MARS: Modular Active Rules in the Semantic Web

Erik Behrends, Oliver Fritzen, Wolfgang May, Franz Schenk

Institut für Informatik, Universität Göttingen, Germany

Supported by the EU Network of Excellence

Further Contributors:

Heiko Kattenstroth, Tobias Knabke, Elke von Lienen, Daniel Schubert, Sebastian Spautz, Thomas Westphal

Joint Work with: José Júlio Alferes, Ricardo Amador

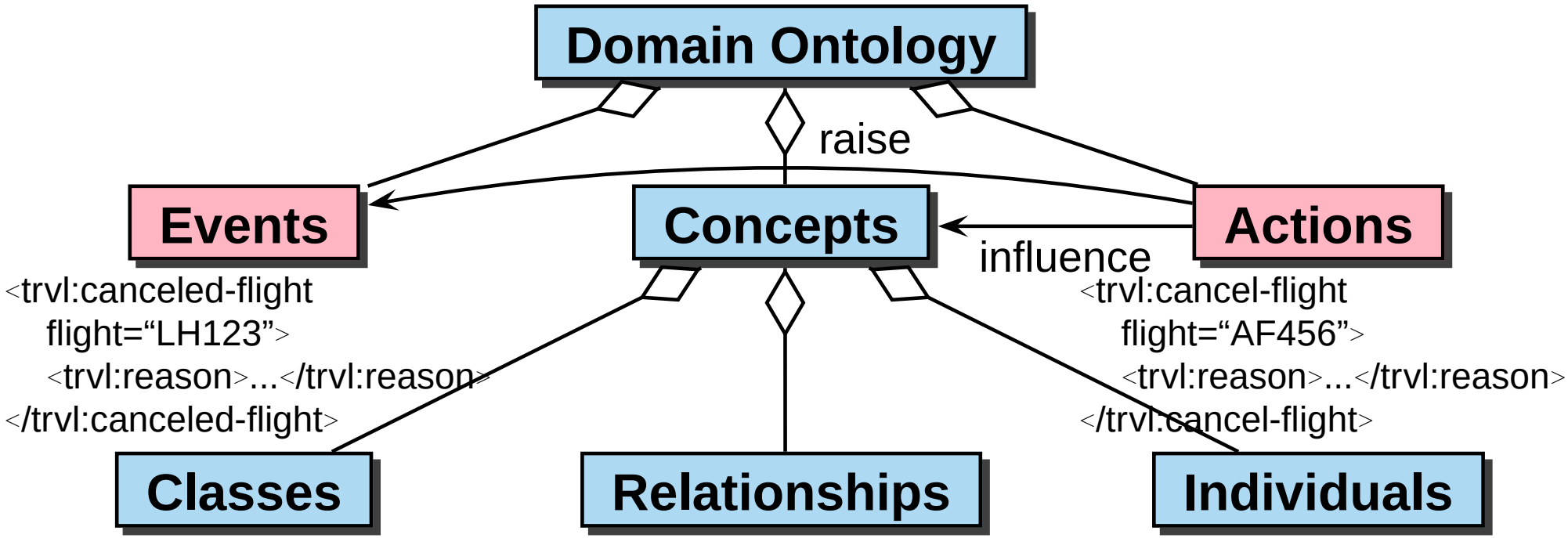
Thesis:
There is not a single formalism/language for describing and implementing behavior in the Semantic Web.

Hypothesis:
Semantical approaches (i.e., not “programming”, but based on an ontology of behavior) follow the *Event-Condition-Action* paradigm.

Justification:
We show that a general framework approach with modular components covers many existing concepts that will prove useful for behavior in the Semantic Web.

Adding Events and Actions to the Ontologies

- Domain languages also describe behavior:

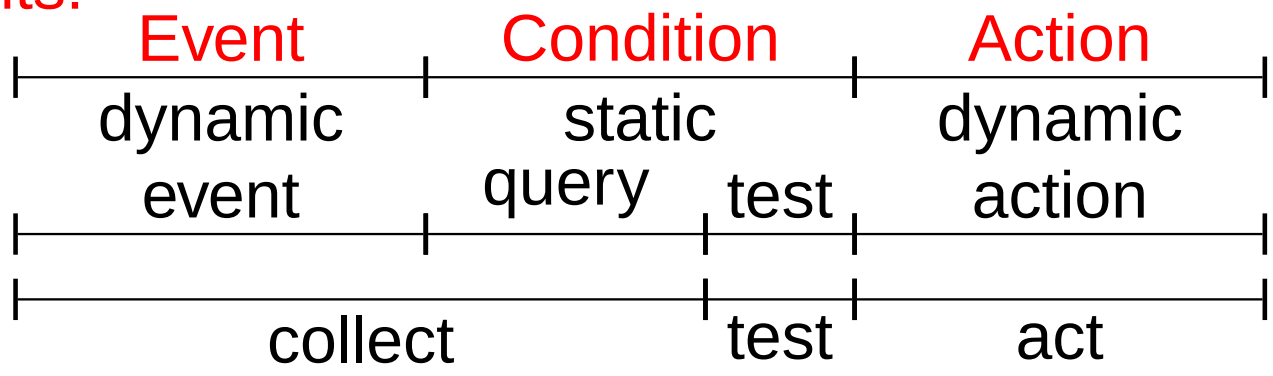


- Ontology of behavior aspects
- correlate and axiomatize actions, events and state
- combine application-dependent semantics with generic concepts/patterns of behavior

Analysis of Rule Components

"On Event check Condition and then do Action"

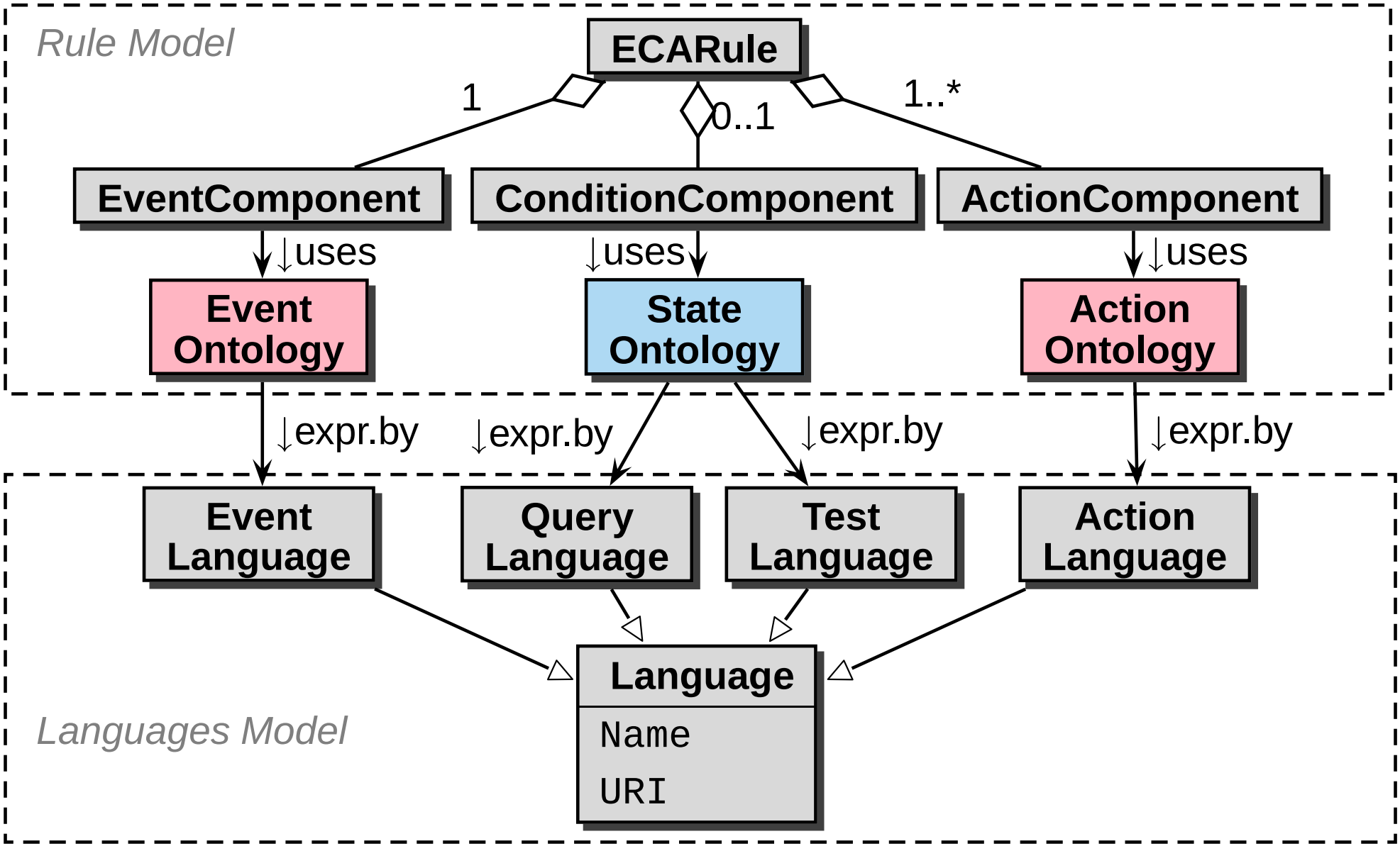
Rule Components:



- Event: detect just the dynamic part of a situation,
- Query: then obtain additional information by queries,
- Test: then evaluate a *boolean* condition,
- Action: then actually do something.

⇒ Modular concepts with Web-wide services

Modular ECA Concept: Rule Ontology

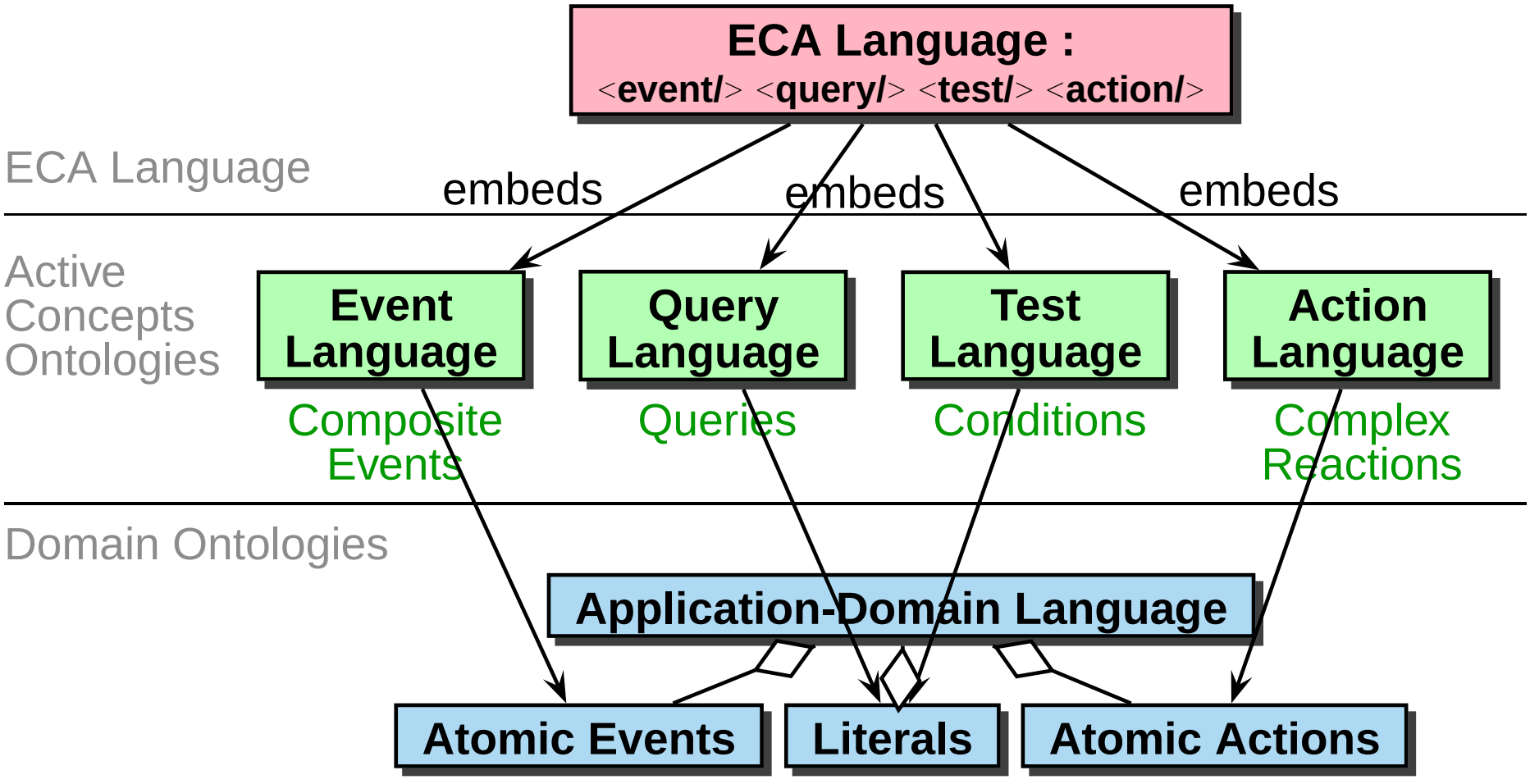


Rule Markup: ECA-ML

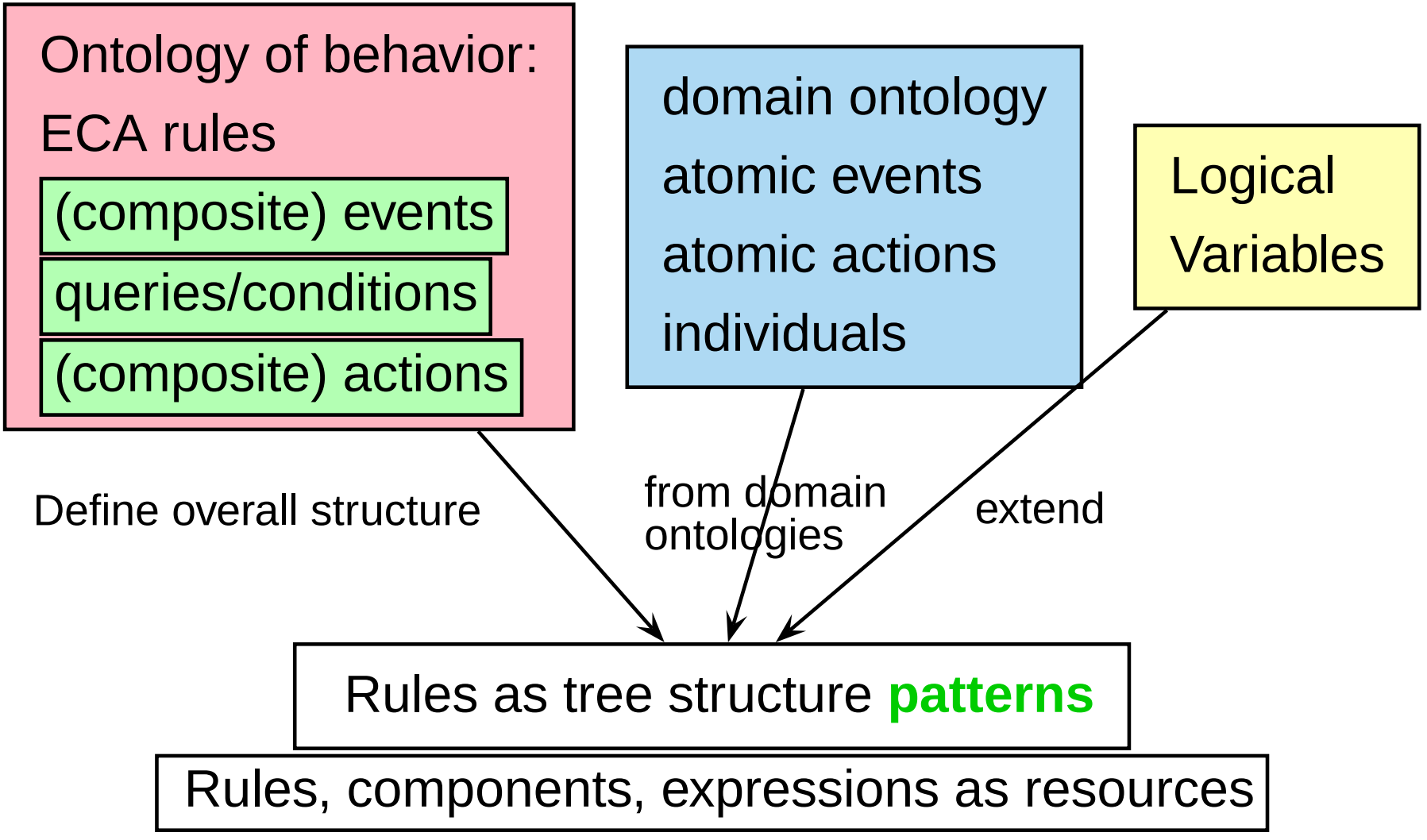
<!ELEMENT rule (event,query*,test?,action+) >

<eca:Rule *rule-specific attributes*>
  <eca:Event *identification of the language* >
    event specification, probably binding variables
  </eca:Event>
  <eca:Query *identification of the language* > <!-- there may be several queries -->
    query specification; using variables, binding others
  </eca:Query>
  <eca:Test *identification of the language* >
    condition specification, using variables
  </eca:Test>
  <eca:Action *identification of the language* > <!-- there may be several actions -->
    action specification, using variables, probably binding local ones
  </eca:Action>
</eca:Rule>

Embedding of Languages



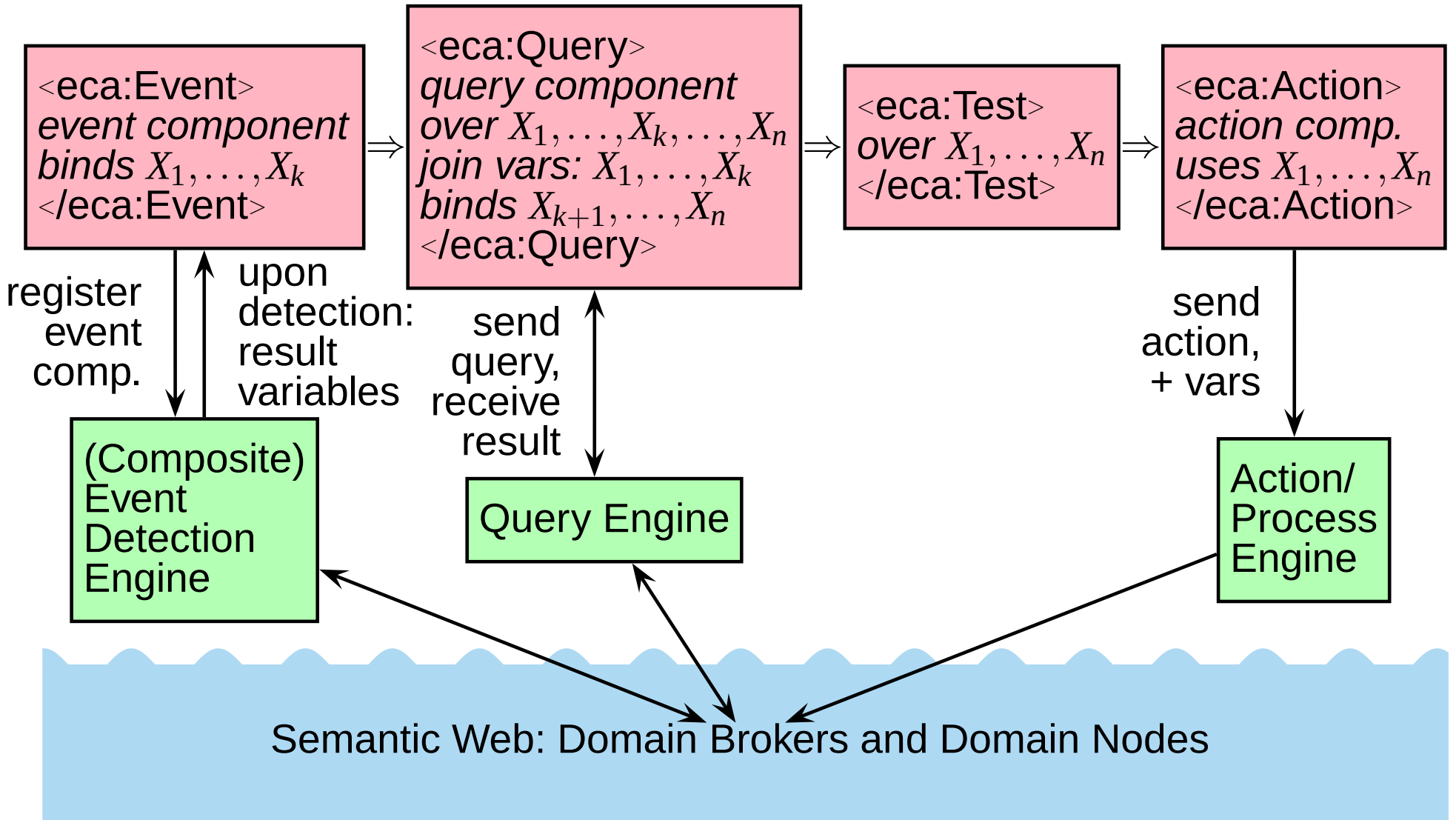
ECA Rule Markup



Binding and Use of Variables in ECA Rules

$action(X_1, \ldots, X_n) \leftarrow$

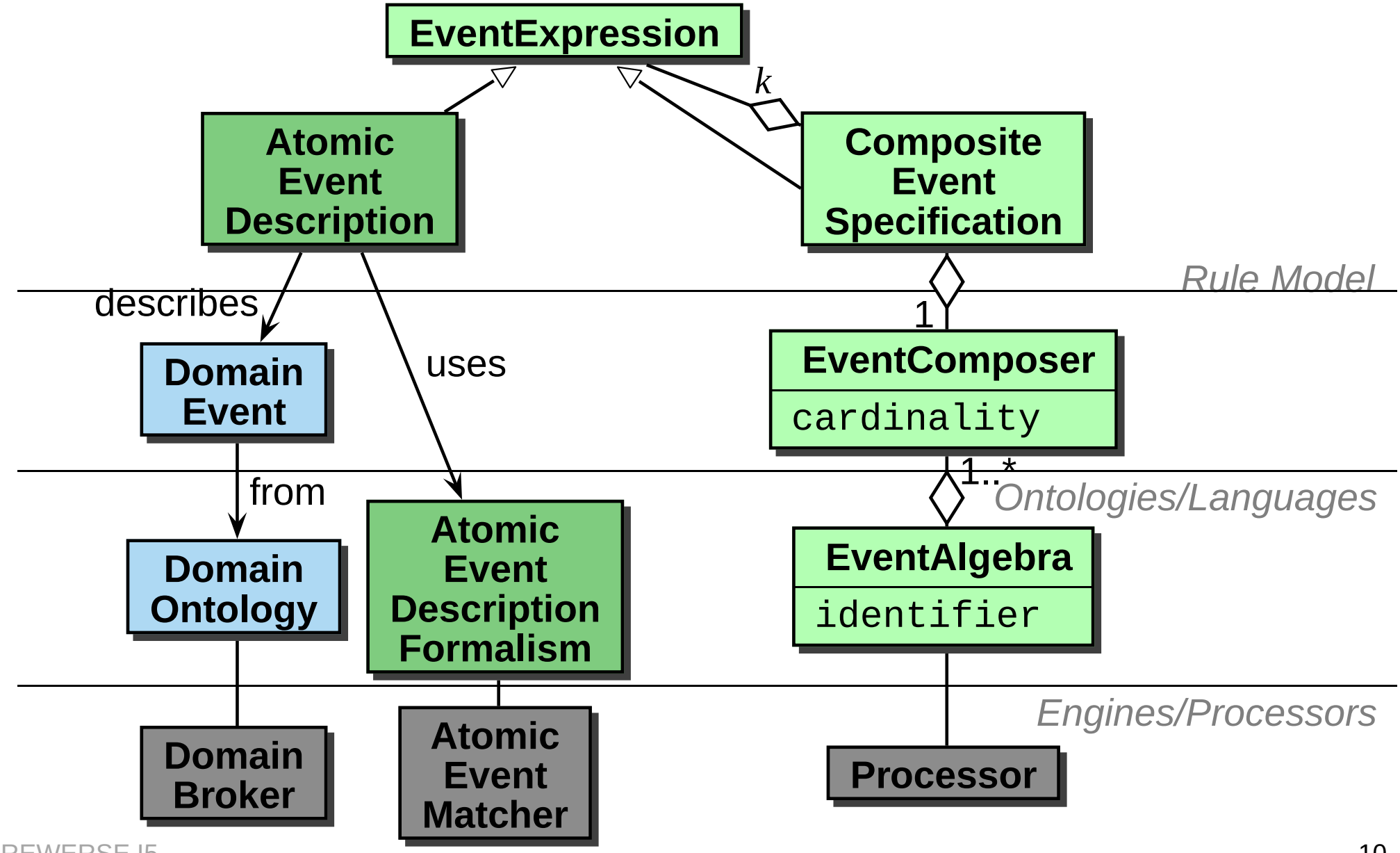
$event(X_1, \ldots, X_k),\ query(X_1, \ldots, X_k, \ldots X_n),\ test(X_1, \ldots, X_n)$



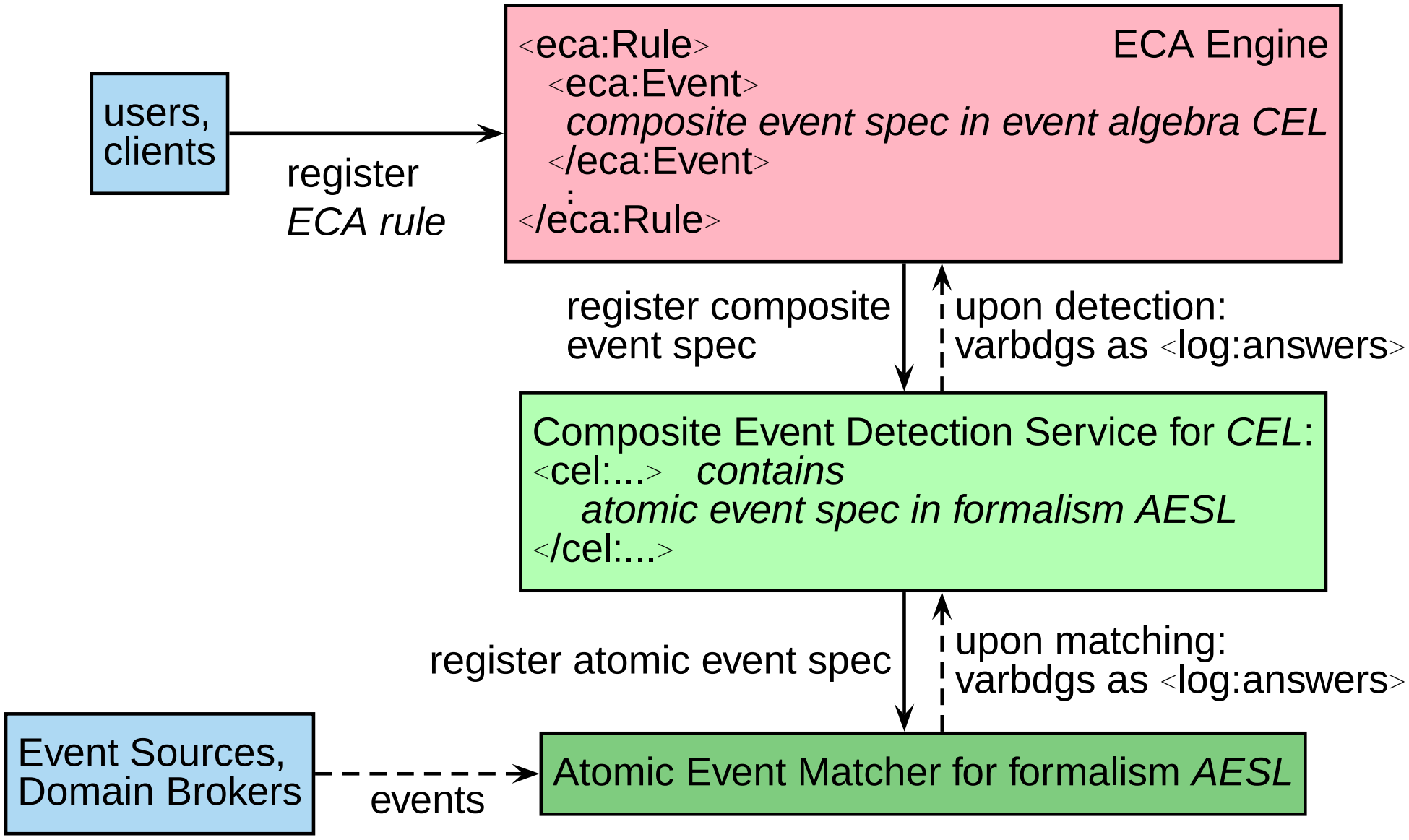
Rule Markup: Example (Stripped)

```
<!ELEMENT Rule (Event, Query*, Test?, Action+) >
<eca:Rule xmlns:travel="http://www.travel.com">
  <eca:Event xmlns:snoop="http://www.snoop.org">
    <snoop:Sequence>
      <travel:delayed-flight flight="{$flight}"/>
      <travel:canceled-flight flight="{$flight}"/>
    </snoop:Sequence>
  </eca:Event>
  <eca:Query bind-to-variable="email">
    <eca:Opaque language="http://www.w3.org/xpath">
      doc("http://xml.lh.de")/flights[code="{$flight}"]/passenger/@e-mail
    </eca:Opaque> </eca:Query>
  <eca:Action xmlns:smtp="...">
    <smtp:send-mail to="$email" text="..."/>
  </eca:Action>
</eca:Rule>
```

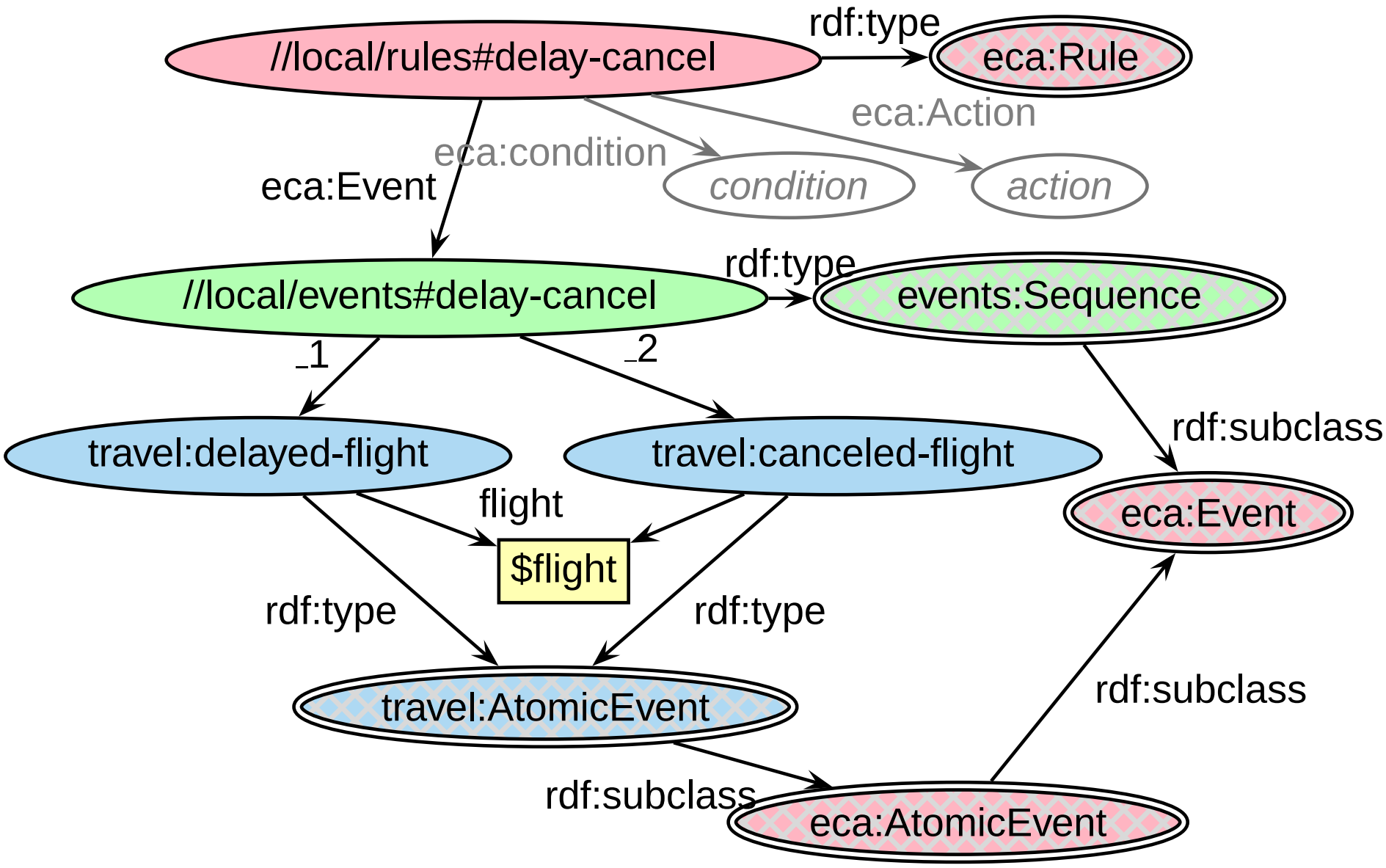
Event Expressions: Languages



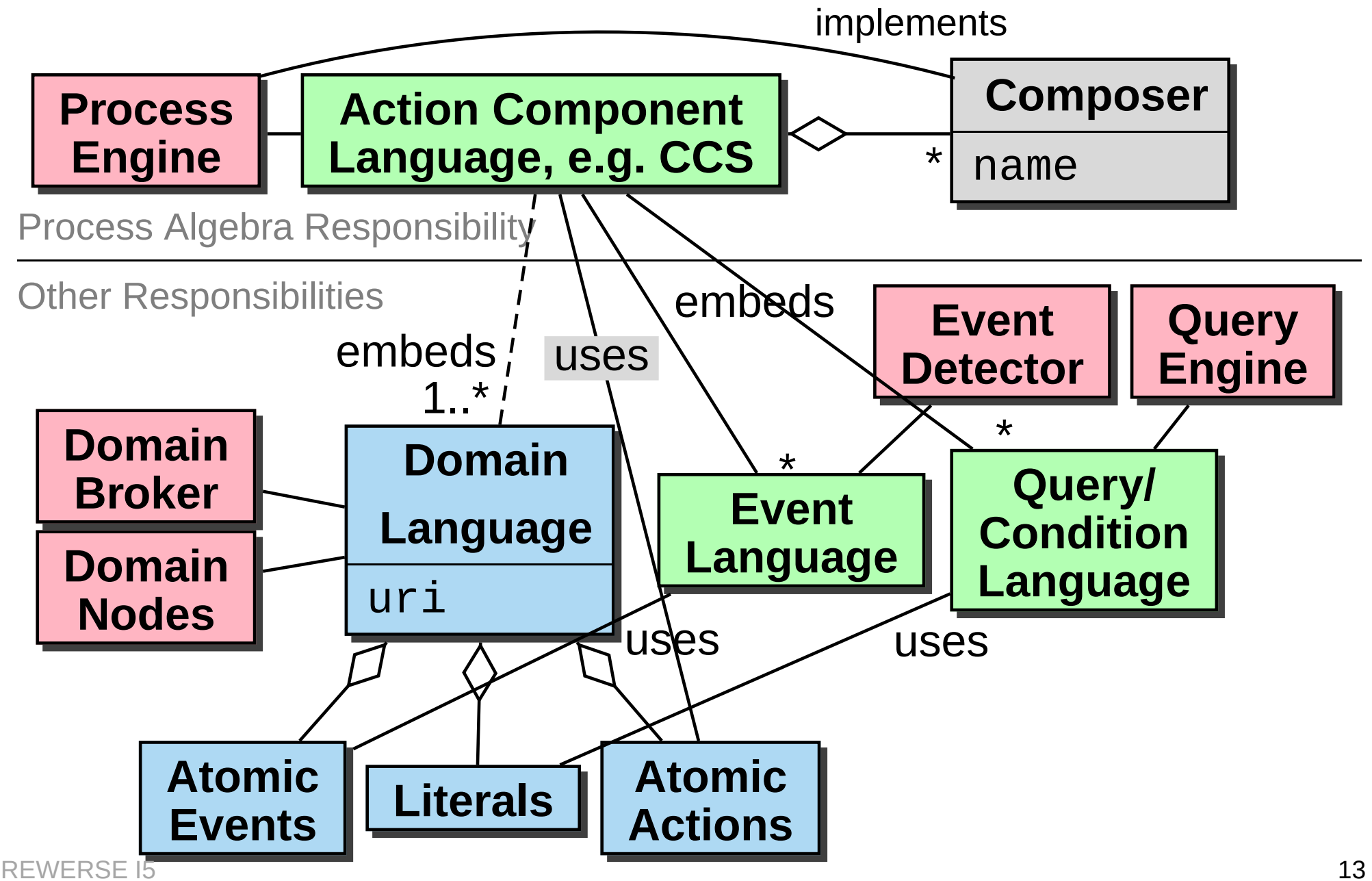
Event Detection Communication



Example as RDF

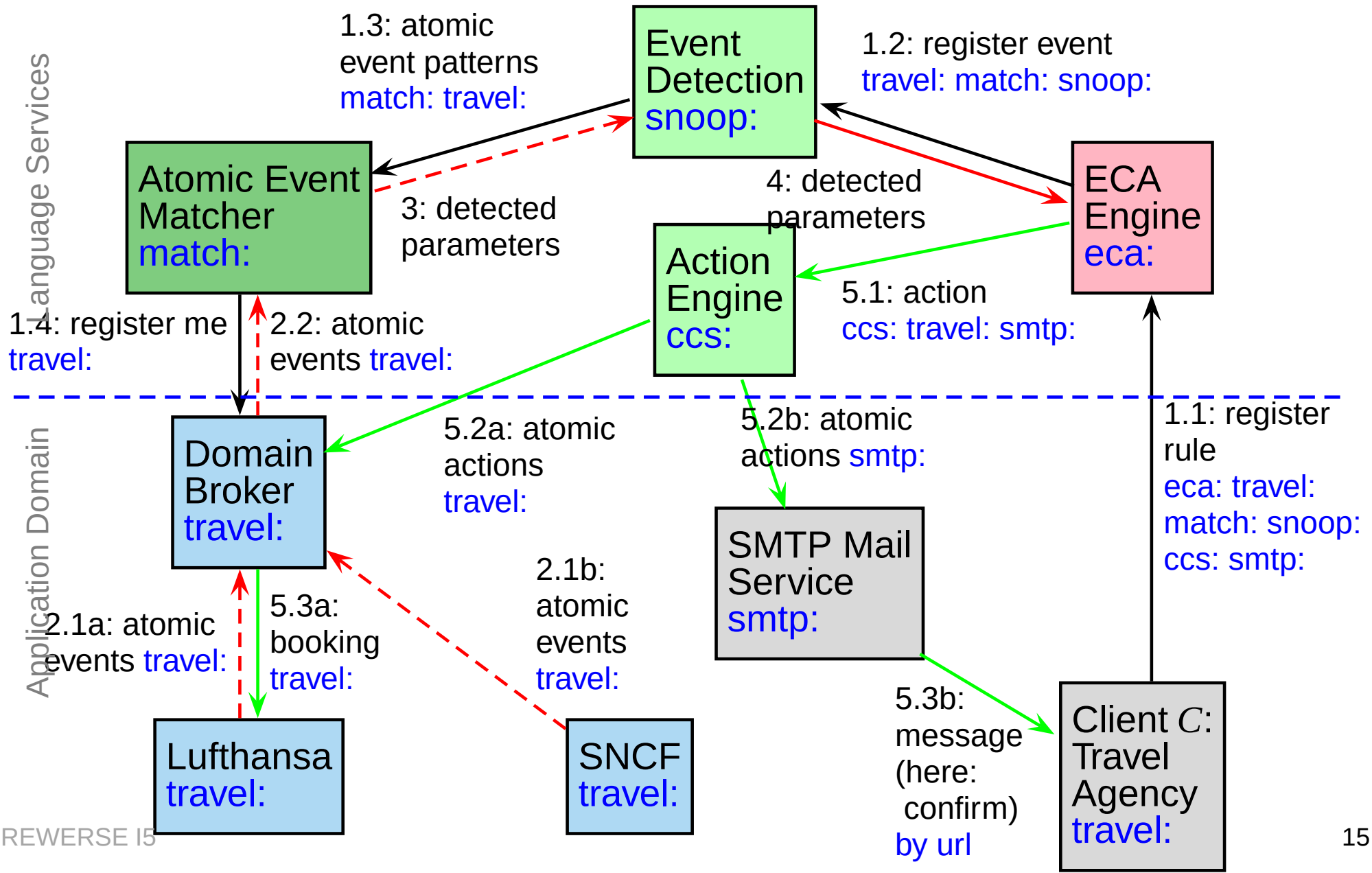


Languages in the Action Component

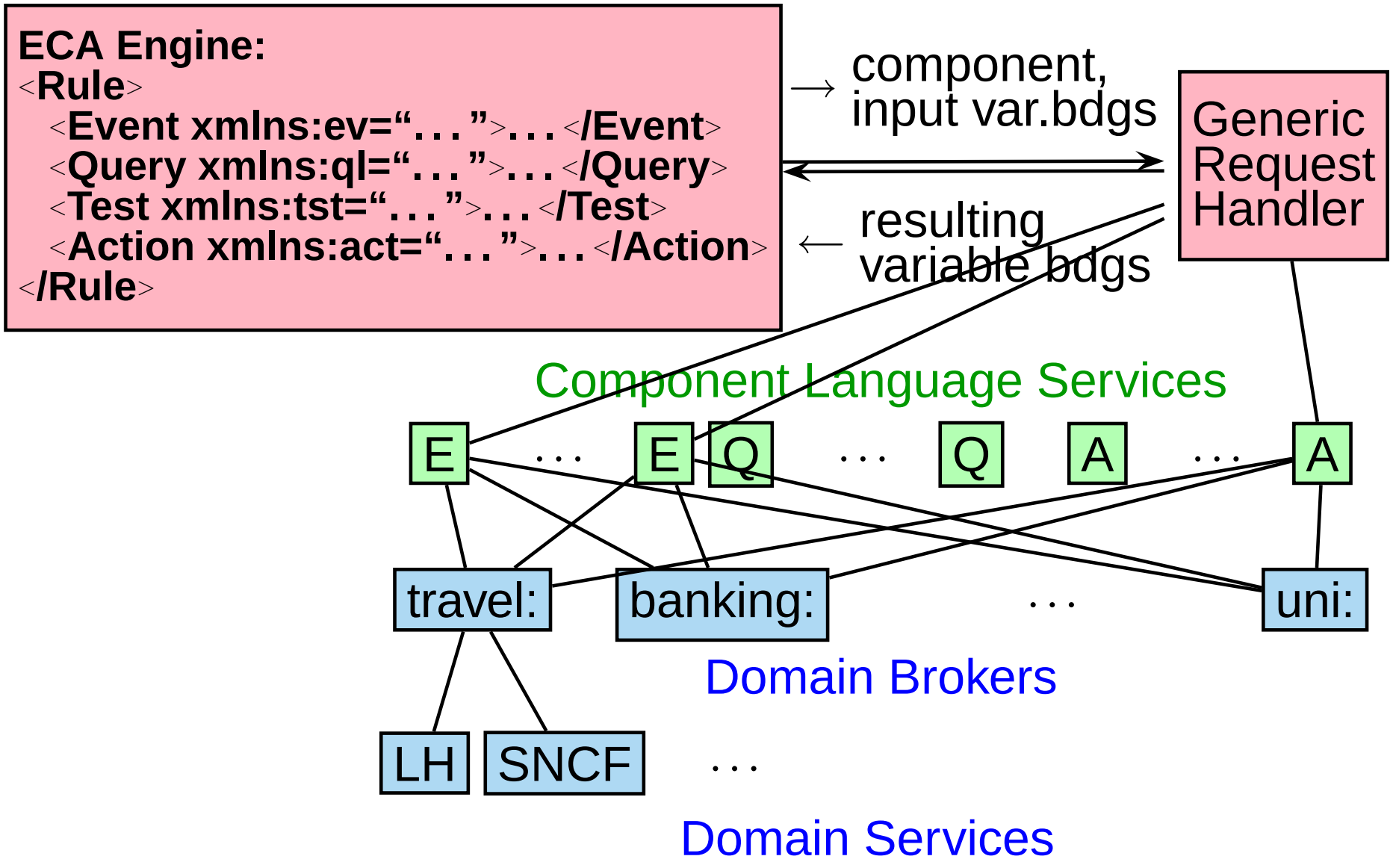


```
<eca:Rule xmlns:uni="http://www.education.de">
  <eca:Event> failed twice – binds $student ID and $course  </eca:Event>
  <eca:Query> binds e-mail addresses of the student and the lecturer </eca:Query>
  <eca:Action xmlns:ccs="...">
    <ccs:Sequence>
      <ccs:Fixpoint variables="X" index="1" localvars="$date $time $room">
        <ccs:Sequence>
          <ccs:Atomic> send asking mail to lecturer </ccs:Atomic>
          <ccs:Event> answer binds $date and $time</ccs:Event>
          <ccs:Query> any room $room at $date $time available? </ccs:Query>
          <ccs:Alternative>
            <ccs:Test> yes </ccs:Test>
            <ccs:Sequence>
              <ccs:Test> no</ccs:Test>
              <ccs:ContinueFixpoint withVariable="X"/>
            </ccs:Sequence>
          </ccs:Alternative>
        </ccs:Sequence>
      </ccs:Fixpoint>
      <ccs:Atomic> send message ($date, $time, $room) to student </ccs:Atomic>
      <ccs:Atomic> send message ($date, $time, $room) to lecturer </ccs:Atomic>
    </ccs:Sequence>
  </eca:Action>
```

Architecture



ECA Architecture



Communication of Variable Bindings

Sample XML markup for communication of a query and variable bindings:

```
<eca:Query xmlns:ql="url"
  rule="rule-id" component="component-id">
  <!-- query component -->
< eca:Query>
<log:variable-bindings>
  <log:tuple>
    <log:variable name="name" ref="URI"/>
    <log:variable name="name"> any value </log:variable>
      :
  </log:tuple>
  <log:tuple> . . . </log:tuple>
      :
  <log:tuple> . . . </log:tuple>
</log:variable-bindings>
```

Communication Component Engine → GRH

- result-bindings-pairs (semantics of expression)

```
<log:answers rule="rule-id" component="component-id">
  <log:answer>
    <log:result>
      <!-- functional result -->
    </log:result>
    <log:variable-bindings>
      <log:tuple> ... </log:tuple>
          :
      <log:tuple> ... </log:tuple>
    </log:variable-bindings>
  </log:answer>
  <log:answer> ... </log:answer>
        :
  <log:answer> ... </log:answer>
</log:answers>
```