An Event-Condition-Action Logic Programming and Rule Markup Language

Chair of Internet-based Information Systems

ECA-LP / ECA-RuleML

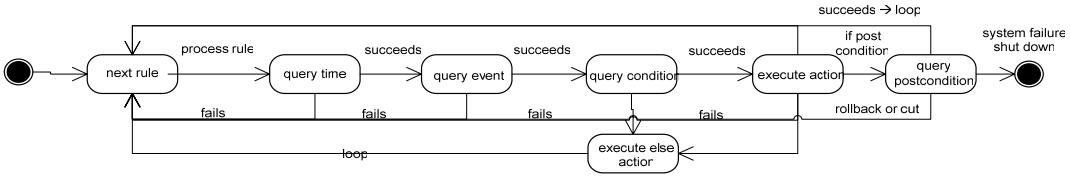
*ECA-LP Syntax – Homogeneous **Event-Condition-Action Logic Programming** Language*

ECA rule: *eca (<Time>,<Event>, <Condition>, <Action>, <Post-Cond.>, <Else Action>)**

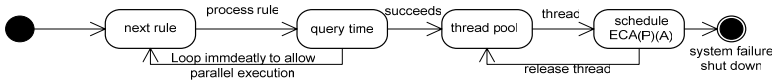
* All ECA rule parts are optional, except of action; An ECA rule is interpreted as top query

(Time): Pre-conditional time function used as validity clock / timer
(Event): Actively detect/listen to internal and external (complex) events (clocked by time function)
(Condition) : Conditional test
(Action): Internal self-update action or external action with side effects; might be complex and transactional
(*Post-Condition*): Post-conditional test; might commit or rollback action; supports cuts and variable quantifications
(Else Action) Executes alternative action if condition or action fails (akin to "if then else" logic)

Operational Semantics – ECA Interpreter with Active Query Daemon for arbitrary Rule Engines



Multi-Threading Parallel Scheduling of Reaction Rules



Declarative Semantics: Logic Programming

- ECA rule is top query: *T ∧ E ∧ ((C ∧A ∧ P) ∨ EL) ?.*
- Declarative Logic Programming semantics for **PROGRAMMING** of ECA functionalities in terms of derivation rules or Boolean-valued procedural attachments (assigning truth values)
- Interval Based Event Calculus
  - Transient and non-transient events/actions
  - State/fluent processing / KR reasoning
  - complex interval-based event / action algebra
- 3-Phases for event
  (1) definition (2) selection (3) consumption
  - Configurable selection and consumption policies
- Transactional complex updates or external actions
  - Dynamic OID-based transactional LP updates
  - Sequence of transitions with integrity tests and possible rollbacks
  - External actions with side effects via attachments

⇒Homogenous representation with other rule types, e.g. derivation rules, integrity constraints
⇒ECA Interpreter for arbitrary rule engines
- **Active Reaction Rule Processing**
- **Variables + quantifications, negation, connectives, attachments, preferences**

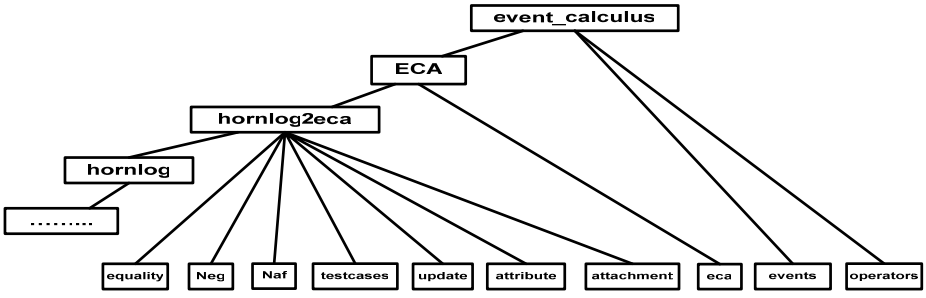
⇒Complex interval-based event / action algebra
- **Algebra operators, e.g. sequence, xor**
- **Event selection / consumption**
- **Complex actions and active rules**

⇒Temporal KR event / action logics
- **state processing / state transitions**
- **KR reasoning (retrospective / planning)**

⇒Transactional OID-based Update Actions
- **Intensional and extensional updates**
- **Transactional knowledge state transitions with rollbacks/commits**
- **Post-conditional integrity tests**

Within the scope of:

ECA-RuleML: Layered Serialization Syntax for Reaction Rules based on RuleML



Dipl. Wirtsch-Inf. Adrian Paschke
Roland Berger & O2 Germany Lehrstuhl
Internetbasierte Geschäftssysteme
Fakultät für Informatik (I18)
Technische Universität München
Boltzmannstr. 3
85748 Garching / München

Telefon: 089-289-17504
Fax: 089-289-17535
E-Mail: paschke@in.tum.de
WWW: http://ibis.in.tum.de

Contact: Adrian.Paschke@in.tum.de

Project Site: http://ibis.in.tum.de/research/eca-ruleml