# Towards The Reactive Web

Master Thesis Report

Dominic Bosch
Departement Mathematics and Computer Science
University of Basel

October 21, 2013

**Abstract.** t.b.d.

## 1 Introduction

t.b.d.

## 2 Related Work

### 2.1 Rules Languages

In this section we examine different existing rule languages with respect to a simple use case. We want the rule language react on the receipt of an email (event), check for a distinct email address (condition) and store it in a remote location, via a Web API (action). The email only contains the parts we require for this use case (the sender and a subject). A JSON representation of the email would be:

```
{
  "event": "email",
  "sender": "sender@mail.com",
  "subject": "An important message!"
}
```

### 2.1.1 RDF & XML

An early ECA Rule Language for XML repositories [4] was postulated in 2003 and was picked up by many researches afterwards. It was designed to react on insert and delete events within XML repositories and as an action change XML documents.

```
ON INSERT document('inbound_queue.xml')/mails/mail
IF $delta/sender[.='sender@mail.com']
DO DELETE document('inbound_queue.xml')/mails/mail;
   LET $api = resource("www.webapi.com") IN
   INSERT ($api, newcontent,
      <content>New mail: {$delta/subject}</content>)
```

Now apart from implementing a rules engine, we would also need to add an XML document event manager which interpretes and pushes events into the XML file *inbound_queue.xml*. Then again this instance would interprete the ouptuts of the ECA engine, which would theoretically manifest in other XML documents, and produce meaningful actions on remote hosts. This wouldn't be an architecture which has its focus on the solution of our use case and, as a result, add complexity and create an unnecessary overhead.

### 2.1.2 Notation 3

To make the lengthy RDF definitions smaller and more readable, Notation 3 [1] was designed and announced in 2005. Through the implies operator(=>) an ”event” can be connected to an ”action”, both expressed in RDF’s subject, predicate, object notation, which makes the expression of ECA rules a complicated and not very intuitive task. A solution to our use case would somewhat look like:

```
{ ?x :event "email". ?x :sender "sender@mail.com" } => { :webapi :newcontent ?x}
```

It’s obvious that these rules are meant to express relations between entities and is not really suitable for our use case and would need another interpreter to infer the actions. But concepts and ideas of the work that was done in these consortias could eventully still find influence into our solution.

#### 2.1.3 XChange/Xcerpt

#### 2.1.4 JSON Rules

t.b.d. *JSON Rules* [3]

#### 2.1.5 R2ML

t.b.d.

#### 2.1.6 OWL

t.b.d.

#### 2.1.7 SWRL

t.b.d.

#### 2.1.8 ETALIS

t.b.d.

### 2.1.9 ESPER

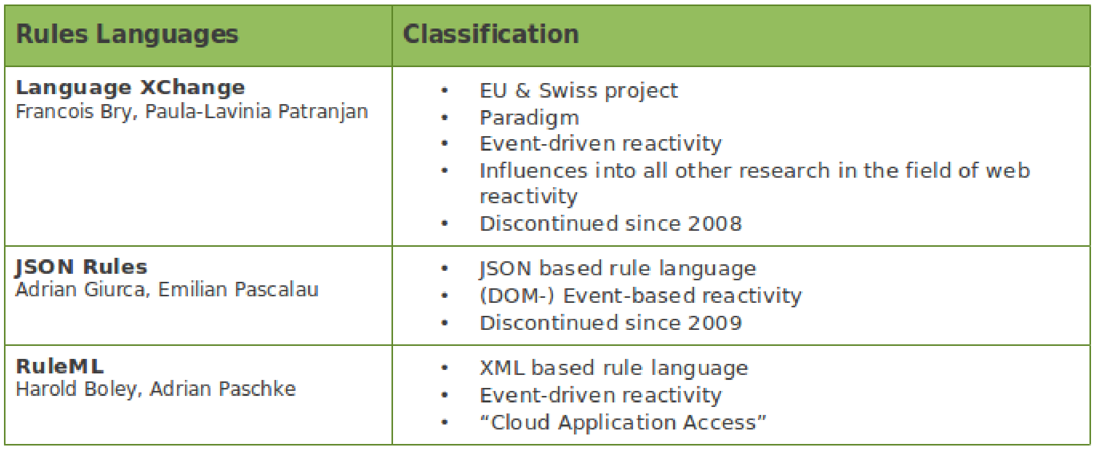
t.b.d.

### 2.1.10 KRL

t.b.d.

### 2.1.11 (Reaction) RuleML

*RuleML* [2]



| Rules Languages | Classification |
|---|---|
| **Language XChange**<br>Francois Bry, Paula-Lavinia Patranjan | • EU & Swiss project<br>• Paradigm<br>• Event-driven reactivity<br>• Influences into all other research in the field of web reactivity<br>• Discontinued since 2008 |
| **JSON Rules**<br>Adrian Giurca, Emilian Pascalau | • JSON based rule language<br>• (DOM-) Event-based reactivity<br>• Discontinued since 2009 |
| **RuleML**<br>Harold Boley, Adrian Paschke | • XML based rule language<br>• Event-driven reactivity<br>• “Cloud Application Access” |

Figure 1: Examined Rules Languages

### 2.1.12 (OO) jDrew

t.b.d.

### 2.1.13 Prova

t.b.d.

#### 2.1.14 Kinetic

t.b.d.

#### 2.1.15 Drools Fusion

t.b.d.

#### 2.1.16 Rule Responder

t.b.d.

## 3 Conclusion

t.b.d.

## 4 Future Work

t.b.d.

## References

[1] Tim Berners-Lee. Notation 3 Logic. `http://www.w3.org/DesignIssues/Notation3.html`, 2005. Accessed: 2013-10-21.

[2] Harold Boley. The RuleML family of web rule languages. In *Principles and Practice of Semantic Web Reasoning*, pages 1–17. Springer, 2006.

[3] Adrian Giurca and Emilian Pascalau, Json rules. *Proceedings of the Proceedings of 4th Knowledge Engineering and Software Engineering, KESE*, 425:7–18, 2008.

[4] George Papamarkos, Alexandra Poulovassilis, Ra Poulovassilis, and Peter T. Wood. Event-Condition-Action Rule Languages for the Semantic Web. In *In: Workshop on Semantic Web and Databases*, pages 309–327, 2003.

## Appendix

t.b.d.

## A ECA Rules Engine Resources

### A.1 Node.js Rules Engine Code

#### A.1.1 ecaserver.js

```
'use strict';
var express = require('express');
var qs = require('querystring');
var engine = require('./ecainference');
```