Reaction RuleML

Adrian Paschke, Alexander Kozlenkov, Harold Boley, Michael Kifer, Said Tabet, Mike Dean, Keara Barrett

– – – – – http://ibis.in.tum.de/research/ReactionRuleML/ – – – – –

Abstract— Reaction RuleML is a general, practical, compact and user-friendly XML-serialized language for the family of reaction rules.

Index Terms— Rule Markup, Event Driven Architectures, ECA, Production Rules, Rule Interchange, Event / Action / State Processing

– – – – – – – – – – – ◆ – – – – – – – – – – –

Reaction RuleML is a general, practical, compact and user-friendly XML-serialized language for the family of reaction rules. It is intended for distributed event driven architectures (EDAs) and reactive, rule-based Semantic Web Service-Oriented Architectures (SOAs), where reaction rules of the various kinds need to be

- serialized in a homogeneous combination with other rule types such as derivation rules, normative rules or integrity constraints;
- managed, maintained and interchanged in a common rule markup and interchange language;
- internally layered to capture sublanguages such as production rules, ECA rules, event notification rules, KR event/action/state processing and reasoning rules;
- translated and executed in different target environments with different operational, execution and declarative semantics;

This extension of RuleML underpins rule-based systems for (pro-)active real-time or just-in-time (re-)actions to events (short-term perspective) but also retrospective and prospective reasoning on events, actions and their effects on changeable knowledge states (long-term perspective akin to state machines, process algebras or transition systems). Due to the formal semantics reasoning produces traceable and verifiable results, which is crucial as a foundation of complex and reliable event driven architectures and behavioural agent and business logic systems. This combination of reactive, reasoning and verification capabilities is a key factor in upcoming agile and flexible IT infrastructures, distributed loosely coupled service-oriented environments, new business models such as On-Demand or Utility Computing as well as new industry trends such as Real-Time Enterprise (RTE), Business Activity Management (BAM) or Business Performance Management (BPM) and closely related areas such as Service Level Management (SLM) with monitoring and enforcing of Service Level Agreements (SLAs) and Business Policies.

Reaction RuleML incorporates different kinds of production, action and reaction rules into the native RuleML syntax using a system of step-wise extensions. In particular, the approach covers a broad spectrum of different kinds of reaction rules with a rich syntax for complex event, action, (post-)condition, time/interval and state processing capabilities from various domains such as active databases with ECA(P) rules and triggers, forward-directed production rule systems, temporal KR/AI event / action / fluent logics, event notification & messaging systems as well as active KR/AI update, transition, process and transaction logics. External systems can be easily integrated into the rule executions in an active style via procedural calls on programming APIs (e.g. Java) enabling active sensing (pull) and triggering of actions (push) or in a passive listening style where events might be inbound messages communicated by an external agent node, event notification or publish/subscribe system. This coherent combination naturally fits into open distributed environments, particulary on the Web. The layered and uniform design of Reaction RuleML makes it easier to learn the language and to understand the relationship between the different features and it provides certain guidance to vendors who might be interested only in a particular subset of the features and do not need support for the full expressiveness of Reaction RuleML. Reaction RuleML fulfils typical criteria for good language design such as minimality, symmetry and orthogonality and satisfies typical knowledge representation adequacy criteria such as epistemological adequacy in view of expressiveness of the language. Full Reaction RuleML does not typically need to be executed directly, but its various sublanguages can be transformed into target execution languages of underlying rule-based systems, e.g., a business rule management system (BRMS), a production rule expert system, an active database system or another kind of event-driven architecture (EDA). XSLT stylesheets are provided that transform Reaction RuleML (sublanguages) into executable rule scripts for execution in the targeted computation environments.

In summary, Reaction RuleML provides a serialization support for systems that automate business and behavioural processes. Compared to traditional event-driven systems, this approach has the following major advantages:

- rules are externalized and easily shared among multiple applications (avoiding vendor lock-in) ;
- encourages reuse and shortens development time;
- changes can be made faster and with less risk;
- lowers cost incurred in the modification of business and reaction logic;

Reaction rules constitute the next step in the application of information system (IS) technology aimed at automating reactions to events occurring in open service-oriented Web applications. Automated services that have business and reaction logic embedded inside often take substantial time to change, and such changes can be prone to errors. In a world where the life cycle of business models and applications/services is steadily shortening, it has become increasingly critical to be able to adapt to changes in external environments promptly. These needs are addressed by Reaction RuleML. Reaction RuleML improves the agility of event driven architectures and the manageability of business processes and behavioural reaction logic, since rules become more accessible.