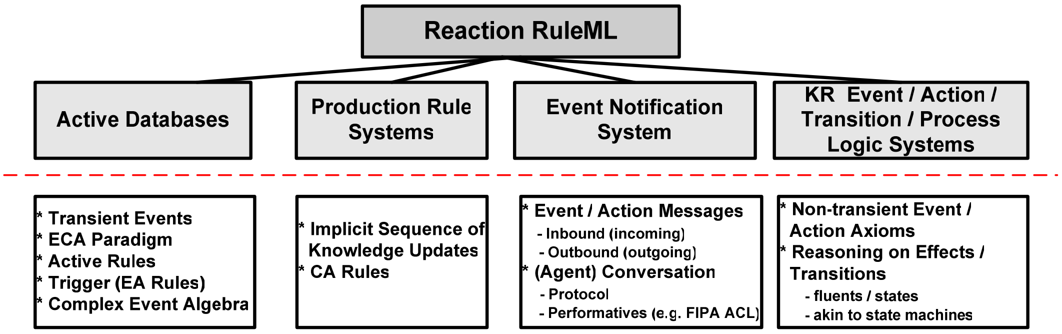
Int. Conf. On Rules and Rule Markup Languages for the Semantic Web (RuleML'06)

Reaction RuleML
Technical Group

Reaction RuleML

Scope of Reaction RuleML

Reaction RuleML is a general, practical, compact and user-friendly XML-serialized language for the family of reaction rules. It incorporates different kinds of production, action, reaction, and KR temporal/event/action logic rules into the native ***RuleML syntax*** using a system of step-wise extensions. In particular, the approach covers different kinds of reaction rules from various domains such as active-database ECA rules and triggers, forward-directed production rules, backward-reasoning temporal-KR event/action logics, event notification & messaging and active update, transition, process and transaction logics.



Reaction RuleML

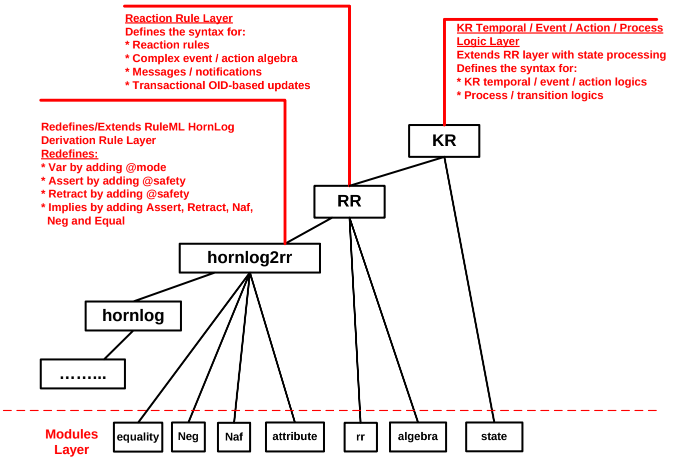
Active Databases
- Transient Events
- ECA Paradigm
- Active Rules
- Trigger (EA Rules)
- Complex Event Algebra

Production Rule Systems
- Implicit Sequence of Knowledge Updates
- CA Rules

Event Notification System
- Event / Action Messages
  - Inbound (incoming)
  - Outbound (outgoing)
- (Agent) Conversation
  - Protocol
  - Performatives (e.g. FIPA ACL)

KR Event / Action / Transition / Process Logic Systems
- Non-transient Event / Action Axioms
- Reasoning on Effects / Transitions
  - fluents / states
  - akin to state machines

Layered Reaction RuleML Language



Reaction Rule Layer
Defines the syntax for:
* Reaction rules
* Complex event / action algebra
* Messages / notifications
* Transactional OID-based updates
KR Temporal / Event / Action / Process Logic Layer
Extends RR layer with state processing
Defines the syntax for:
* KR temporal / event / action logics
* Process / transition logics
Redefines/Extends RuleML HornLog Derivation Rule Layer
Redefines:
* Var by adding @mode
* Assert by adding @safety
* Retract by adding @safety
* Implies by adding Assert, Retract, Naf, Neg and Equal
KR
RR
hornlog2rr
hornlog
....
Modules Layer
equality
Neg
Naf
attribute
rr
algebra
state

⇒General, practical, compact, user-friendly XML serialization syntax for reaction rules

⇒Expressive with minimal, symteric and orthogonal language design

⇒Supports different reaction rule types such as ECA rule, active rules, production rules, temporal KR event/action logics, state processing and transition rules, update transactions etc.

⇒Intended for (Semantic Web) based Event-Driven Architectures (EDAs) and Service-Oriented Architectures (SOA)

⇒Supports e.g.:

- ♦ Real-time Enterprises (RTE)
- ♦ Business Activity Management (BAM)
- ♦ Service Level Management (SLM),

⇒Tool support via Validators, Translators, Editors

⇒Layered Uniform Schema Design

- ♦ Easy to learn and understand
- ♦ Guidance to vendors which need smaller subset
- ♦ Easier to maintain and extend

Core Syntax

Glossary

Reaction: General reaction rule construct
@exec: Denotes execution style of the reaction rule: "active | passive | reasoning"; default = "passive"
@kind: Required attribute denoting the kind of the reaction rule, i.e. the rule pattern which defines the constituent parts of the reaction rule
@eval: Attribute denoting the interpretation of the reaction rule: "strong | weak"; default="strong"
event, body, action, postcond, alternative: Role tags for the reaction rule parts which might be omitted (see RuleML role and type tags)

Syntax

Reaction ::= [oid,] [event,] [body,] [action] [,postcond] [,alternative]
event ::= Naf | Neg | Atom | Message | Reaction
body ::= Naf | Neg | Atom | And | Or
action ::= Atom | Assert | Retract | Message
postcond ::= Naf | Neg | Atom | And | Or
alternative ::= Atom | Assert | Retract

Example

```
<Reaction exec="active" kind="ecapa">
                    <event> <Atom>…<Atom> <event>
                   <body> <Atom>…</Atom> </body>
                   <action> <Atom>…</Atom> </action>
                   <postcond> <Atom>…</Atom> </postcond>
                   <alternative> <Atom>…</Atom></alternative>
</Reaction>
```

Contact: Adrian.Paschke@gmx.de and Alex.Kozlenkov@betfair.com
Project Site: http://ibis.in.tum.de/research/ReactionRuleML/

Reaction RuleML Technical Group
Alexander Kozlenkov (Co-Chair)
Adrian Paschke (Co-Chair)

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