1. ECA-RuleML Constructs

<action>

The content model of the action role is defined as (Naf | Neg | Cterm | Assert | Retract | RetractAll). The role is used in the content models of <ECA>, <Happens>, <Planned>, <Initiates> and <Terminates> (See ECA Example).

(See: eca_module.xsd)

<Attachment>

The Attachment element enables the integration of procedural attachments in ECA-RuleML. The content model of the element is defined as (oid?, (Ind | Var | Cterm ), Ind ). The elements <oid>, <Ind>, <Cterm> and <Var> are defined by RuleML. On the eca layer of ECA-RuleML the <Cterm> has been redefined so that <Attachment> is included. The content model of <Cterm> has been changed as follows:

(oid?, (op | Ctor | Attachment), (slot)*, (resl)?, (arg | Ind | Data | Skolem | Var | Reify | Cterm | Plex )* , (repo)?, (slot)*, (resl)?)

Example:

```
<CTerm>
  <Attachment>
    <oid> JavaPrintOut </oid>
    <Ind> System.out </Ind>
    <Ind> print </Ind>
  </Attachment>
  <Ind> Hello! </Ind>
</CTerm>
```

The <Cterm> redefinition enables nesting.

Example:
The binding to a variable is enabled by `<Equal>` (defined by RuleML – see: [6]/0.9/xsd/modules/equality_module.xsd).

**Example:**

```
<Equal>
  <Var> Y </Var>
  <Cterm> [Attachment] </Cterm>
</Equal>
```

(See: attachment_module.xsd)

**<condition>**

The **condition** role has the following content model: (Naf | Neg | Cterm | Assert | Retract | RetractAll). The role is used in the content models of `<ECA>` element.

(See: eca_module.xsd)

**<ECA>**

ECA’s content model is (oid?, time?, event?, condition?, action, postcondition?, else?). The ECA element enables serialization of reactive rules.

**Example:**

```
ECA-RuleML:

<ECA>
  <time>
    <Cterm> everySec </Cterm>
    <Ind> 10 </Ind>
  </Cterm>
</time>
```
The `else` role has the following content model: `(Naf | Neg | Cterm | Assert | Retract | RetractAll)`. The role is one of the parts of the `<ECA>` element.

(See: eca_module.xsd)

The `event` role has the following content model: `(Naf | Neg | Cterm | Assert | Retract | RetractAll)`. The role is one of the parts of the content models of `<ECA>`, `<Happens>`, `<Planned>`, `<Initiates>` and `<Terminates>` elements (See ECA Example).

(See: eca_module.xsd)

The fluent role is defined with its content model `(Ind | Var | Cterm)` in the `events_module` of the ECA-RuleML language.

(See: events_module.xsd)

`Happens` is declared in its module with the following content model: `(oid?, (event | action | Ind | Var | Cterm ), (time | Ind | Var | Cterm))`. 
<HoldsAt>

The primary structure of **HoldsAt** is declared as follows: (oid?, (fluent | Ind | Var | Cterm ), (time | Ind | Var | Cterm)).

(See: events_module.xsd)

<Initially>

Its primary content model as declared in events_module is (oid?, (fluent | Ind | Var | Cterm)).

(See: events_module.xsd)

<Initiates>

Its primary structure as implemented in events_module is (oid?, (event | action | Ind | Var | Cterm), (fluent | Ind | Var | Cterm), (time | Ind | Var | Cterm)).

(See: events_module.xsd)

@mode

The role of the **mode** attribute is to show if a variable is intended to be an input or an output term. The attribute is a restriction with the following three values: “?” undefined, “+” to be input and “−” to be output. Its use is optional. The attribute is added to the attribute list of the `<Var>` element at the hornlog2eca layer.

(See: attribute_module.xsd)

<parameter>

Its structure is described by the following content model: (Ind | Var | Cterm).

(See: events_module.xsd)
<Planned>

The primary structure of Planned is defined by the events_module as (oid?, (event | action | Ind | Var | Cterm), (time | Ind | Var | Cterm)).
(See: events_module.xsd)

<postcondition>

The postcondition role has the following content model: (Naf | Neg | Cterm | Assert | Retract | RetractAll). The role is one of the parts of the <ECA> element (ECA Example).
(See: eca_module.xsd)

<ECA-RuleML>

ECA-RuleML is the top element of the ECA-RuleML language. The content model is as follows: (Assert*, Query*, Protect*).
(See: root_module.xsd)

<Retract>

The Retract element is defined as follows: ((oid | Atom | ECA | Happens | Planned | Initially | Initiates | Terminates | HoldsAt | ValueAt)*, TestCase? ).
(See: update_module.xsd, eca.xsd and event_calculus.xsd)

<RetractAll>

The RetractAll element has the same content model as <Retract>. The content model of RetractAll is as follows: ((oid | Atom | ECA | Happens | Planned | Initially | Initiates | Terminates | HoldsAt | ValueAt)*, TestCase? ). For more details see the description of <Retract>.
(See: update_module.xsd, eca.xsd and event_calculus.xsd)
<Rulebase>

The content model of <Rulebase> is: (Fact*, Rule*, ECA*, Query*, Integrity*, Assert*, TestCase*, Retract*, RetractAll*).

(See: repository_module.xsd)

@safety

The safety attribute is restricted to the values transaction and normal. Its role is to indicate when the function must be started as transaction and when not. The safety attribute is included by a redefinition of <Assert> on the hornlog2eca layer. The attribute is part of the attribute lists of <Retract> and <RetractAll>.

(See: attribute_module.xsd)

@semantics

The semantics attribute is restricted to string values. Its role is to provide information about different semantics. It occurs just in <TestCase>.

(See: testcases_module.xsd)

<Terminates>

The structure of Terminates is: (oid?, (event | action | Ind | Var | Cterm), (fluent | Ind | Var | Cterm), (time | Ind | Var | Cterm)).

(See: events_module.xsd)

<Test>

Test’s content model is as follows: (oid?, Ind?, Query). The Test element is part of <TestCase>.

(See: testcases_module.xsd)

<TestCase>
The **TestCase** element is defined in the testcases_module with the following content model: `(oid?, Test+, Atom*, Implies*, Integrity*)`. The usage of the `@semantics` attribute is optional.

(See: testcases_module.xsd)

**<time>**

The **time** role has the following content model: `(Naf | Neg | Cterm | Assert | Retract | RetractAll)`. The role is one of the parts of the content models of `<ECA>`, `<Happens>`, `<Planned>`, `<Initiates>`, `<Terminates>`, `<HoldsAt>` and `<ValueAt>` elements (See ECA Example).

(See: eca_module.xsd)

**<ValueAt>**

The content model is `(oid?, (parameter | Ind | Var | Cterm), (time | Ind | Var | Cterm), (Ind | Var | Cterm))`.

(See: events_module.xsd)

2. ECA-RuleML Extensions to the RuleML Schemas

The ECA-RuleML language builds on the existing XML derivation language RuleML. A little glossary of the extended RuleML elements in ECA-RuleML is given in this section.

**Glossary**

**<Assert>**

The **Assert** element is defined by RuleML and redefined and extended by ECA-RuleML. The original content model of the element at the hornlog layer is: `(oid?, (formula | Atom | Implies | Equivalent | Forall)*)`. The new top level content model
of `<Assert>` in ECA-RuleML is: ( oid?, (formula | Atom | Implies | Equivalent | Forall | TestCase | ECA | Happens | Planned | Initially | Initiates | Terminates | HoldsAt | ValueAt | Overrides)* ). `<Assert>` provides the structure for adding of new knowledge in the knowledgebase and is defined under the `<RuleML>` element in RuleML and under the `<ECA-RuleML>` element in the ECA-RuleML language. `<Assert>` is the element that should provide connectivity between the different contract modules.

**Example:**

**Assert in a module definition:**

```xml
<Assert>
  <oid> new knowledge </oid>
  <Atom>
    <Rel> consumption </Rel>
    <Ind> 1er BMW </Ind>
    <Ind> max 6,5l </Ind>
    <Ind> per 100 km </Ind>
  </Atom>
</Assert>
```

**Assert as reference to a module definition:**

```xml
<Assert>
  <oid> rules/module.rbsla </oid>
</Assert>
```

Thereby, the oid element contains a reference to the file where the definition of the imported module is made.

(See: [6]/0.9/xsd/modules/performative_module.xsd, hornlog2eca.xsd, eca.xsd, event_calculus.xsd)

**<Cterm>**

The **Cterm** element is redefined by the first layer of ECA-RuleML. The ECA-RuleML element **Attachment** is added and the new content model of **Cterm** is: ( oid?, (op | Ctor | Attachment), (slot)*, (resl)?, (arg|Ind|Data|Skolem|Var|Reify|Cterm|Plex)*, (repo)?, (slot)*, (resl)? )

(See: hornlog2eca.xsd)

**<Implies>**
The **Implies** element is redefined by ECA-RuleML. The content model at the hornlog layer is defined as follows: 
\[(oid?, \ (\ head, \ body) \ | \ (\ body, \ head) \ | \ (\ Atom \ | \ And \ | Or), \ Atom)\].

The new top level content model in ECA-RuleML is: 
\[(oid?, \ (\ head, \ body) \ | \ (\ body, \ head) \ | \ (\ Atom \ | \ And \ | Or \ | Assert \ | Retract \ | RetractAll \ | Happens \ | Planned \ | Initially \ | Initiates \ | Terminates \ | HoldsAt \ | ValueAt ), \ (\ Atom \ | \ formula \ | \ Happens \ | Planned \ | Initially \ | Initiates \ | Terminates \ | HoldsAt \ | ValueAt ))\].

The attributes are @closure, @direction, @kind and @variety.

(See: [6]/0.9/xsd/modules/connectiv_module.xsd, hornlog2eca.xsd, event_calculus.xsd)

### <Integrity>

The **Integrity** element is used to define integrity constraints:

**Example:**

```xml
<Integrity>
  <Neg>
    <Atom>
      <Rel>cold</Rel>
      <Var>object</Var>
    </Atom>
    <Atom>
      <Rel>hot</Rel>
      <Var>object</Var>
    </Atom>
  </Neg>
</Integrity>
```

The content model at top level of ECA-RuleML language is: 
\[(oid?, \ (\ formula \ | \ Atom \ | \ And \ | \ Or \ | \ Implies \ | \ Happens \ | Planned \ | Initially \ | Initiates \ | Terminates \ | HoldsAt \ | ValueAt ))^+\]

(See: [6]/0.9/xsd/modules/connective_module.xsd, hornlog2eca.xsd and event_calculus.xsd)

### <Naf>

The ECA-RuleML content model of **<Naf>** is: 
\[(oid?, \ (Atom \ | \ Cterm))\].

(See: [6]/0.9/xsd/modules/naf_module.xsd and hornlog2eca.xsd)
<Neg>

<Neg> is the construct that provides the classical negation. Its ECA-RuleML content model is: (Atom | Equal | Cterm)

(See: [6]/0.9/xsd/modules/neg_module.xsd and hornlog2eca.xsd)

<Query>

The ECA-RuleML language extends it by adding the constructs for event processing. The top level content model becomes (oid?, (formula | Atom | And | Or | Exists | Happens | Planned | initially | Initiates | Terminates | HoldsAt | ValueAt)).

(See: [6]/0.9/xsd/modules/performative_module.xsd and event_calculs.xsd)

<Var>

<Var> is extended at the first ECA-RuleML layer by adding the @mode attribute.

(See: hornlog2eca.xsd)
Figure 1: RuleML schema’s structure
Appendix B – ECA-RuleML 0.1

**Redefine:**
- Assert, Implies, Integrity, Query, Retract and RetractAll by adding event processing constructs
  - Assert by adding ECA
  - Retract and RetractAll by adding ECA

**Redefine:**
- Neg by adding Equal and Cterm
- Naf by adding Cterm
- Cterm by adding Attachment
- Integrity by adding Implies
- Assert by adding TestCase and @safety
- Implies by adding formula, Assert, Retract and RetractAll
- Var by adding @mode

**Diagram:**

- **event_calculus**
- **ECA**
- **hornlog2bsia**
  - **hornlog**
  - Modules
    - Layer
      - equality
      - Neg
      - Naf
      - testcases
      - update
      - attribute
      - attachment
      - eca
      - events