

Where is this research now? A new, more concise representation scheme is under development. One focus of this scheme is a formal expression of how non-atomic rule types are derived from atomic ones. This would allow reduction of rules to a common base of fundamental rule types, in order to support automatic analysis of conflict and overlap in systematic fashion.

This is opening exciting new avenues of research, and significant opportunities for those interested in getting involved.

REFERENCES

[1] Ross, Ronald G., *The Business Rule Book*, Business Rule Solutions, 2nd Edition, 1997.

[2] Ross, Ronald, G., *Business Rule Concepts*, Business Rule Solutions, 1998.

[3] Halpin, Terry, *Conceptual Schema and Relational Database Design*, 2nd Edition, Prentice-Hall, 1995.

[4] http://www.omg.org

[5] Hugh Darwen and Chris J. Date, *Foundation for Object/Relational Databases: The Third Manifesto*, Addison-Wesley, 1998.