Supporting Heterogeneous Fault-Tolerance Requirements in Complex Event Processing

Ruben Mayer, Ahmad Slo
Kurt Rothermel
Institute of Parallel and Distributed Systems
University of Stuttgart, Germany
first.last@ipvs.uni-stuttgart.de

Markus Endler
Dept. of Informatics, PUC-Rio
Rio de Janeiro, Brazil
endler@inf.puc-rio.br

Francisco J. da Silva e Silva
Dept. of Informatics, Federal University of Maranhao
Sao Luis, Brazil
fssilva@deinf.ufma.br

ABSTRACT

For Industrial Internet-of-Things (IIoT) applications, Complex Event Processing (CEP) is a well accepted approach for processing and transforming the raw events received from sensors into application-relevant complex events. Although fault-tolerance in CEP has been largely studied, most of work found in literature assumes a homogeneous system architecture, homogeneous sensor types, and do not consider that the IIoT application may tolerate some failures in the production of the application-relevant complex events. In this position paper, we present the event fault-tolerance requirements of a specific IIoT application in hospital management, show the underlying general system architecture and failure modes, and how the application fault-tolerance requirements influence the decisions in the deployment of the CEP operator graph.

KEYWORDS

Complex Event Processing, Fault Tolerance, IIoT

ACM Reference format:
Ruben Mayer, Ahmad Slo, Kurt Rothermel, Markus Endler, and Francisco J. da Silva e Silva. 2017. Supporting Heterogeneous Fault-Tolerance Requirements in Complex Event Processing. In *Proceedings of Technical Report 2017/03, University of Stuttgart, Nov 2017,* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Stream Processing, and in particular, Complex Event Processing has been recognized as a suitable middleware paradigm for IIoT that is capable of transforming low-level event streams received from sensors into actionable information about state changes of the monitored environment. Although several works have tackled fault-tolerant CEP, most of them have made, on the one hand, too strong assumptions about the failure modes of sensor and communication environment and their homogeneity, and on the other hand, neglected that most IIoT applications do not pose the same strong requirements on all types of complex events. Thus, we believe that it is time for the CEP research community to start investigating flexible CEP approaches and architectures that consider the hybrid nature of IIoT systems architecture, including different types of sensors, wireless connectivity, mobility, and situation-specific variable consistency requirements (reliability, timing, ordering) on the delivery of the events probed from the physical environment being monitored.

Technical Report 2017/03, Nov 2017, University of Stuttgart
2017. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 MOTIVATION SCENARIO

In order to illustrate that some IIoT applications may have different consistency requirements for each complex event, we introduce Hospital 4.0, which is an IoT service for improving management of a hospital Emergency Department (HED) in two ways: supporting better resource allocations on the long term, and optimizing online coordination of medical staff on duty whenever a high risk (e.g., stroke) patient has to be treated. Hospital 4.0 is being developed at PUC-Rio in cooperation with the ProCardiaco Hospital of Rio de Janeiro. Essentially, it tracks the whereabouts of all patients and healthcare personal at the level of hospital rooms (and sections), and captures the moments of times at which a person enters and leaves a room, and consequently detects when, and for how long, different actors are co-located in a room, supposedly one of them doing some healthcare procedure on the other one, the patient.

The two scenarios that we handle in Hospital 4.0 are the following: (i) We assume that a patient waiting room (PWR) is the antechamber for 3 doctor offices that do the first examination of newly arriving patient. The Hospital 4.0 manager needs to know when the PWR is full, while at the same time, any of the three doctor offices is empty (for a longer time). This would indicate that the fullness of PWR is due to some doctor's absence. In this case, Hospital 4.0 is expected to deliver three types of complex events *PWR-full/PWR-norm* and *Doc-Missing*. For these events, it becomes clear that false negatives are not acceptable for either events, while false positives are not critical for PWR-full, while they are not acceptable for *PWR-norm* and *Doc-Missing*. Neither is the order of delivery of events *PWR-full* and *Doc-Missing* important, assuming that the delay between the deliveries is reasonable. Finally, the restrictions on delay are very relaxed, since the application/manager is allowed to take some action with some delay upon slowly changing states of the environment. (ii) When a critical patient with a suspected stroke arrives, a strict procedure (sequence) of actions and exams has to be performed with the patient within very concrete time intervals. For example, after patient arrival (event *A*) until the end of patient admission (*B*), the maximum time span can be 10 minutes. So the Hospital 4.0 will deliver event *missed-AB* if this constraint is not fulfilled. Then, one must start monitoring his/her vital functions (*C*), start basic exam support (*D*), start and finalize the Cranial Tomography (*FCT*). Again, between event *A* and *FCT* not more than 25 minutes should pass. So, Hospital 4.0 may generate event *missed-AF* if this deadline is missed. In this scenario, all events must be delivered (false negatives are not acceptable), but for aggregate events *missed-AB* and *missed-AF*, false positives may be acceptable. According to the criticality of the action associated to each event, some of them may be delivered with some delay, such as events *C* and *D*, since they do not contribute to time checks. In regard to out-of-order delivery, it is acceptable, for example, that *B* and *missed-AB* are out of order.

3 SYSTEM ARCHITECTURE

The system architecture we assume for implementing the described motivation scenario is presented in Figure 1. At the bottom there is the physical environment, whose state changes on time depending on the occurrence of specific events. For instance, the environment changes from state 1 (s_1) to state 2 (s_2) in the occurrence of event 2 (e_2) at a given time (t_1). The events that occur in the physical level are perceived by sensors connected to a mobile hub [2] device through a short range wireless network, such as Bluetooth Smart. Some sensors periodically perceive a physical environment property, such as the temperature or humidity level, sending the probed value periodically to the mobile hub (e.g. each second), while others only produce a data event whenever there is a change on the perceived physical environment property, such as the turn on/off of a device. We call these latter ones aperiodic events. The mobile hub then forwards the sensor data to a CEP system deployed in a cloud infrastructure. This communication is usually done through a wireless link using a WLAN, LPWAN or WMAN (e.g. the cellular network). In spite of the attribute "Mobile", in case of Hospital 4.0 these hubs are kept stationary and are attached to the hospital walls or ceiling. The effective processing of the sensors data is performed through an Event Processing Network (EPN) composed by a set of CEP operators forming a graph structure. The derived complex events are then consumed by the application. As can be seen in the figure, the application usually only requires to learn about those state changes of the environment that are relevant for its action goals. In such system architecture, there can be several types of failures, such as communication link failures, node failures, and software component failures, and each of them has to be dealt in a different way.

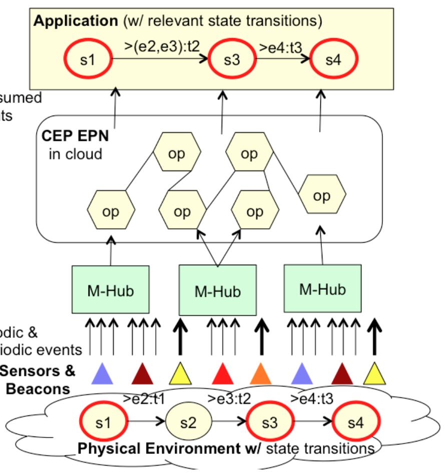

Figure 1: System Architecture.

4 FLEXIBLE FAULT-TOLERANCE IN CEP

In our previous work [1] ,we describe an approach for fault tolerance in CEP systems based on a rollback-recovery mechanism that provides strong consistency of the resulting event stream in case of multiple simultaneous CEP operator failures, such that neither false negatives nor false positives occur. The approach uses the mechanisms for providing strong consistency in all CEP operators, which implies that all events consumed by the application will be delivered with strong consistency. As we have shown in the motivation scenario (Section 2), this is not necessarily needed. If the application does not require strong consistency for all complex events, the recovery process in case of an operator failure can be implemented more efficiently, e.g., decreasing the delay required for failure recovery and the memory footprint at operator nodes.

However, to provide such a heterogeneous fault-tolerant approach, it will not be sufficient to simply propagate the efficient mechanism for providing a relaxed fault-tolerance consistency model to all operators of the operator graph that contribute to the detection of a complex event that has weak consistency requirements. Some of those operators may also be used in the processing steps of other event types with high consistency requirements. In other words, if we isolate the operators responsible for the generation of each complex event type that an application consumes, there may be overlapped regions in the operator graph.

Another issue that must be addressed is how to expose a simple interface for the application developer through which he/she can specify the different consistency requirements of the complex events. Finally, a tool that receives as input the required consistency models specified by the programmer and automatically performs all deployment and configuration of the fault tolerance mechanisms through the entire operator graph must be provided.

5 CONCLUSION

Most of the recent works on fault-tolerant CEP have made strong assumptions about the failure modes of sensor and communication environment and their homogeneity. This paper argues for the necessity of providing more flexible approaches, considering the hybrid nature of IIoT systems architecture and applications. To demonstrate that, we presented Hospital 4.0, an IoT service for improving management of a hospital Emergency Department (HED), and its system architecture, describing its heterogeneous requirements regarding the event types consumed by this application.

REFERENCES

[1] Boris Koldehofe, Ruben Mayer, Umakishore Ramachandran, Kurt Rothermel, and Marco Völz. 2013. Rollback-recovery Without Checkpoints in Distributed Event Processing Systems. In *Proceedings of the 7th ACM International Conference on Distributed Event-based Systems (DEBS '13)*. ACM, New York, NY, USA, 27–38.

[2] L. E. Talavera, M. Endler, I. Vasconcelos, R. Vasconcelos, M. Cunha, and F. J. d. S. e. Silva. 2015. The Mobile Hub concept: Enabling applications for the Internet of Mobile Things. In *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. 123–128.