EVENT PROCESSING IN ASSET MANAGEMENT

Maja Škrjanc[1], *Klemen Kenda*[1], *Gašper Pintarič*[2]
Artificial Intelligence Laboratory, Jožef Stefan Institute[1]
Špica International d.o.o.[2]
Jamova cesta 39, 1000 Ljubljana, Slovenia
Tel: +386 1 4773528; fax: +386 1 4773851
e-mail: maja.skrjanc@ijs.si

ABSTRACT

This paper presents the conceptual architecture of the system for optimization of maintenance of assets that are equipped with adequate sensors. Beside the conceptual architecture we also present our implementation in the telecommunications use case, where we developed a system that helps optimizing technician response time and reduces effort needed for resolving problems with maintaining the mobile telephony base stations.

1 INTRODUCTION

Maintenance of remote assets is a task that can be significantly optimized, when the correct data about the state of the asset can be provided. This means that the assets have to be equipped with appropriate sensors to measure the state of the system, relevant external data needs to be provided and a pipeline for the streaming sensor data needs to be in place. With these prerequisites fulfilled, a system can be developed to support event detection and fast and efficient response time by technicians in case of failures or other alarms.

The market for applications of Complex Event Processing (CEP) is estimated to grow enormously across various industry verticals like banking and healthcare in the next 5 years. By using CEP technology, organizations can monitor and predict compliance risks in advance, ensure smoothness of business processes and remove inconsistencies [8].
Therefore, some of the software providers such as TIBCO, Oracle, IBM and many others are already providing CEP technologies in their software product portfolio. These solutions however come in packages that require demanding calibration and final implementation process. These types of solutions also do not support bottom up approach, which is crucial for a learning organization.

In the paper we describe the conceptual architecture for Asset Management (ASM) system and present a partial implementation that has been carried out in a use case in the KC OpComm project. The implementation uses a bottom up approach, since it supports learning process in the organization. It also enables external data usage and provides event triggers in a human readable form.

In section 2 we discuss the conceptual architecture, where the central part handling also event processing is taken by a dedicated stream processing engine, which we discuss in section 3. Section 4 is dedicated to event processing, where two main tasks are rule discovery and event detection. In section 5 we present our implementation of the system and conclude in section 6.

2 CONCEPTUAL ARCHITECTURE

Conceptual architecture of the system is depicted in Figure 1. It consists of analytical platform, which is essentially a stream processing engine, which includes event detection engine (ED) and analytical capabilities to aid an expert user to discover, test and refine rule definitions for events. When rule conditions are fulfilled, an alarm is triggered. Alarm consists of an asset ID, priority and a basic description. Basic description can also be generated with a natural language module.

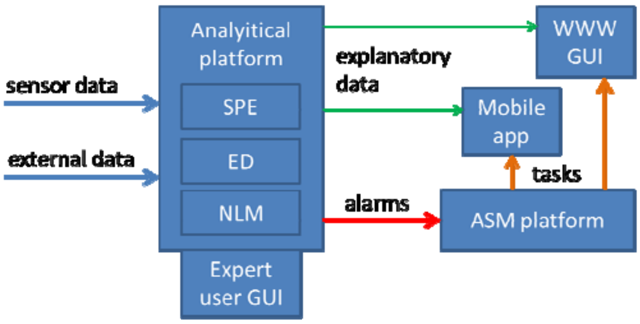

Figure 1: *Proposed conceptual architecture for event processing in asset management use case. SPE represents Stream Processing Engine, ED Event Detection and NLM Natural Language Module.*

Alarms are sent to the ASM platform. ASM platform is able to enrich alarm information with additional data, such as recent measurements or measurement aggregates from the relevant sensors or external data sources or with the detailed information about the rule that triggered the alarm. This

additional information could be describing weather conditions or electrical properties of relevant electronic devices. The information about the underlying causes of an alarm are important for further steps in the alarm resolution process. All the described information is used to create appropriate response actions.
These response actions include assignment of tasks to the responsible personell, task scheduling and other administrative actions. End user, who is typically a technician in charge of an intervention at the asset, is notified of the alarm via instant messaging (typically via SMS) and can then further investigate the issue either via a dedicated mobile app or a WWW graphical user interface.
For instance, one of the triggers could be device malfunction. In this case the maintenance personell would recieve the work order and could then execute the repair in the shortest possible time. Arrows in the left side of the figure represent incoming sensor and external data. There are two types of outputs from the analytical platform: outgoing alarms, detected by Event Processing engine and explanatory data that is retrieved, by personell in charge of the repair. Alarms are received by ASM platform, which automatically or through a system supervisor generates a work order to handle the alarm and assigns it to the technician in the field.

3 HANDLING THE SENSOR DATA STREAM

Analytical platform is a solution for streaming data processing. It can be implemented in three different ways as identified in [3]: as a traditional database management system (DBMS), as a rule engine or as a dedicated stream processing engine. For a traditional event processing engine a rule engine implementation would be the most natural solution. With the asset management scenario there is, however, a demand to access also historic data or different aggregates of the sensor measurements, either to obtain additional knowledge of the alarm or to define, discover or refine a rule as an expert user. The latter requirements suit best an SQL DBMS solution. However, the need for low latency response, fast rule evaluation and access to raw data and aggregates, make the dedicated stream processing engine option the most appropriate.
We propose to use a data layer schema as explained in [1] with an addition of two stores (tables) as depicted in Figure 2. Basic unit of the schema is a sensor measurement, which is taken by a sensor. Sensor is a device that is usually a part of a bigger electronic device that we call a node. A node carries additional meta-data about the measurement, which is for example geographical location or coexistent sensors. Every sensor also has a type defined. Type of a sensor joins a sensor measurement with the data about the measured phenomena, units of measurements, information about the used hardware, frequency of measurements and others.
Data layer consists of two additional independent stores, which are: a store with a rule definition database and a store with events. Events can be either human detected – observed (and can be used to learn the models for detecting alarms) or system detected.

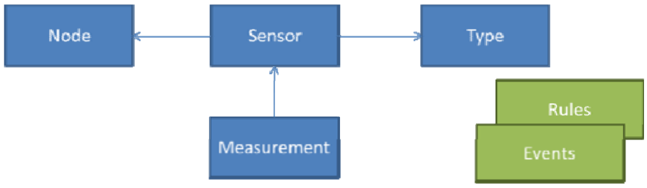

Figure 2: *Proposed data layer schema.*

4 EVENT PROCESSING

Event processing consists of three phases. Firstly, one needs to discover the rules or build classification models to detect events. Secondly, the system has to evaluate these rules on the current state of the system and lastly, an appropriate action should be taken according to the triggered alarm.
There are various possible scenarios for discovery of the rules:

1. Expert user has sufficient knowledge of the system and is able to create a rule without any support.
2. Expert user knows about a certain type of events that are happening and is using a graphical user interface to analyse the sensor and external data, different aggregates to get an idea about the behaviour of the system prior to the event. Expert user is able to test his hypothesis, evaluate its results and refine it if needed. Also a rule suggestion method would be useful in this scenario (e. g. a method that would be able to detect relevant sensors for an event or a series of events and create a rule, based on the constraints on the values of the sensor measurements at the time of events).
3. Expert user provides a list of events of a certain type and the system tries to build a model, based on the timestamp of events and corresponding sensor data.

A simple and effective GUI should be available for creating the rules.

System should be able to evaluate high number of rules on a huge amount of data in a reasonable timespan. Effective indexing and pre-processing of sensor measurement would be needed therefore.
Event detection, where rules are provided by an expert user, can only return true or false, therefore priority of the event (case that the rule is fulfilled) should be entered by the expert user. In the case of event prediction based on a model, a classification algorithm is able to return probability of the event happening based on the learning process. Expert user should define a probability threshold for triggering such an event; probability can be used also as a priority parameter, which should be determined by an expert user.
Exports of rules and data could have additional value for usage with other event processing/data mining systems. Relevant standards for exporting rules are RuleML and Datalog.

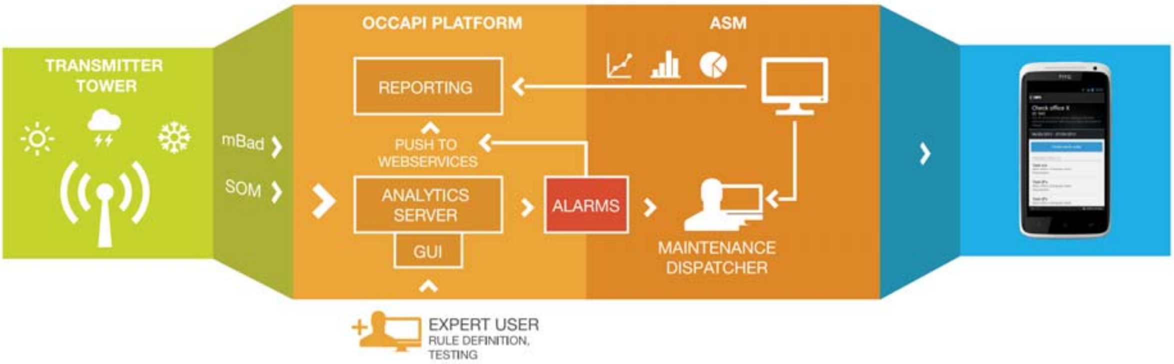

Figure 3: *Architecture of the ASM OpComm solution.*

5 IMPLEMENTATION

In this paper we are presenting an implementation of proposed conceptual architecture for event processing in asset management in Figure 1. Part of the functionality, described in previous sections has been implemented in the ASM scenario within the OpComm competence centre project.

The aim of the ASM scenario is to optimize maintenance tasks on mobile telephony base stations for a telecommunications company. Base stations have been equipped with different sensor nodes, measuring environmental data like temperature, humidity or pressure inside and outside the base station. A sensor system has also been implemented that is able to obtain the data about the electronic devices within the base station and send it to the analytical platform.

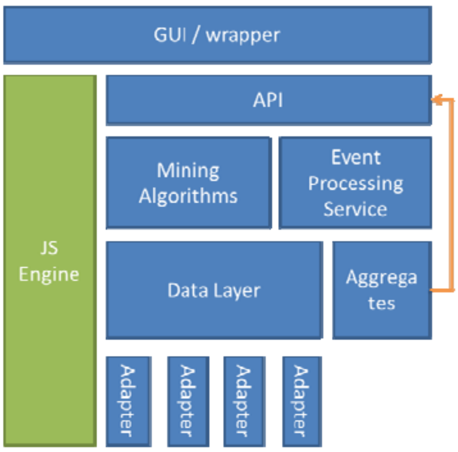

Figure 4: *EnStreaM architecture.*

Sensor sources are depicted in the left side of Figure 3. mBad is the data source about the state of electronic devices in the base stations and SOM (Smart Object Management) is responsible for the delivery of environmental data. Analytical platform consists of two interconnected components. The first component serves as a *reporting component* and is responsible for performing basic operations on the data stream and the delivery of results to the end user. *Advanced analytics* component consists of an implementation of a stream processing engine EnStreaM [7], which is based on the QMiner solution. Architecture of the EnStreaM is depicted in Figure 4.

QMiner platform provides core functionality in a set of in-house C++ libraries. A data layer schema as proposed in section 3 has been implemented. The platform already provides native support for a set of data mining algorithms and an API on top of it. Additionally, a naïve event processing engine (evaluating rules on a current state of the system after each measurement is received) has been implemented to handle event detection.

Event definitions consist of a rule, natural language event description template, priority index and an asset ID. When conditions for a rule are met, an alarm is triggered. Request is made to the ASM component, which starts a proper response procedure. Based on the asset ID a proper end user is notified with instructions, how to access detailed information about the alarm (history of measurements, etc.). A WWW GUI (see Figure 5) is available to the responsible technician, but also a mobile app (see Figure 6) [6] for even faster response and access to all the needed data from the terrain.

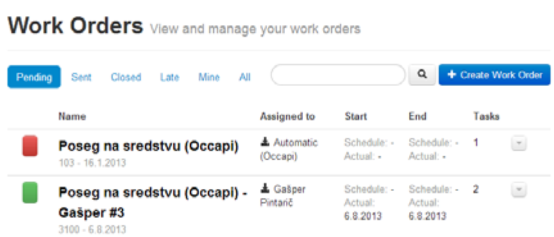

Figure 5: *Screenshot from ASM WWW client.*

5.1 Rules in the ASM use case

Telecommunication expert users have created a limited set of fairly simple rules. The justification for the usage of the simple rules the experts have given is that with a concrete simple rule there is no doubt about the source of the alarm. The end-users also pointed out the step-by-step approach.

First they need to integrate a module with analytical capabilities into the ASM solution and only in the second phase they wanted to test more complex rules and include also predictive capabilities, such as event prediction.

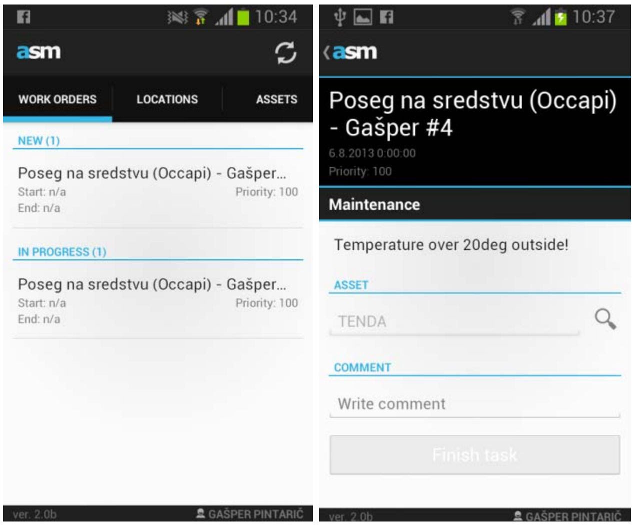

Figure 6: *Screenshots from mobile ASM client.*

Some illustrative examples of rules are listed below:

- if temperature inside the station is lower than 5°C or higher than 40°C
- if the temperature inside the emitting cell is lower than 5°C or higher than 35°C
- if the voltage on all 5V devices is lower than 4V or higher than 6V
- if the value on the WLTS device is not equal to 1186

Rules are added, edited or disabled through the expert user GUI depicted in Figure 7.

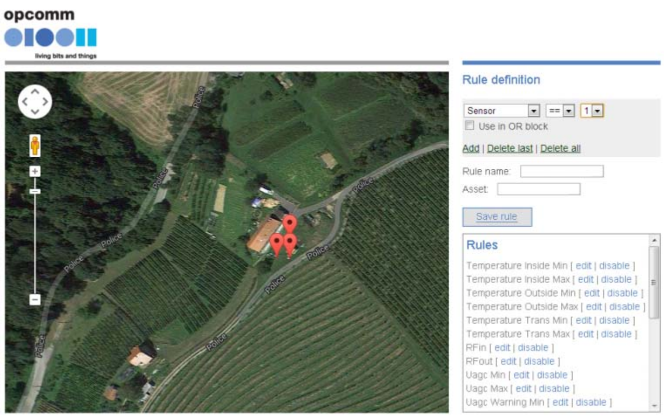

Figure 7: *Screenshot from expert user GUI.*

The expert user is able to visualize the geographical information of the sensor nodes, show measurements and their aggregates etc. The main functionality of the GUI is to assists the user to create rules through key/value pairs, which are then encoded in the rules.

Implemented rule language is based on JSON. An example of the rule is shown in Figure 8. Rule language consists of rule atoms based on a key/value pairs, where a value can be either equal, not equal, greater or smaller than the reference value.

{"Phenomena":"air_temperature", "Value":{"$gt":", 40.0"}}

Figure 8: *Example of an expert user rule, encoded in JSON.*

6 CONCLUSION

In this paper we propose the conceptual architecture for analytical module for stream processing in Asset management and present a concrete example of its implementation. The first results of testing, which are evaluated by end-user's feedback, showed promising results. The maintenance process is being more controlled and maintenance crew has all the needed information in their work orders, which appear on their mobile devices.

The proposed architecture is considered to be appropriate also for other business cases, since it enables data stream processing, complex event processing and also supports natural language, which is used in this case as one of the options for result interpretation. With the implementation we have demonstrated usability of the system for small/medium-sized companies.

Further work will include implementation of predictive capabilities, such as prediction of selected input streams, as well as simple and complex events.

Acknowledgements

This work was supported by the Slovenian Research Agency, by the Ministry of Education, Science and Sport within the Competence Center Open Communications Platform and the ICT Programme of the EC under PlanetData (ICT-NoE-257641), ENVISION (IST-2009-249120) and NRG4Cast (ICT-EeB- 600074).

References

[1] K. Kenda, C. Fortuna, A. Moraru, D. Mladenić, B. Fortuna, M. Grobelnik, "Mashups for The Web of Things," in Brigitte Endres-Niggemeyer (ed.) Semantic Mashups, Springer, 2013.

[2] Cugola G, Margara A, "Processing Flows of Information: From Data Stream to Complex Event Processing," ACM Computing Surveys, vol. 44, no. 3, 2012.

[3] Stonebroker et al., "The Eight Rules of Real-Time Stream Processing," ACM SIGMOID Record, vol. 34, pp. 42-47, 2006.

[4] M. M. Gaber, A. Zaslavsky, S. Krishnaswamy, "Mining Data Streams: A Review," SIG-MOD Record, vol. 34, no. 1826, 2005.

[5] J. Gama, Knowledge Discovery from Data Streams, CRC Press, 2010.

[6] A. H. D. Sakelšek, "ASM - Obravnavanje alarmov s platforme OCCAPI," Zavod tehnološka mreža ICT, Ljubljana, 2013.

[7] A. Moraru, K. Kenda, B. Fortuna, L. Bradeško, M. Škrajnc, D. Mladenic, C. Fortuna, "Supporting Rule Generation and Validation on Environmental Data in EnStreaM", ESWC 2012.

[8] marketsandmarkets.com, "Complex Event Processing (CEP) Market", Worldwide Market Forecast & Analysis (2013 - 2018), 2013.