11/22/00

A Roadmap for Rules and RuleML in the Semantic Web

Benjamin Grosof
MIT Sloan School of Management, Cambridge, MA, USA
bgrosof@mit.edu ; **http://ebusiness.mit.edu/bgrosof**

RuleML is the leading approach to Semantic Web rules: Over the last two and a half years or so, a broad consensus has evolved in the Semantic Web community that the vision of the Semantic Web includes, specifically, rules – as well as ontologies cf. OWL. RuleML has emerged as the leading current standardization approach to rules for the Semantic Web. We are a co-founder and co-chair of the RuleML Initiative, and co-chair of the closely related DAML Rules effort which is currently the main focus of the Joint Committee that developed DAML+OIL. In this paper, we give a roadmap for rules in the Semantic Web, including overview, design rationale, and current directions of RuleML.

Declarative knowledge representation: Standards should be founded upon techniques that are well established at a research level. The only approach to making the Semantic Web be "Semantic" that is well understood – and well accepted – from a substantial body of previous research is that it must be founded upon *declarative knowledge representation* (KR). RuleML, like OWL and RDF, shares this approach. To declarative KR, each then adds *Webized* syntax.

``Declarative " here means in the sense of (AI) KR theory. A given set of premises entails a set of sanctioned conclusions, independent of inferencing control strategy or procedural aspects, e.g., independent of whether inferencing direction is goal-directed query answering (``backward") vs. data-driven (``forward"). Declarativeness greatly facilitates reuse, and multiple kinds of uses, of knowledge. It enables one knowledge-based application (i.e., "agent") to anticipate precisely and completely what meaning will be drawn by anotheragent from knowledge that has been communicated (e.g., shared).

"Webized" here means using the Web's overall open spirit and standards suite – in particular XML, URI's, and namespaces – so as to support modules of multiply authored, widely distributed knowledge. RuleML has initially emphasized XML as the form of syntax, to facilitate building of tools for translation and inferencing. But it also has drafted an RDF syntax and a human-oriented string syntax (as well as an abstract syntax that bridges all these).

However, for rules, as for ontologies, the most crucial design choice is the choice of underlying fundamental knowledge representation.

Declarative logic programs as KR shared by commercially important rule systems: There are four families of rule systems that are the most currently commercially important (``CCI"): SQL (relational database), Prolog, production rules (cf. OPS5, CLIPS, Jess) and Event-Condition-Action rules (ECA). These kinds of rules today are often found embedded in systems built using Object-Oriented (OO) programming languages (e.g., C++ and Java), and are often used for business process connectors / workflow. These families of rule systems since the 1980's have been in growing commercial deployment, with manifold diverse e-business applications. RuleML takes as a prime requirement that a Semantic Web rules language must, first and foremost, support interoperability among rule-using applications (agents) that make use of heterogeneous members of these CCI rule systems. A key observation is that these four families of CCI rule systems all have a common core abstraction: the declarative *logic programs* KR ("LP"). RuleML has, accordingly, started from this KR.

In particular, the Datalog Horn case of LP is the kernel expressiveness of RuleML. Why? It is a relatively simple core shared by all four CCI families. It is the most well-studied subset of LP. It is logically monotonic. It is a subset of classical First Order Logic (FOL). (It is a moderate weakening of Datalog Horn FOL: conclusions are essentially restricted to ground atoms.) It is the heart of SQL / relational algebra. It has the great virtue of computational tractability (polynomial-time inferencing, given a bounded number of logical variables per rule). Tractability enables practical scaleability and inferential completeness.

Wide familiarity with LP, Horn FOL, and Rules: Hundreds of thousands of developers, and millions of IT users, are more or less familiar with one or more of these CCI families of rule systems – especially SQL, and Prolog (including via academic training) to a lesser extent. Ditto for Horn FOL (via academic training). Declarative rules (vs. code) provide a relatively high level of conceptual abstraction that makes it easier for non-programmers to understand, specify, and dynamically modify and merge them. They are executable but can be managed as data, separate from code.

Nonmonotonicity and procedural attachments are vital and doable: To this kernel, RuleML's approach is to evolutionarily add a *family* of extensions for various further expressive features and restrictions, cf. established research and driven by applications needs. Especially important are extensions to enable nonmonotonicity and procedural attachments. One nonmonotonic feature is *negation-as-failure* (NAF), which is found in all the CCI families, and is heavily used in all of them except SQL. LP with NAF ("Ordinary" LP, a.k.a. "Normal" LP) has been well studied. We have proposed the design of two other major features, which are currently under discussion within the RuleML Initiative. The first is the nonmonotonic feature of *prioritized conflict handling*, which also is found in all the CCI families, and is heavily used in all of them. Examples include: priority between rules in Prolog based on static rule sequence; dynamically-computed priorities among rules in production rule and ECA rule systems; inheritance with exceptions; and updating in databases (where more recent assertions override previous ones). The third feature is *procedural attachments*: to perform *actions* triggered by drawing conclusions, and to perform *queries* when testing rule antecedent conditions. Procedural attachments are found in all the CCI families, and are used heavily in their applications. The *Situated Courteous* extension of LP (SCLP), which we have developed [1], includes all three of these features. It is declarative and preserves tractability. However, it is as yet less well studied than ordinary LP. RuleML defines a number of other expressive extensions, and also expressive restrictions, as well.

Classical logic is thus not enough: Knowledge Interchange Format (KIF), and its close successor (Simple) CommonLogic, are based on the underlying KR of FOL. This fundamentally lacks the ability to express nonmonotonicity or procedural attachments, and (beyond LP) has thus not become widely deployed for commercial applications – partly due also to its intractability.

But LP's overlap with classical logic is important to support: LP and FOL overlap, but each also has non-overlap with the other. Ditto LP and OWL's Description Logic (DL). We believe that it is important to support non-LP FOL (e.g., material implications between complex formulae) as an additional direction of expressiveness for rules. For example, this is very useful in the Description Logic Programs approach [2] that we have developed to combining the semantics of LP and DL, where RuleML/LP rules refer-to/import ontological knowledge from OWL/DL (e.g., definitions of class or property predicates). The Lloyd-Topor transformation [3] provides an rich expressive extension of LP that is tractably reducible to Ordinary LP. Limited classical ("strong") negation is another such extension [1]. CommonLogic and LP RuleML can and should share syntax as well as semantics to a great extent. RuleML has led the way on developing Webized such syntax. Indeed, it would be reasonable to treat, and develop, CommonLogic as part of the RuleML family.

There's lots of room for new services applications, research, and standards: Rules promise to be useful in a variety of new Semantic Web Services applications, e.g., our work on e-contracts, financial knowledge integration, and travel packages. Much more than ontologies, rules can *do* stuff. However, achieving the full vision of Semantic Web rules will require more research, e.g., to extend understanding of how LP relates to DL/FOL, and of how to reconceive rule KR as essentially highly distributed, especially when featuring nonmonotonicity, procedural attachments, and events. And more kinds of KR's closely related to rules – notably, probabilistic, inductive, and constraint-based – are needed for the full Semantic Web vision, yet have hardly been addressed yet in standardization efforts.

References:

[1] Grosof, B., Labrou, Y., and Chan, H., "A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML", Proc. 1st ACM Conf. on Electronic Commerce (EC-99), 1999.
[2] Grosof, B., Horrocks, I., Volz, R., and Decker, S., "Description Logic Programs: Combining Logic Programs with Description Logic". Proc. 12th Intl. Conf. on World Wide Web (WWW-2003), 2003.
[3] Lloyd, J., "Foundations of Logic Programming", 2nd edition, Springer-Verlag, 1987.