PROVA AND PROVALETS FOR RULE-BASED SEMANTIC COMPLEX EVENT PROCESSING

Prof. Dr. Adrian Paschke
Head of Data Analytics Center (DANA)
Head of Corporate Semantic Web (CSW)
CEP IS A MATURE TECHNOLOGY WITH AN ESTABLISHED MARKET

The CEP market is expected to grow from $1,005.0 million in 2014 to $4,762.0 million in 2019. This represents a CAGR of 36.5% from 2014 to 2019.

Source: Paul Vincent, ComplexEvents.com

ResearchAndMarkets, November 2014
EXAMPLE – REACTION RULEML STANDARD

RuleML

LegalRuleML (OASIS)

Deliberation

Consumer

Reaction

Derivation

Fact

Query

Hornlog

Datalog

SBVR (OMG)

RIF (W3C)

SWRL (W3C)

HOL

FOL

FOL

Hornlog

Datalog

KR

ECAP

CEP

Trigger (EA)

Production (CA)

RIF (W3C)

PRR (OMG)

subClassOf

optional mix in of

Overlaps

Syntactic specialization of

Semantic Profiles (e.g. Consumption Policies)

Qualifications (e.g. temporal windows, event groups)

Operators (e.g. Event Algebra, Action Algebra)

http://reaction.ruleml.org

Adrian Paschke: Reaction RuleML 1.0 for Rules, Events and Actions in Semantic Complex Event Processing, Proceedings of the 8th International Web Rule Symposium (RuleML 2014), Springer LNCS, Prague, Czech Republic, August, 18-20, 2014
**WHAT WE DO …**

**SEMANTIC COMPLEX EVENT PROCESSING (SCEP)**

- **handle** and **analyze** real-world event streams with domain-specific semantic background knowledge

**DANA’s SCEP Market**, e.g. Emergency Response Services, Transportation and Logistics, Algorithmic Trading, Fraud Detection, Behavioral Analytics, Movement Analytics, Predictive Maintenance, Sentiment Analysis, Sport Analytics, Competitor Analytics, Supply Chain Optimization, Revenue Assurance, Pollution Detection, …
EXAMPLE: SEMANTIC CEP - FILTER PATTERN

Filter Pattern:
Stocks of companies, which have production facilities in Europe and produce products out of metal and have more than 10,000 employees.

Event Stream – stock quotes
{(Name, “OPEL”) (Price, 45) (Volume, 2000) (Time, 1)}
{(Name, “SAP”) (Price, 65) (Volume, 1000) (Time, 2)}

Semantic Knowledge Base
{(OPEL, is_a, car_manufacturer),
 (car_manufacturer, build, Cars),
 (Cars, are_build_from, Metall),
 (OPEL, hat_production_facilities_in, Germany),
 (Germany, is_in, Europe)
 (OPEL, is_a, Major_corporation),
 (Major_corporation, have, over_10,000_employees)}
BUT CEP IS NOT JUST „SIMPLE“ STREAM FILTERING …

EPTS CEP REFERENCE ARCHITECTURE: FUNCTIONAL VIEW

<table>
<thead>
<tr>
<th>Event Reaction</th>
<th>0..*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Preparation</td>
<td>0..*</td>
</tr>
<tr>
<td>Event Production</td>
<td>0..*</td>
</tr>
<tr>
<td>Event Consumption</td>
<td>0..*</td>
</tr>
</tbody>
</table>

### Event Reaction
- Assessment, Routing, Prediction, Discovery, Learning

### Complex Event Detection
- Consolidation, Composition, Aggregation

### Event Preparation
- Identification, Selection, Filtering, Monitoring, Enrichment

### Event Analysis
- Analytics, Transforms, Tracking, Scoring, Rating, Classification

### Event Production
- Publication, Retrieval

### Event Consumption
- Dashboard, Apps, External Reaction

### State Management

---

**Event Process Monitoring, Control**

**Design time**
- Event Definitions
- Event and Complex Event Model
- (Pattern, Control, Rule, Query, RegEx, etc)

**Run time**
- Event Actions
- Event Correlations and patterns
- Event Computations
- Event Selections
- Event Production/Consumption

**Administration**
- Process Updates
- Resource Utilization
- High Availability
- Security
- Start/Stop

---

see: Adrian Paschke, Paul Vincent, Alexandre Alves, Catherine Moxey: Tutorial on advanced design patterns in event processing. DEBS 2012: 324-334;
EXAMPLE: SEMANTIC ENRICHMENT OF EVENT STREAM

Event Stream: e1, e2, e1

Raw Events

Semantic Enrichment

Derived Events: e3, e4, e3

Knowledge Base

Final Processing on EPN

Complex Events
EXAMPLE: SEMANTIC EVENT QUERY PRE-PROCESSING

Event Stream

\[ e_1 \quad e_2 \quad e_1 \]

Raw Events

Event Stream

Rewrite Simple Queries

\[ q_1 \quad q_2 \quad q_3 \]

Complex Query Pre-Processing

Complex Query

Knowledge Base

Final Processing distributed on a network of processing Agents

Complex Events

Event Processing Network
EXAMPLE: SAMPLING THE EVENT STREAM

- Sample $m$ items **uniformly** from event data
- Useful: approximate costly computation on small sample

- Challenge: don’t know how large total input is
- Useful: sample the data included in a window
- Window = Reservoir in reservoir sampling algorithm

Kia Teymourian, Adrian Paschke: Plan-Based Semantic Enrichment of Event Streams. ESWC 2014: 21-35
OUR IMPLEMENTATION: PROVA RULE ENGINE

- Java JVM based, open source rule language for reactive agents
- Leverages declarative ISO Prolog standard extended with (event, message) reaction logic, type systems (Java, Ontologies), query built-ins (e.g. SQL, SPARQL, RDF Triples, Xquery), dynamic Java integration.
- Combines declarative, imperative (object-oriented) and functional reactive programming styles
- Designed to work in distributed Enterprise Service Bus and OSGi environments
- Supports strong, loose and decoupled interaction with reactive messaging rules
- Compatible with rule interchange standards such as Reaction RuleML

© DANA @ Fraunhofer FOKUS
Event Production
Application Time
Publication, Retrieval

Event Preparation
Identification, Selection, Filtering,
Monitoring, Enrichment

Complex Event Detection
Consolidation, Composition,
Aggregation

Event Analysis
Analytics, Transforms, Tracking,
Scoring, Rating, Classification

Event Reaction
Assessment, Routing, Prediction,
Discovery, Learning

Event Consumption
Dashboard, Apps,
External Reaction

State Management

see: Adrian Paschke, Paul Vincent, Alexandre Alves, Catherine Moxey: Tutorial on advanced design patterns in event processing. DEBS 2012: 324-334;
% Select stock ticker events from stream "S&P500"
% Each received event starts a new subconversation (CID) which further processes the selected event (select)

rcvMult(CID,stream,"S&P500", inform, tick(S,P,T)) :-
  select(CID,tick(S,P,T)).

% Indefinitely (count=-1) receive further ticker events from other streams that follow the previous selected event in event processing group (group=g1). If the price differs for the same stock at the same time [T1=T2, P1!=P2] then ...

select(CID,tick(S,P1,T1)) :-
  @group(g1) @count(-1)
  rcvMsg(CID,stream, StreamID ,inform, tick(S,P2,T2)) [T1=T2, P1!=P2] %guard: if at same time but different price,
  println ("["Suspicious:",StreamID, tick(S,P2,T2)]," ").
see: Adrian Paschke, Paul Vincent, Alexandre Alves, Catherine Moxey: Tutorial on advanced design patterns in event processing. DEBS 2012: 324-334;
% stream1 is trusted but stream2 is not, so one solution is found: X=e1

@src(stream1) event(e1).
@src(stream2) event(e2).

%note, for simplicity this is just a simple fact, but more complicated rating, trust, reputation policies could be defined
trusted(stream1). %only event from „stream1“ are trusted

ratedEvent(X):-
  @src(Source) %scoped reasoning on @src
  event(X) [trusted(Source)]. %guard on trusted sources

:-solve(ratedEvent(X)). % => X=e1 (but not e2)
**Event Reaction**
Assessment, Routing, Prediction, Discovery, Learning

**Complex Event Detection**
Consolidation, Composition, Aggregation

**Event Analysis**
Analytics, Transforms, Tracking, Scoring, Rating, Classification

**Event Preparation**
Identification, Selection, Filtering, Monitoring, Enrichment

**Event Production**
Application Time Publication, Retrieval

**Event Consumption**
Dashboard, Apps, External Reaction

**State Management**
% This reaction operates indefinitely. When the timer elapses (after 25 ms), the groupby map Counter is sent as part of the aggregation event and consumed in or group, and the timer is reset back to the second argument of @timer.

groupby_rate() :-
  Counter = ws.prova.eventing.MapCounter(), % Aggr. Obj.
  @group(g1) @timer(25,25,Counter) % timer every 25 ms
  rcvMsg(XID,stream,From,inform,tick(S,P,T)) % event
      [IM=T,Counter.incrementAt(IM)]. % aggr. operation

% receive the aggregation counter in the or reaction
  @or(g1) rcvMsg(XID,self,From,or,[Counter]),
  ... <consume the Counter aggreation object>.
rcvMsg(XID, Process, From, event, ["A"]) :-
  fork_b_c(XID, Process).

fork_b_c(XID, Process) :-
  @group(p1) rcvMsg(XID, Process, From, event, ["B"]),
  execute(Task1), sendMsg(XID, self, 0, event, ["D"]).

fork_b_c(XID, Process) :-
  @group(p1) rcvMsg(XID, Process, From, event, ["C"]),
  execute(Task2), sendMsg(XID, self, 0, event, ["E"]).

fork_b_c(XID, Process) :-
  % OR reaction group "p1" waits for either of the two
  % event message handlers "B" or "C" and terminates the
  % alternative reaction if one arrives
  @or(p1) rcvMsg(XID, Process, From, or, _).
PROVALETS – REST-BASED PROVA MICROSERVICES

Docker Container

Provalet Microservice

Smart Agent

Karaf Container

OSGi Container

Provalet Microservice

Smart Agent
IMPLEMENTATION

- **Provalet Microservices**
  - REST-based Microservices deployed to standardized containers (OSGi, Docker)
  - Integrated into Workflows as Inference Logic Nodes in Event Processing Pipelines

- **Prova Rule-based Agents**
  - Java-based Prova rule engine
  - OSGi bundel runs in Provalet

- **Workflows / Pipelines**
  - User-friendly Workflow Modelling Language

Control-flow pattern examples:

- Sequence
- Exclusive Choice
- Synchronization
- Multi-Choice
- Deferred Choice

Dataflow pattern examples:

- Task to Task (Global data store)
- Task to Environment (Push Oriented)
- Task Precondition – Data Existence
- Event-Based Task Trigger
SUMMARY SEMANTIC CEP: SELECTED BENEFITS

- Event data becomes declarative knowledge while conforming to an underlying formal semantics
  - e.g., supports automated semantic enrichment and mediation between different heterogeneous domains and abstraction levels
- Reasoning over situations and states by event processing agents
  - e.g., a process is executing when it has been started and not ended
  - e.g. a machine is in maintenance when a predictive maintenance event / abnormal event occurred
  - e.g. a plane begins flying when it takes off and it is no longer flying after it lands
- Better understanding of the relationships between events e.g., temporal, spatial, causal, .., relations between events, states, activities, processes
  - e.g., a service is unavailable when the service response time is longer than X seconds and the service is not in maintenance state
  - e.g. a landing starts when a plane approaches. During landing mobile phones must be switched off
- Declarative rule-based processing of events and reactions to situations
  - Semantically grounded reaction rules
Prof. Dr. Adrian Paschke
Head of Data Analytics Center (DANA)
http://www.data-analytics-center.org/
adrian.paschke@fokus.fraunhofer.de

Head of Corporate Semantic Web
http://www.corporate-semantic-web.de
paschke@inf.fu-berlin.de

Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
www.fokus.fraunhofer.de

Freie Universität Berlin
Königin-Luise-Str. 24
14195 Berlin
FURTHER READING – RULEML AND REACTION RULEML

- Adrian Paschke: Reaction RuleML 1.0 for Rules, Events and Actions in Semantic Complex Event Processing, Proceedings of the 8th International Web Rule Symposium (RuleML 2014), Springer LNCS, Prague, Czech Republic, August, 18-20, 2014
  http://dx.doi.org/10.1007/978-3-642-16289-3_15

  http://dx.doi.org/10.1007/978-3-642-16289-3_15

- Adrian Paschke, Harold Boley, Zhili Zhao, Kia Teymourian and Tara Athan: Reaction RuleML 1.0: Standardized Semantic Reaction Rules, 6th International Conference on Rules (RuleML 2012), Montpellier, France, August 27-31, 2012
  http://link.springer.com/chapter/10.1007%2F978-3-642-32689-9_9
  http://www.slideshare.net/swadpasc/reaction-ruleml-ruleml2012paschketutorial

  http://www.igi-global.com/chapter/rule-markup-languages-semantic-web/35852

  http://www.igi-global.com/book/handbook-research-emerging-rule-based/465

- Adrian Paschke and Harold Boley: Rule Responder: Rule-Based Agents for the Semantic-Pragmatic Web, in Special Issue on Intelligent Distributed Computing in International Journal on Artificial Intelligence Tools (IJAIT), Vol. 20,6, 2011
  https://www.researchgate.net/publication/220160498
FURTHER READING – SURVEYS AND TUTORIALS

- Adrian Paschke, Paul Vincent, Alexandre Alves, Catherine Moxey: Tutorial on advanced design patterns in event processing. DEBS 2012: 324-334;
  http://www.igi-global.com/book/handbook-research-emerging-rule-based/465
  http://link.springer.com/chapter/10.1007%2F978-3-642-04985-9_8
  http://link.springer.com/chapter/10.1007%2F978-3-642-24908-2_17
- Jon Riecke, Opher Etzion, François Bry, Michael Eckert, Adrian Paschke, Event Processing Languages, Tutorial at 3rd ACM International Conference on Distributed Event-Based Systems. July 6-9, 2009 - Nashville, TN
  http://www.slideshare.net/opher.etzion/debs2009-event-processing-languages-tutorial
  http://www.igi-global.com/chapter/rule-markup-languages-semantic-web/35852
FURTHER READING – RULE-BASED SEMANTIC CEP

- Corporate Semantic Web – Semantic Complex Event Processing
  http://www.corporate-semantic-web.de/semantic-complex-event-processing.html
- Kia Teymourian, Adrian Paschke: Plan-Based Semantic Enrichment of Event Streams. ESWC 2014: 21-35
- Adrian Paschke: Semantic Rule-Based Complex Event Processing. RuleML 2009: 82-92
  http://link.springer.com/chapter/10.1007%2F978-3-642-04985-9_10
  http://dl.acm.org/citation.cfm?id=1804769
- Kia Teymourian, Adrian Paschke: Towards semantic event processing. DEBS 2009
FURTHER READING – PROVA

- Provalets

- Prova Rule Engine http://www.prova.ws/

- Prova 3 documentation

- Journal: Adrian Paschke and Harold Boley: Rule Responder: Rule-Based Agents for the Semantic-Pragmatic Web, in Special Issue on Intelligent Distributed Computing in International Journal on Artificial Intelligence Tools (IJAIT), V0l. 20,6, 2011