Publ. Astron. Obs. Belgrade No. 96 (2017), 375 - 378 *Contributed Paper*

USE OF COMPLEX EVENT PROCESSING ENGINES IN TIME DOMAIN ASTRONOMY

V. VUJČIĆ[1,2], J. ALEKSIĆ[1,3], S. NEŠKOVIĆ[2] and D. JEVREMOVIĆ[1]

[1]*Astronomical Observatory Belgrade, Serbia*
E–mail: jaleksic, veljko, darko@aob.rs

[2]*Faculty of Organizational Sciences, Univ. of Belgrade, Serbia*

[3]*Faculty of Mathematics, Univ. of Belgrade, Serbia*

Abstract. The expansion of volume and complexity of events that need to be processed in high-growth industries (such as finance, telecommunications, banking, medicine) created the need for new paradigms and tools. CEP engines offer scalability which cannot be easily achieved through previous standard practices, loose coupling between event processing logic and the mainstream application code, and decoupling between event producers and consumers. In this document, basic concepts of CEP are introduced and it is discussed how they could be utilized in astronomical contexts.

1. INTRODUCTION

Large, deep and fast sky surveys of the future, such as LSST, will detect significant amount of near-real time transient astronomical events. In case of LSST, it is estimated that every visit will produce ∼2.000 alerts on average (up to 40.000), which sums up to ∼2 million alerts per night[1]. Software which will be handling these alerts will have to be able to act like a human expert, on a scale that is impossible for a human to ingest; it will need to know how to classify and discover, to dispatch and "ask for a second opinion", all in efficient and scalable manner. On top of that, a scientist should be able to describe inference mechanisms using a language with high level of abstraction. There are several CEP open-source solutions which could potentially be tailored to satisfy astronomical needs.

2. MECHANISMS OF EVENT PROCESSING

2. 1. FROM SIMPLE TO COMPLEX EVENTS

In an event-driven system, such as health monitoring, algoritmic trading, banking fraud detection or sky survey, we define a simple event as a *discrete incidence inside of a domain that system is capable of detecting.* After an image of the portion of the sky has been captured through the lens, data has been read from the CCD, image has been compared to a template image, astrometric and photometric properties have been calculated, we end up with a bunch of simple events: instances of astronomical objects in a particular moment of time which are either *1)* new to us *2)* known but have changed their properties significantly enough. Such simple events might be just false positives or a sign that something important is going on.

By putting related simple events into a common context and by applying pattern matching mechanisms we might eventually come to a conclusion that something important *has* happened. We call such a notable occurence a complex event: *an event that summarizes, represents or denotes a set of other events*[2].

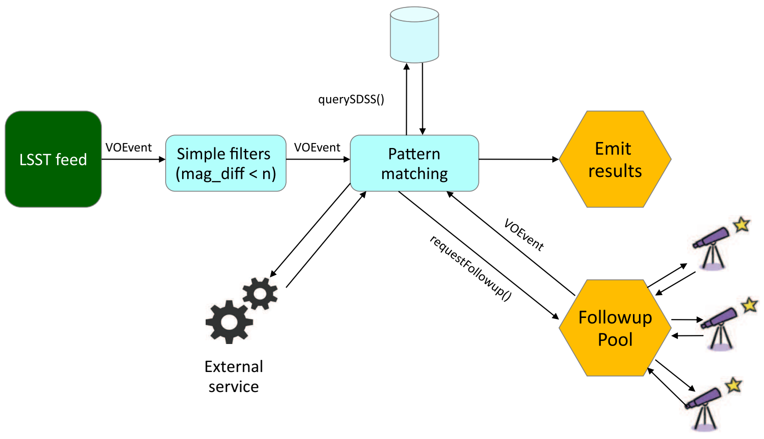

Figure 1: A simplified event processing network for a sky survey feed.

2. 2. EVENT PROCESSING LOGIC

An event processing agent (EPA) should be able to apply inference mechanism and temporal reasoning in order to give a high-quality conclusion whether a complex events has happened. Event processing includes concepts such as[3]:

- Sliding windows based on time or number of events.
- Applying spatial, spatiotemporal, segmentation or state oriented context.
- Filtering, transformation (splitting, aggregating, projecting, translating...) and pattern detection
- Enriching events from external service/db.

In a typical scenario, an interesting simple event would trigger EPA to open a temporal window, i.e to “wait” for the next event which pertains to the same astronomical object. Depending on the inference rules, EPA could apply aggregation function or statistical knowledge on a number of events related by context.

2. 3. BUILDING AN EVENT PROCESSING NETWORK

As shown on Figure 1, an event processing network may look like this: LSST feed is emitting simple events (in form of VOEvent XML objects) which are pulled through simple filtering agent which output a subset of interest; another agent applies pattern matching, and may consult historical data (e.g SDSS database), external classification engine or ask a followup pool if there are available telescopes; after a conclusion, complex event is dispatched to a channel which emits results to subscribers.

3. CURRENT STATUS OF TECHNOLOGY

3. 1. VOEVENT COMMUNITY

A significant amount of work has been done in VOEvent community. IVOA defined a standardized XML message structure for exchanging information about transient events - VOEvent[4]. Although XML is transport-agnostic, there is a TCP-based transport protocol for transmitting VOEvents - VTP[5] which supports concepts such as node roles. Dakota and Comet are VCP implementations available for free use, along with Skyalert - event stream collector, filter and distributor. There is a number of events streams available for subscription [6].

3. 2. OPEN SOURCE CEP

There is several open source CEP engines on the market which implement (subsets of) concepts of event processing stated earlier. Esper and WSO2 Siddhi implement SQL-based language (select... from... where...) while JBoss Drools works on top of a rule-base engine (when... then...). The market is dynamic with lots of merges and acquisitions from larger, commercial brands. Esper was used in internal monitoring and analysis for ATLAS experiment at CERN[7].

4. CONCLUSIONS

VOEvent community needs a tool which will be able to handle events from heterogenous and dynamic environment on a much larger scale than today. Skyalert has plenty of concepts implemented (multiple input streams, event portfolios, filtering, custom trigger funcions...) but lacks support for temporal reasoning (sliding windows) and doesn't scale well. XML format for VOEvent might be outdated, with substantial overhead and signing issues.

Open source CEP solutions claim to handle millions of events per second on single server with commodity hardware. Analyzing syntatic capabilities of Esper and Drools, we came to the conclusion that Esper offers stronger support for sliding windows and pattern matching by aggregation of attributes.

First step will be to build a prototype based on Esper which would plug-in to existing event streams and LSST alert simulator. This prototype should ingest events at realistic rates to test scalability and capabilities of declarative language syntax for scientific use. Further reasearch will include “factories” of CEP engines where a researcher could build her/his own engine using DSL statements at even higher domain-specific level of abstraction.

Acknowledgments

The authors are grateful for the support provided by Ministry for Education and Science of Republic of Serbia through project III 44002. "Astroinformatics: Application of IT in Astronomy and Close Fields".

References

Jurić, M. et al.: 2013, Large Synoptic Survey Telescope Data Products Definition Document (LSE-163).
Luckham, D., Schulte, R. et al.: 2011, Event Processing Glossary Version 2.0 http://www.complexevents.com/2011/08/23/event-processing-glossary-version-2-0/ ISBN: 9781935182214
Etzion, O., Niblett P.: 2010, Event Processing in Action. Manning Publications Co.
Seaman, R. et al.: 2011, Sky Event Reporting Metadata Version 2.0 http://www.ivoa.net/documents/VOEvent/
Allan, A., Denny, R. B.: 2009, VOEvent Transport Protocol Version 1.1 http://www.ivoa.net/documents/Notes/VOEventTransport/
http://skyalert.org/streams/
Kazarov, A., Lehmann Miotto G., Magnioni L.: 2012, The AAL project: automated monitoring and intelligent analysis for the ATLAS data taking infrastructure J. Phys.: Conf. Ser. 368.