Evaluating RuleCore as Event Processing Network Model

Irina Astrova[1], Arne Koschel[2], Sebastian Kobert[2], Jan Naumann[2], Tobias Ruhe[2] and Oleg Starodubtsev[2]

[1]*Department of Software Science, School of IT, Tallinn University of Technology, Akadeemia tee 21, 12618 Tallinn, Estonia*

[2]*Faculty IV, Department of Computer Science, Hannover University of Applied Sciences and Arts, Ricklinger Stadtweg 120, 30459 Hannover, Germany*

Keywords: Event Processing Network (EPN), Event Processing Network Model, RuleCore.

Abstract: Our work is motivated primarily by the lack of standardization in the area of Event Processing Network (EPN) models. We identify general requirements for such models. These requirements encompass the possibility to describe events in the real world, to establish temporal and causal relationships among the events, to aggregate the events, to organize the events into a hierarchy, to categorize the events into simple or complex, to create an EPN model in an easy and simple way and to use that model ad hoc. As the major contribution, this paper applies the identified requirements to the RuleCore model.

1 INTRODUCTION

A typical database management system (DBMS), e.g., a relational database is designed to store, query and manipulate static data presented as rows and columns. In contrast, a data stream management system (DSMS) is used to process continuous feeds of data in a way analogous to a DBMS. For example, whereas relational databases use Structured Query Language (SQL) to query and manipulate static data, DSMSs use a modified form of SQL, e.g., Event Processing Language (EPL) to work more efficiently with real-time streaming data.

Event processing in a DSMS takes place on just one stream of data. Complex event processing (CEP) takes place on more than one stream of data, where the streams may be of varying types. For example, a CEP engine may combine processing of data streams from one or more sensors with static data stored in a relational database.

CEP, especially within smart grid solutions, can be leveraged to promote safety and innovation in the domain of portable micro CHPs (block power plants) (Thomas, 2007). Smart grid solutions specialize in providing high quality, smart grid technologies and products to the marketplace that improve the electrical distribution infrastructure. It is essential to ensure error-free operation of block power plants. This is where CEP can help.

2 MOTIVATION

It is a common theme that Internet of Things (IoT) is all about data. But a variety of heterogeneous sensors can exhibit a continuous stream of data, which can represent thousands to millions of events per second.

An event is “a significant change of state” or “a happening of interest at a certain point in time in a certain location” (Luckham, 2002). Event Processing Networks (EPNs) can be seen as generalized software systems that allow for the processing of events. However, EPN models lack standardization.

Table 1: Requirements for EPN models.

ID	Requirement
1	Provide a specific way to model events with their inherent attributes as a central component of the system
2	Allow for a “real-world” description of events
3	Allow for simple, complex or aggregated events
4	Allow for relativity of events and their temporal and causal relationships
5	Allow for a description of the flow of events through the system
6	Allow for a description of the components in EPNs as well as their properties and used event patterns
7	Allow for a description of the components outside of the system, which is interacted with
8	Provide a means to create an EPN model in an easy way and to use the created model on-the-fly

As an attempt to fill in a gap in this field, in our previous work (Koschel et al., 2017), we identified

Astrova, I., Koschel, A., Kobert, S., Naumann, J., Ruhe, T. and Starodubtsev, O.
Evaluating RuleCore as Event Processing Network Model.
DOI: 10.5220/0008192302970300
In *Proceedings of the 15th International Conference on Web Information Systems and Technologies (WEBIST 2019)*, pages 297-300
ISBN: 978-989-758-386-5

general requirements for EPN models (see Table 1). These requirements help to analyze EPN models with respect to their overall modelling and usage capabilities. As such, the identified requirements provide a valuable tool for comparison of different EPN models. As an example of EPN model, we considered RuleCore.

3 RULECORE

RuleCore is a CEP engine that is based on the concept of loosely coupled, event-driven components, which communicate with each other indirectly using a publish/subscribe model. Data are internally stored down in a relational database. This allows for an offline analysis, visualization, simulation and reporting of collected data using traditional database tools.

3.1 Typical Application Areas

RuleCore is typically used in:

- Application monitoring;
- Equipment timing constraint;
- Route compliance;
- Fraud detection.

3.2 Architecture

Figure 2 gives an overview of RuleCore architecture, which generally looks like an EPN model.

3.2.1 Event Model

An event model is formally specified using XML Schema descriptions. XML itself is used for event representations. All events have a common structure with mandatory attributes, which are defined in the root element. They have the following characteristics:

- Events are immutable (meaning that they cannot be modified after they have been created);
- Events have a globally unique id;
- Events have a type;
- Events happen at a given point in time indicated by a time stamp;
- Events can be caused by other events;
- Events are processed as ordered streams.

3.2.2 Event Producers

Events are generated and published by an event source immediately whenever some activity is detected at the source. An event source is normally a business system and is responsible for publishing events and transferring them into RuleCore, possibly using different kinds of technical infrastructure for integration and event transport. Multiple event sources can use multiple events channels to convey the events in RuleCore.

3.2.3 Event Aggregation

A combination of events is specified using the concept of situations. A situation is a formal description of a combination of events as they happen over time along with optional timing restrictions.

3.2.4 Event Consumers

Event consumers consume events and remove them from RuleCore.

3.2.5 Event Transport Protocols

Within RuleCore semantics and representation of events are totally transport-independent. Rules are evaluated exactly in the same way, independently of the chosen event transport protocol.

3.2.6 Active Rules

Events can be either inbound or outbound. Events sent into RuleCore are inbound and events destined for delivery to systems outside RuleCore are outbound. This difference is purely logical and is not visible in the events as such. It is a feature of an event depending on the context in which it is used.

RuleCore uses an approach based on active rules. Each rule is basically an active event processing entity reacting to specific combinations of inbound events over time. When a specific combination of events is found, an outbound reaction event is published with a summary of those events.

3.3 Event I/O Logical Model

The event I/O module can be seen as a bi-directional pipeline, each pipe consisting of a chain of event processing sub-steps or sub-components. Each processing step filters, refines or enriches the event before passing it on to the next step (see Figure 1).

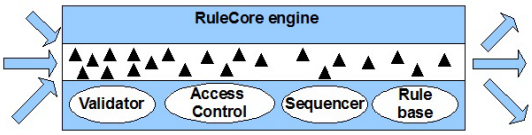

Figure 1: RuleCore I/O logical model.

3.3.1 Event Inputs

An event input receives all or a subset of the events from one or more sources. Each event input is a contributor to the inbound events stream.

3.3.2 Event Validator

An event validator makes basic format and syntax checks on the inbound event messages. Events that are well-formatted and parsable XMLs are forwarded to the valid events stream. Events that fail the checks are forwarded to the invalid events stream instead.

3.3.3 Event Access Control

Received events are checked for having the correct security token. The events that contain the correct token are forwarded to the authorized events stream. Events that fail the checks are sent to the access error stream instead.

3.3.4 Event Sequencer

Operating inside a distributed and heterogeneous environment usually implies the use of several unsynchronized time sources and varying event transmission times. As a result of this, events arriving at the event I/O module will in nearly all cases not be picked up in the same order as they were originally emitted. RuleCore imposes one unconditional requirement on the inbound event stream, namely, that all events in the stream are arranged in order by ascending time.

To allow for receiving events from unsynchronized event sources, an event sequencer is introduced. It makes sure the requirement of an ordered event stream is fulfilled at all times by putting the inbound events on hold for a certain amount of time before releasing them to RuleCore. The maximum amount of time for which the event is put on hold is a configurable parameter, making it easy to adapt the sequencing to the particular operating environment.

Inbound events whose time stamp is older than the maximum age are instantly rejected. Inbound events with a time stamp in the future may or may not be kept for sequencing – this is a configurable option. The effect of using the event sequencer is thus twofold: (1) events will be forwarded to RuleCore in the order of ascending time; and (2) events arriving at RuleCore will be delayed by the predefined amount of time.

4 CONCLUSION

While in our previous work (Koschel et al., 2017, Koschel et al., 2018) we applied the identified requirements for EPN models to the EPiA model (Etzion, Niblett, 2011) and the Business Event Modeling Notation (BEMN) model (White, 2006), in this paper we demonstrated the mapping of the same requirements to the RuleCore model. In particular, we examined RuleCore architecture and modelling capabilities as well as its typical application areas.

Our examination showed that RuleCore fully meets Requirements 1, 3–6. For example, RuleCore distinguishes between simple and complex (or aggregated) events, which consist of multiple simple events. Each event in RuleCore gets a time stamp during their creation. Thanks to this stamp, temporal relations among events and their order can be easily identified. Furthermore, one can model events and their inherent attributes on a high level, which enables a real-world description of events, thereby meeting Requirement 2. The typical application scenarios for RuleCore showed this as well. In addition, a large set of connectors for event producers and consumers help to meet Requirement 7. However, some costs are caused by the overall complexity of the engine, which makes it somehow difficult to create an EPN model. As a result, RuleCore meets Requirement 8 only partially.

ACKNOWLEDGEMENT

Irina Astrova's work was supported by the Estonian Ministry of Education and Research institutional research grant IUT33-13.

REFERENCES

Luckham, D., 2002. *The Power of Events*, Addison Wesley Pub Co Inc, USA.

Koschel, A., Astrova, I., Kobert, S., Naumann, J., Ruhe, T., Starodubtsev, O., 2017. Towards Requirements for

Event Processing Network Models. In *Proc. 8th Intl. Conf. on Information, Intelligence, Systems, Applications (IISA), Larnaca, Cyprus.* IEEE.

Koschel, A., Astrova, I., Kobert, S., Naumann, J., Ruhe, T., Starodubtsev, O., 2018. On Requirements for Event Processing Network Models Using Business Event Modeling Notation. In *Proc. 2018 Computing Conf., SAI 2018 (London), Part 1, v. 858 of Advances in Intelligent Systems and Computing, pp. 756-762.* Springer.

White, S., 2006. *Process Modelling Notations and Workflow Patterns.* https://web.archive.org/web/20100706013817/, http://www.bpmn.org/Documents/Notations_and_Workflow_Patterns.pdf [acc. 2019]

Etzion, O., Niblett, P., 2011. *Event Processing in Action*, Manning, Stanford, CT 06901.

Thomas B., 2007. Mini-Blockheizkraftwerke (Mini-Block Power Plants), Vogel, Würzburg, Germany.

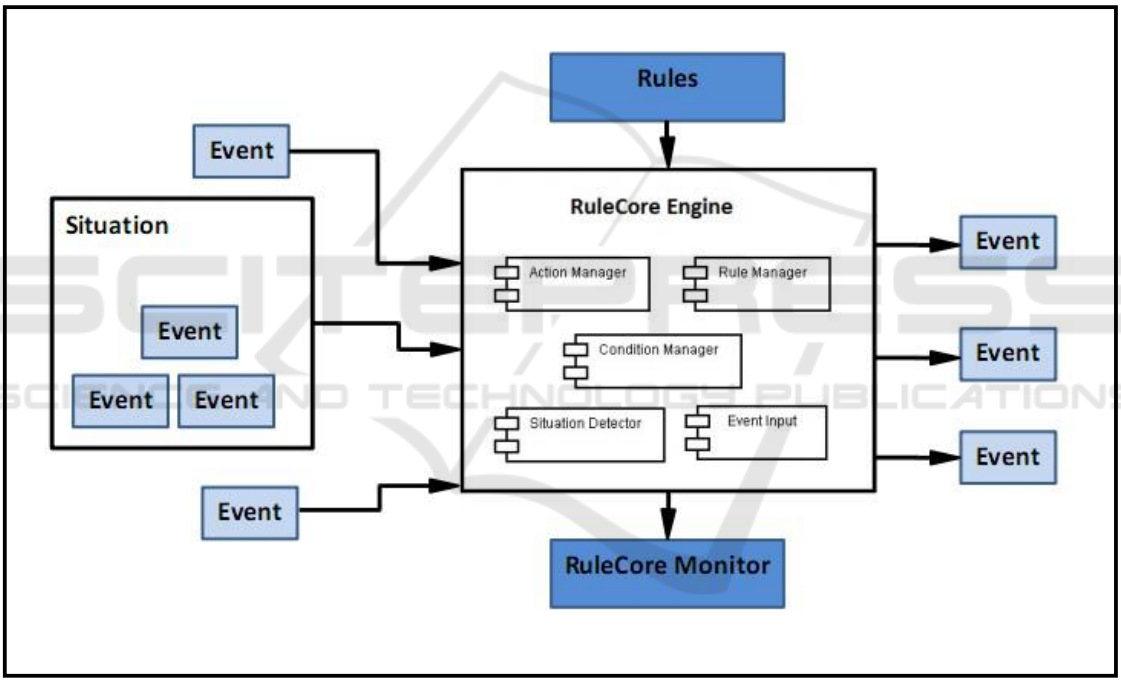

Figure 2: RuleCore architecture and event flow.