Event Processing with Dynamically Changing Focus

Doctoral Consortium paper

Marc Schaaf
University of Applied Sciences and Arts Northwestern Switzerland
Olten, Switzerland
Email: marc.schaaf@fhnw.ch

***Abstract*—Event Processing technologies are likely to play an increasingly important role in future IT systems due to the increasing demand for on-line analytical systems as well as big data processing applications. Event Processing is particularly suitable for those applications due to its active processing approach combined with scalability. Today's approaches for achieving scalability are however focused on rather static event stream partitioning mechanisms to allow parallelization. Such approaches are well suited as long as a feasible partitioning for all processing tasks can be found. However when such a scaling mechanism is faced with processing tasks with dynamically changing focus areas, no effective pre-determined partitioning can be found which massively hampers with the required scalability. This paper presents the first steps towards a focused processing framework to overcome these limitations.**

I. Introduction and Problem Statement

Event processing systems are well suited for the rapid processing of measurement data to produce near real time results. To achieve scalability, a usual approach is the parallelization of the processing by subdividing the overall event stream into suitable partitions, that can be handled in parallel by several machines. This requires that the processing tasks are *focused* only on a subset of the event stream which needs to be known before the processing starts. However, when the focus of a processing task can only be determined during runtime or even changes during the ongoing processing, static partitioning approaches are not suitable anymore. In the research project DYNE („Dynamic Complex Event Processing for Hybrid Telecommunication Networks and Smart Grids"), we intend to overcome this limitation, following the overall research question, how event processing with dynamically changing focus can be realized for the processing of big data streams without hampering with the processing performance of the overall system. Our approach is to define a specialized processing model for focused event processing that separates the overall processing task into separate phases that can be executed in multiple stages to allow the handling of big amounts of event data. However this approach raises several detail challenges that are discussed in this paper.

The research work for the dissertation that is discussed in this paper is conducted as part of the DYNE project. The research methodology applied for this work is design research as the central intention is the creation of an approach for dynamically focused event processing to solve the given challenges. To allow a verification of the results, the created concepts will be tested against gathered real world test data for the considered use cases together with our project partner BaseN[1].

In the research project DYNE we aim at utilizing event processing concepts for the detection, analysis and tracking of dynamic complex situations which can be found for example in the two application areas, telecommunication network monitoring and Smart Grid monitoring alike as they require to rapidly detect and track situations of interest in country scale networks. We classify such dynamic complex situations by their two central requirements:

1) The need to correlate measurement data from various event sources to detect and analyze a situation.
2) The need to access a *non pre-defined* set of event sources which further *varies during the analysis* to allow tracking monitored situation.

For this paper we will discuss one of the project's use cases from the Smart Grid domain to introduce the challenges we are faced with and to outline our solution approach.

One major challenge, Smart Grids will be faced with is the replacement of big power plants by small, mostly renewable power providers. Households will increasingly install solar panels on their roofs; more and more wind turbine parks will be built. But these kinds of power suppliers are highly affected by changing weather conditions. To maintain the overall grid stability, a close monitoring of the power grid will be needed. An essential part of this monitoring will be the intelligent detection and tracking of situations like power production holes as they can be caused for example by clouds that affect closely related solar panels so that they not produce energy anymore. Solar panels at the border will however still produce energy, allowing the detection of the location of the clouds as well as their trajectory. Based on such information, dynamic load management systems can be created that compensate for such regional production holes by automatically switching of currently unused

[1] http://www.basen.net/

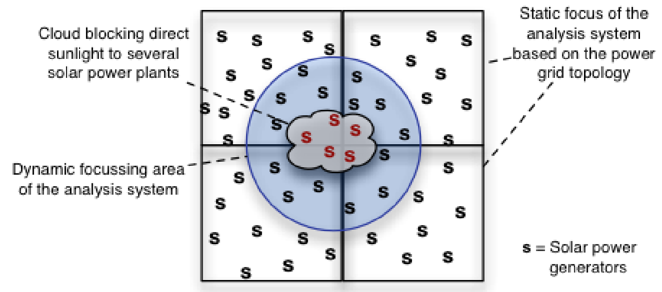

Fig. 1. Detection and tracking of an energy production hole caused by a cloud blocking direct sunlight to solar panels

household devices to reduce the local grid utilization and thus helping with the overall grid stability. The main goal of the DYNE project lies in the detection of complex situations like the above mentioned power production holes in near real-time. To achieve this goal we follow the approach of providing an event processing framework which is capable of handling dynamically changing foci to allow an in depth analysis of occurring incidents including the capability to correlate side effects with an analyzed incident. The remainder of this paper is structured as follows. The next section will discuss a concrete use case from the DYNE project and classify it with regard to its suitability for the focused processing approach. Section 3 will then briefly outline current event processing approaches related to the given problem set and discuss their limitations. Based on this, Section 4 will discuss our approach in more detail and point out the major challenges that we are facing. Section 5 concludes the paper with a description of the next steps.

II. Use Case and its Classification

One of the Smart Grid use cases investigated in the DYNE project is the detection of energy production holes as they are caused by clouds that block the direct sunlight to solar panels in a rural or urban areas. One or several clouds may affect closely related solar panels that do not produce energy while solar panels at the border of a cloud will still produce energy. The DYNE system needs to be able to monitor the solar panels to detect such a situation with all relevant aspects like its position, extend and trajectory in near real-time so the information can be used for intelligent grid management.

On the technical level, the use case constitutes strong challenges to a near real-time processing system responsible of monitoring a country wide network. The processing system must determine the presence of a power production hole from the huge stream of measurement data. To actually determine the position and size it further needs to combine several affected event streams dynamically (power generators in and around a power production hole) into a focus area. For this use case, the *focus area* needs to be formed based on the spacial neighborhood of the measured solar panels (Figure 1). However the actual extend of this focus area will only be known once a production hole was detected and the processing has started. Moreover, due to the movement of the power production holes, the focus area that is needed for the tracking must be adjusted continuously and dynamically. These two aspects however break with static event stream partitioning approaches as it is very likely that a focus area spans multiple partitions and is thus preventing an efficient parallelization .

To overcome the limitations imposed by static data partitioning approaches, we follow the approach of a dynamically focused event processing, that separates the in depth analysis of an identified problem, which we call *focused processing* from the indication that the problem probably exists which we call the *situation indication*. As part of this approach, we also introduced a transitional task, the *initial focus area determination*, that derives from the initial situation detection the required preconditions for the focused processing. As a first step, our approach is based on the following two assumptions:

1) A situation indication itself can be realized based on a small and pre-defined portion of the event stream and can be executed massively parallel as each indication task is separate from the other tasks.
2) Based on the initial situation indication it is possible to determine which parts of the overall event stream are required for an in depth analysis.

Based on our approach and the assumptions, we classified the use cases of the DYNE project to determine the general suitability of our approach for the given problem set. For the here outlined use case, the classification is as outlined in Table I. We also classified the other use cases covered by the project, which led to similar conclusions.

The classification of the use cases supports our assumption that the situation indication can be separated from the actual in depth analysis. Furthermore for all of the use cases we analyzed, the situation indication could be executed based on the stream of a single event source (e.g. the measurements from a single solar panel) which allows for a very flexible data partitioning for the initial situation indication.

III. Review of existing approaches

Several sophisticated Complex Event Processing Systems exist on the market like for example Esper[2] , StreamBase[3] or JBoss Drools Fusion[4] as well as in the scientific community like Rapide [3] or SASE [9]. Those engines already feature extensive query mechanisms including the capabilities for event stream processing. In the DYNE project we will evaluate those candidates to

[2]http://www.espertech.com/ (Accessed: 11.01.13)
[3]http://www.streambase.com/ (Accessed: 11.01.13)
[4]http://www.jboss.org/drools/ (Accessed: 11.01.13)

Situation Detection
The indicating situation, a certain drop in the energy production of a solar pane can be detected by monitoring the solar panels independent from each other. Thus the use case situation detection can take place on each single event stream separately without the need for correlation with other event streams which we consider as a *local problem*.
Initial Focus Area Determination
The initial focus area for the focused processing *can be identified based on the detected situation* by specifying a neighborhood relation based on the geospacial neighborhood information usable for querying for the solar panels in direct neighborhood to the triggering situation. Later adaptations to focus area are needed based on interim results.
Focused Processing
The focused processing itself can for this use case be subdivided into two phases: (1) Determining if a sudden drop in the energy production of a solar panel is possibly caused by a cloud by finding the border of the cloud (2) Determining the trajectory of the cloud based on positional changes within a certain time frame. All two phases represent a *regional problem* as they require the correlation of a set of event streams. Furthermore they all have a a *non pre-determined focus area*, thus they require a non pre-determined part of the overall event stream. In addition they also require the a *dynamic adaptations of this focus area* as for example the search for alternate traffic paths can't specify upfront complete set of event streams that will be required during the processing.

TABLE I. CLASSIFICATION OF THE SMART GRID USE CASE

identify their suitability for our problem set and their extendability for focused processing.

Current approaches for achieving scalability of event processing applications are mostly focused on efficient data partitioning and effective operator placement which is realized as a static optimization task that takes place before the actual instantiation of the processing system as for example discussed in [5]. Hirzel [1] describes an additional operator for the event processing language of System S that allows automatic partitioning of the event stream. However their work is again focused on determining the data partitioning before the actual processing has started. Therefore those approaches cannot effectively deal with continuous changes in the processing focus which require a flexible adaptation of the running system without hampering with the other parts of the processing.

Several approaches for optimizing the event processing itself by introducing new query languages or extensions to the concepts of existing ones have been published. Wu et al. [9] proposes SASE, a system that allows a more efficient processing of queries with large sliding windows among other optimizations. Wang et al. [8] extend this concept further by distributing the processing onto various machines based on a query plan mechanism from SASE. Maier [4] proposes an extension

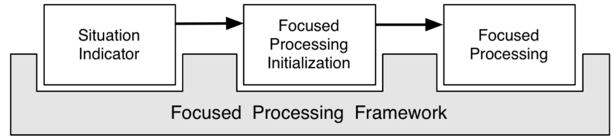

Fig. 2. Focused processing framework with the three main processing phases

to window based queries towards situation specific windows. Where the window size matches the occurrence of the situation not some fixed values like number of events or a time frame. Liu et al. [2] proposes a new query language NEEL that allows the specification of nested CEP patterns that allow the reuse of subsequences as partial results from various queries to optimize the processing. Even though such approaches do not address the required partitioning of the incoming event stream, they can be seen as a likely future addition to the DYNE processing system to optimize the processing responsiveness even further.

The approach for dynamically focused processing presented in this paper, requires the capability to dynamically provide event data to processing components to allow them to execute the various focused processing tasks. A very initial concept for a related dynamic event stream subscription system is presented in [6], [7]. Their motivation for such a dynamic event stream assignment lies in data dissemination for mobile computing applications and does not focus on the specification of a focused processing framework.

IV. APPROACH AND CHALLENGES

As currently no approaches are providing support for event processing with dynamically shifting foci, our aim is to specify a focused processing framework. It will define the structure for focused processing rules as well as their semantics and their execution process. This will allow the specification of focused processing rules based on the well defined semantics of our framework to guarantee for the correct execution. Our framework defines the focused processing in three phases, Situation Indication, Focused Processing Initialization and Focused Processing (Figure 2):

1) Situation Indication

The situation indication phase handles the initial detection of a situation of interest that requires special attention by a focused processing. Such a situation would for example be the energy production drop of a monitored solar panel as discussed in the use case. It is important to note that the result of this processing phase can in some cases only be an indication of a possible situation where the actual determination if the indicated situation exists is done separately in the focused processing phase. This uncertainty of the situation indication is caused by its limited view of the situation of interest. For example for the cloud tracking use case, the situation indication only indicates that a certain drop in the energy production has occurred for a single solar panel which might also be caused some failure in the solar panels.

One of the central aspects for the processing model and the corresponding language for the specification of this part of the rule, is the possibility to evaluate the specified triggering situation patterns in an efficient form against a huge amount of streaming data.

2) Focused Processing Initialization

Once a possible situation has been indicated, a focused processing needs to be started for the in depth analysis. This processing task is intended only to run on a very small subset of the overall event stream to be able to realize complex processing tasks in an acceptable time frame. The second processing phase is responsible to deduce the required part of the event stream, the *focus area*, and to prepare a separate processing environment where this data is available for the focused processing. To be able to deduce the focus area from the initial situation indication, an additional specification in a suitable language will be needed. This *Focus Area Definition Language* will express the initial processing focus area as a function of the gathered information of the situation indication step. For this, the language needs to be able to refer to the previous processing results and to correlate them with background knowledge both in spatial and temporal terms.

3) Focused Processing

Once the environment for the focused processing is set up, the actual, *focused situation processing* can start. This processing itself can be subdivided into three parts:

Focused Situation Processing

The actual focused situation processing happens in this third phase and can in contrast to the first processing phase be much more time consuming per processed event as the amount of events that need to be processed should already be reduced dramatically. This allows the use of much more expressive languages for this processing step which is needed in may of our use cases as for example for the correlation of the various energy production drops of solar panels that are shaded by clouds in a country wide power grid.

Focus Area Adaptation

During the setup of the focused processing, an initial focus area was defined to allow the focused situation processing to start. However over time, the processing focus will in many cases shift or be extended (e.g. when the tracked cloud moves or grows in size) which also requires the capability to specify the required adaptations as a function from the current processing state. Therefore a language needs to be found (in a similar form to the Focus Area Definition Language) that specifies how to deduce required adaptations of the processing focus from the current processing state.

Failure and Success Conditions and Actions

As the focused situation processing takes place on a continuous stream of events, conditions need to be defined when the processing should be stopped. This is required for both termination cases: The successful end of the processing as well as the cancellation of the processing, as the indicated situation was not found. Furthermore it needs to be possible to specify which actions shall be executed in either of the cases in addition to the termination of the focused processing. Therefore a language needs to be found or defined to specify these conditions as function from the current processing state.

The three phases specified by this processing framework will also be referred to when specifying a focused processing rule as outlined by the following pseudo code:

```
<Situation␣Indication␣Pattern>
TRIGGERS SPECIALIZED PROCESSING
<Focused␣Processing␣Definition>
BASED ON <Focus␣Area␣Description>
REQUIRES FOCUS CHANGE BY <Dynamic␣Focus
Adaptation␣Description>
EXECUTES <External␣Action>
OR IS CANCELED BY <Cancelation␣Condition>
```

We aim at providing a general framework for such focused event processing which allows the embedding of various specialized rule languages into the different parts of such a focused processing rule to allow a flexible tailoring to the needs of a given use case. For example for the described Smart Grid use case, the situation indication is a fairly simple analysis (drop in the energy production of a solar panel following a certain pattern) that however needs to be very fast as it needs to be done for a huge part of the overall event stream (in this case all solar panels), thus the use of language with limited expressiveness but fast execution characteristics would be required for this use case where another use case might need much more expressiveness already in the situation indication phase.

V. Challenges

Aside from the general challenge to provide a dynamic mechanism for the in depth analysis of complex situations in high volume event streams, which we aim to tackle with the presented processing approach, several detail challenges arise from the approach itself:

C1: Focused Processing Initialization; Determining the initial focus area from the situation Indicator:

The focused processing is intended to look at a suitable subsection of the overall event stream. As the processing requires the availability of all necessary data to work properly, it is essential to be able to define the initial focus area in such a way, that all required data is available while keeping that focus area as small as possible as adding too much to the focus area would hamper with the performance. Consequently, the focus area needs to be determined in a sensible way from an early situation indicator. It is important to note, that C1 covers the need to define an initial focus area, therefore the focus area that needs to be determined *before* the actual focused processing starts. Therefore the mechanism used needs to cope with the fact that the only information that is available so far is from the initial situation indication and static background knowledge on for example the topology of the monitored telecommunications network.

C2: Focus Area Adaption; Determining *when* the focus areas must be adapted and *how* they need to be adapted:

For the use cases we analyzed, the initially defined processing focus will not be enough for the entire focused processing as the focus shifts over time, possibly outside of the initial focus area. Therefore suitable mechanisms need to be found to

1) detect the need to adapt the focus area during the on-going focused processing and to
2) determine how the focus area needs to be adapted (which event streams are now relevant and which aren't anymore) and to
3) adapt the focus area for the on-going processing while guaranteeing for correct results.

Updating the focus area dynamically is one of the important research challenges for our approach as it is essential for the effectiveness of the focused processing. E.g., when a focused processing is based on the geographical position of a cloud, a change in its position would require also a shift in the focus area to include the event streams related to the new geographical location of the cloud as well as removing event streams from the old location which are not needed anymore. In addition to such a change, also the characteristics of focused situation can change over time.

C3: Failure and Success Conditions; Determining the end of the focused processing:

As the focused processing is done based on a continuous stream of events, it is necessary to specify some criteria, which can be used to identify if a focused processing is finished. This can be separated into two sub-challenges:

1) Determine if the focused processing reached its goal and thus can be terminated.
2) Determine if the focused processing is in vain and needs to be terminated as the initial indicator that triggered the processing misfired. Also the focused situation might disappear while the focused processing is still running like for example a cloud can disperse over time in such a way that it does not give a significant amount of shade to the solar panels anymore. One possibility to realize those checks would for example be the state of the indication. E.g. if the state of the trigger has changed, this could mean a premature end of the focused processing.

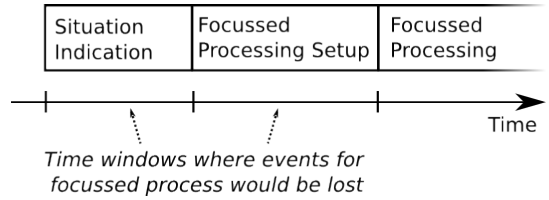

Fig. 3. Time windows from initial situation indication to the actual focused processing

C4: Handling time differences between the situation indication and the focused processing:

As the complete evaluation of a focused processing rule is done in multiple stages, the time difference between the start of a situation indication and the start of the actual focused processing results in a different view on the continuous event stream for the different processing phases. This time difference impacts the processing in two ways (Figure 3):

1) The initial situation indication takes some time before the actual processing can be triggered. During this period of time, the event stream might however already contain information that is required for the focused processing. Normally all events occurring during the situation indication phase will be lost for the focused processing which starts at a later point in time. Thus a suitable mechanism needs to be found to provide the the required event data. This however raises the question of the focus area of the possibly upcoming focused processing task otherwise all event data during the given period of time would need to be saved.
2) In addition to the delay caused by the situation indication, other delays will occur between the initial problem indication and the actual focused processing due to the focus area determination and the overall setup of the focused processing task.

The central questions that need to be answered are: (1) Is it acceptable for a focused processing task, that it misses some parts of the event stream and (2) if it is not acceptable to miss some events, how can it be determined in a sensible way what needs to stored even before the focused processing has been started?

C5: Detection of multiple focused processing tasks for the same situation:

The initialization of a focused processing will be based on an indicator. However it is by far not guaranteed that such indicators don't fire multiple times for various incarnations of the same situation. For example if one cloud blocks direct sunlight to some solar panels, the indication for a possible cloud will be raised by all of the shaded solar panels. This however results in the challenge that ongoing focused processing tasks need to be combined in such cases to grasp the overall situation as well as to free resources. Furthermore an already analyzed situation might still trigger further situation indicators which also need to be grouped to the already running or possibly even finished focused processing. In the case of a false indication of a situation of interest, future indications “that would lead to the same conclusion” should also be suppressed if possible. Thus a sensible way to give a characterization of indicated situations that allows to correlate multiple indications of the same situation needs to be found.

VI. Conclusion and Next Steps

Current trends towards rapid online analytics of big data amounts support the further development of event processing approaches as a fast and scalable technology. However current event processing concepts are not yet capable of providing efficient methods for processing tasks with a dynamically changing focus as they are unsuitable for the typical static optimization methods applied to event processing systems.

To overcome this limitation, we propose the concept of focused processing rules that allow the separation of the dynamic processing, that can’t be effectively handled by static optimization methods, in a task that is triggered by a simple triggering condition which we call the a situation indicator. The central characteristic for these situation indicators is their limitation to a well defined part of the overall event stream, which allows the usage of common optimization approaches.

For the current state we gathered several use cases from the application domains, telecommunications networks and Smart Grids, focused in our research project. Based on these use cases we created a detailed requirements specification and identified the central challenges.

Based on the specified requirements and challenges we will now define the model for a focused event processing. This model definition will be oriented on the use cases but with the aim of providing a general processing model that can be used in a much broader scope. Based on the definitions of the processing model with regard to the processing flow and the detailed semantics of it we will define a rule specification language for dynamically focused processing rules.

Our aim it to design the language with focus on modularity of the different rule parts that need to be expressed. With this we hope to provide the capability to tailor the actual rule definition language to the needs of a certain use case by for example choosing a certain sub-language for specifying situation indicators with geospacial attributes. For the sub languages we expect to be able to largely reuse existing event processing languages.

To evaluate the applicability of our approach we will create a prototype that realizes the focused processing model as an addition to a cloud computing based monitoring platform provided by our industry partner BaseN. Based on this prototype we will realize several use cases which require the detection of complex situations For the telecommunications use cases the evaluations will be based on real measurement data gathered from a country wide telecommunications network. For the Smart Grid use cases the tests will be based on test data that will be partly simulated due to limited availability of detailed large scale measurement data.

Acknowledgment

Parts of the here presented work was done as part of the Eurostars Project E!7377.

References

[1] Martin Hirzel. Partition and compose: parallel complex event processing. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS ’12, pages 191–200, New York, NY, USA, 2012. ACM.

[2] Mo Liu, E. Rundensteiner, D. Dougherty, C. Gupta, Song Wang, I. Ari, and A. Mehta. High-performance nested CEP query processing over event streams. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 123–134, april 2011.

[3] DavidC. Luckham. Rapide: A language and toolset for causal event modelling of distributed system architectures. In Yoshifumi Masunaga, Takuya Katayama, and Michiharu Tsukamoto, editors, *Worldwide Computing and Its Applications — WWCA’98*, volume 1368 of *Lecture Notes in Computer Science*, pages 88–96. Springer Berlin Heidelberg, 1998.

[4] David Maier, Michael Grossniklaus, Sharmadha Moorthy, and Kristin Tufte. Capturing episodes: may the frame be with you. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS ’12, pages 1–11, New York, NY, USA, 2012. ACM.

[5] Scott Schneider, Martin Hirzel, Bugra Gedik, and Kun-Lung Wu. Auto-parallelizing stateful distributed streaming applications. In *Proceedings of the 21st international conference on Parallel architectures and compilation techniques*, PACT ’12, pages 53–64, New York, NY, USA, 2012. ACM.

[6] Yiannis Verginadis, Nikos Papageorgiou, Ioannis Patiniotakis, Dimitris Apostolou, and Gregoris Mentzas. A goal driven dynamic event subscription approach. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS ’12, pages 81–84, New York, NY, USA, 2012. ACM.

[7] Yiannis Verginadis, Ioannis Patiniotakis, Nikos Papageorgiou, and Roland Stuehmer. Service Adaptation Recommender in the Event Marketplace: Conceptual View. In Raúl García-Castro, Dieter Fensel, and Grigoris Antoniou, editors, *The Semantic Web: ESWC 2011 Workshops*, volume 7117 of *Lecture Notes in Computer Science*, pages 194–201. Springer Berlin Heidelberg, 2012.

[8] Yongheng Wang and Shenghong Yang. High-performance complex event processing for large-scale RFID applications. In *Signal Processing Systems (ICSPS), 2010 2nd International Conference on*, volume 1, pages V1–127–V1–131, july 2010.

[9] Eugene Wu, Yanlei Diao, and Shariq Rizvi. High-performance complex event processing over streams. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, SIGMOD ’06, pages 407–418, New York, NY, USA, 2006. ACM.