Towards adaptive quality-aware Complex Event Processing in the Internet of Things

Majid Lotfian Delouee
Bernoulli Institute
University of Groningen
Groningen, The Netherlands
m.lotfian.delouee@rug.nl

Boris Koldehofe
Bernoulli Institute
University of Groningen
Groningen, The Netherlands
b.koldehofe@rug.nl

Viktoriya Degeler
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
v.o.degeler@uva.nl

***Abstract*—This paper investigates how to complement Complex Event Processing (CEP) with dynamic quality monitoring mechanisms and support the dynamic integration of suitable sensory data sources. In the proposed approach, queries to detect complex events are annotated with consumer-definable quality policies that are evaluated and used to autonomously assign (or even configure) suitable data sources of the sensing infrastructure. We present and study different forms of expressing quality policies and explore how they affect the process of quality monitoring including different modes of assessing and applying quality-related adaptations. A performance study in an IoT scenario shows that the proposed mechanisms in supporting quality policy monitoring and adaptively selecting suitable data sources succeed in enhancing the acquired quality of results while fulfilling consumers' quality requirements. We show that the quality-based selection of sensor sources also extends the network's lifetime by optimizing the data sources' energy consumption.**

***Index Terms*—Complex Event Processing, Adaptation, Quality, Internet of Things**

I. INTRODUCTION

In applications of the Internet of Things (IoT), real-time responses to situations are a key requirement, e.g., to detect, with low latency, a traffic jam in a traffic monitoring application. Distributed Complex Event Processing (DCEP) allows the efficient detection of situations of interest in the form of *complex events*. The detection of complex events and the resulting quality depends especially for IoT applications on primary event sources, often based on sensory data. In particular, the quality expressed in form of Quality of Service (QoS) and Quality of Results (QoR) depends on the dynamics of the environment, e.g., the availability of data sources.

In DCEP, an established way of reacting to dynamics is to adapt the configuration of the detection logic and their placement to resources of a distributed environment (cf. [2], [11]–[13], [18]). This allows influencing QoS metrics of a DCEP system, e.g., end-to-end latency, and bandwidth consumption. In combination with other runtime mechanisms like load shedding (e.g., [16], [21]), DCEP can already benefit from trading QoS against QoR. However, state-of-the-art approaches operate by design decoupled from the configuration of data sources and their capabilities. This imposes limitations on reacting to changes in the sensory system, e.g., the energy level of sensors, or the quality of sensed data. Changes in QoR are propagated, but not actively influenced by the DCEP system.

Contrary, in the context of the *sensor selection problem* [5] and *data source switching mechanisms* (e.g., [8], [14]) related approaches study the dynamic selection of sensors in IoT environment to optimize QoR of sensed data. Such optimizations are performed with respect to specific data attributes or a set of specific fused sensor sources. However, integrating such methods in the context of DCEP requires linking them dynamically to different configurations of heterogeneous sensory sources. Only in this way, the flexibility of current DCEP systems in reconfiguring and rewriting the detection logic of complex events can be used to optimize for QoR.

In this paper, we contribute to a better understanding of how to link sensor configurations with the capabilities of adapting DCEP environments to enhance QoR and QoS. We enhance DCEP with the concept of so-called quality policies and corresponding quality monitoring mechanisms. This enables DCEP to observe necessary changes at the sensor deployment *and* the complex event detection logic. As a result, DCEP can define appropriate sensing configuration restrictions (e.g., cost constraints) subject to the quality requirements expressed by consumers. Subsequently, such restrictions can be utilized as a utility metric in an efficient data source assignment.

In summary, this paper offers the following contributions:

1) A new model to represent quality requirements in DCEP systems in order to improve resource utilization in IoT.
2) A CEP quality monitoring mechanism that triggers adaptation decisions to meet consumers' quality requirements.
3) A performance evaluation of the proposed approach by applying quality policies over a synthetic dataset.

The paper is structured as follows. We introduce the problem of quality-driven CEP in Section II. A system design is detailed in Section III. We present evaluation results of quality-driven CEP in Section IV. Section V concludes the paper and discusses future work.

II. Problem Statement

We consider a DCEP system to consist of *producers*, *consumers* and *brokers*. The producers (e.g., mobile phones) generate simple events from sensory data. The sensory data sources (e.g., Bluetooth) that are used at a given time t form the set of active sensing deployments $\mathcal{SD} = \{sd_1, ..., sd_j\}$ where sd_i refers to a specific sensor source.

Correspondingly, consumers (e.g., applications) express their interest in complex events in the form of queries where the set $\mathcal{Q} = \{q_1, \ldots, q_n\}$ denotes the set of currently deployed queries. Each query q_i describes the logic of how to detect complex events in the form of standard CEP operators like pattern matching, windows, or aggregates over event streams and specific event attributes. The imposed detection logic is executed on the brokers. As part of the query, consumers also annotate in queries for event attributes their quality requirements (e.g., the location accuracy less than one meter). We denote the set of *consumer-side constraints* of all deployed queries in $\mathcal{Q}$ by $\mathcal{G} = \{g_1, g_2, ..., g_k\}$.

In this work, we aim to support *quality-driven CEP*, in which, a DCEP system selects suitable data sources, i.e., a sensing deployment $\alpha(\mathcal{SD}) \subset \mathcal{SD}$, where α determines which sensor sources of $\mathcal{SD}$ are used. A quality-driven CEP is required to meet the consumer constraints in quality or notify consumers when no proper sensing deployment is feasible. Furthermore, each sensor source sd_j of a sensing deployment imposes a *system-side cost (SSC)* denoted by $C_{SSC}(sd_j)$ as well as a cost for performing *quality monitoring (QM)* for every query q_k denoted by $C_{QM}(sd_j, q_k)$.

More, formally a quality-driven CEP aims to find α which minimizes the cost factors imposed by system-side costs and quality monitoring costs subject to the quality constraints of a consumer, i.e.,

$$\begin{aligned}
\min \quad & w_s \sum_{sd_j \in \mathcal{SD}} \alpha(sd_j) C_{SSC}(sd_j) \\
+ \quad & w_q \sum_{sd_j \in \mathcal{SD}} \alpha(sd_j) \sum_{q_k \in \mathcal{Q}} C_{QM}(sd_j, q_k) \\
s.t. \quad & \alpha(\mathcal{SD}) \text{ satisfies Consumer-Side Constraints in } \mathcal{G} \\
& \alpha(sd_j) = 1 \text{ iff } sd_j \text{ is selected.} \\
& \alpha(sd_j) \in \{0, 1\}
\end{aligned}$$

where w_s indicates the weight related to system-side costs, and w_q is the weight associated with monitoring costs.

III. System Design

In the system's design, we build on a logically centralized component, called the *controller* to enforce a quality-driven CEP. As illustrated in Figure 1, the controller supports the matching of subscriptions to advertisements and places the detection logic on functional CEP engines hosted by the brokers. In addition, the controller's functionality is enriched by employing quality agents which generate so-called *control events* to enunciate a quality-related situation. Furthermore, the controller reuses concepts for the flexible execution of event processing operators, as proposed in TCEP [12] and CEPLESS [10], enabling to emendate the operator placement utilizing processing engines (e.g., Apache Flink).

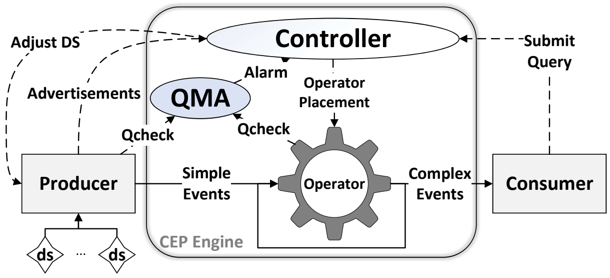

Fig. 1. Proposed System Design. The solid lines indicate the Data/Event Flow, while dashed indicate Query Control Flow.

A. Quality Requirement Description

This research extends the typical query definition by introducing *quality policies*, which specify those quality metrics substantial for the consumers, e.g., the latency of delivered values. In addition, *threshold levels* can be defined to quantify queries' quality requirements. We categorized quality policies as *static* and *dynamic*. In the static class, a constant threshold can be clearly determined in the query definition, only for one type of quality metric. Hence, a distinct static quality policy is required for each aspect of an event, e.g., a policy might define a threshold for the acceptable resolution of images in PPI. In the second class of quality policies, more flexibility is provided since the *dynamic thresholds* can be defined based on a second parameter (e.g., time). This enables a consumer to generate more intricate quality definitions causing a higher level of requirement satisfaction.

The controller validates the data sources' characteristics concerning the requested thresholds and regulates them, if necessary. When non of the available data sources is either qualified or can be adjusted based on the threshold, the controller rewrites the query or quality requirements and notifies the query issuer. Then, the consumer will decide whether to deploy the new query model or withdraw the query. Besides, the quality policy should be revised over time since consumer preferences may vary, e.g., reducing the threshold of the result's interval in case of an emergency in healthcare systems.

B. Quality Monitoring

In the monitoring procedure, the surveillance model (e.g., with the lowest latency) and accessible computation resources should be pondered simultaneously since both influence adaptation decisions. Hence, one of the novelties in this work is to introduce and employ a *Quality Management Agent (QMA)* that is in charge of inspecting the event streams according to the requested quality requirements. This component can be hosted similarly to operators (e.g., at brokers) checking the quality of produced events (i.e., Qcheck). At the same time, it can be connected to producers to update the data sources' status.

To estimate the costs imposed by hiring QMAs, overhead can be seen concerning *time* includes the cost of switching between the current data source (i.e., sd_j) and the next option (i.e., sd_l) as ($C_s(sd_j, sd_l, q_k)$) and the possible delay caused by quality analysis ($C_a(sd_j, q_k)$), as well as *computation* as required resources for sliding window processing ($C_w(sd_j, q_k)$) and the resource for assigning data sources ($C_{ads}(q_k)$), totally as:

$$\begin{aligned} C_{QM}(sd_j, q_k) = \quad & C_s(sd_j, sd_l, q_k) + C_a(sd_j, q_k) \\ & + C_w(sd_j, q_k) + C_{ads}(q_k) \end{aligned}$$

C. Sensing Deployment Adaptation

To make adaptation decisions, a step-by-step instruction set is demanded to receive sufficient information and allocate suitable data sources. To this end, we adapt the whole allocation procedure based on MAPE-K feedback loop [4]. In this vein, a data source is tagged as qualified if its provided service meets all specified quality policies in the subscription. Then, the system-side costs are calculated for each qualified option, e.g., if the latency is the significant factor for the system, the required time to deliver the events to the brokers from each qualified data source will be considered in the utility function. Moreover, the results from $C_{QM}(sd_j, q_k)$ are added as quality monitoring costs. Finally, once all costs are determined, a globally optimized solution will highlight the data sources among all qualified choices to feed the CEP system.

In the meantime, QMAs investigate the status of events and data sources to produce alarms, if necessary. Upon receiving any, the controller checks the alarm's type and reacts proportionally to maintain the quality of results. In more detail, if the alarm implies that a data source is not reachable anymore, the controller updates the list of advertisements and reassigns the data sources, if required. Moreover, since the data source's quality may degrade over time (e.g., reading's accuracy), the controller will perform the data source reassignment if the sensor cannot heal timely. For the sake of shortness, we skip other types of alarms (e.g., a modification in quality policies) that demand similar responses.

IV. Evaluation

In this section, we compare the execution of quality-driven CEP with two baseline strategies; *Optimal Dynamic Accuracy (ODA)* and *Optimal Dynamic Energy (ODE)*. The former dynamically picks the most satisfactory data source in terms of accuracy and adjusts the sensing deployment regarding the person's current location. The latter assesses the energy consumption of the accessible data sources as the distinguishing factor to adaptively opt for the best option on every occasion.

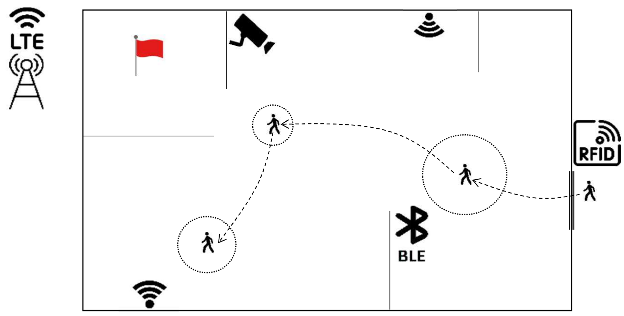

Fig. 2. Use-Case for dynamic quality policy monitoring

In our simulation, we exploit *Apache Kafka* to operate as an event broker that serves data and control events. Besides, we use *FlinkCEP*, a library implemented on top of *Apache Flink* to detect complex events. To exhibit the potential abilities, we evaluate our quality-driven CEP system in two different scenarios with static and dynamic quality policies. But, for the sake of shortness, we elaborate on the second type, which illustrates better the benefits of applying quality policies in query processing.

Use-Case Scenario: The chosen IoT scenario for dynamic quality policy is a security monitoring application for target location tracking. Given a continuous query, the goal is about locating people who are approaching a red-flagged spot in a pre-determined area as depicted in Figure 2. Here, the system aims to warn people who are in proximity to a red-flagged spot with a distance smaller than a predefined value. Moreover, the dynamic quality policy for this query is specified as "The closer a target is to the red-flagged spot, the more accurate the location event is". In this policy, the requested quality metric is *accuracy* and the second parameter based on which the accuracy is varied is *location*. In the query submission, a consumer should accurately specify the details for the red-flagged spot, and thresholds for the quality policy.

All moving targets in this scenario previously introduced their accessible sensing infrastructure (e.g., smartphone) with their sensing capabilities (e.g., location tracking by WiFi module). In addition to these data sources carried by targets, stationary data sources are embedded in the environment (e.g., a surveillance camera). We assume that all targets will give continuous access to their registered data sources and not deliberately block the connection. In order to estimate the energy consumption in this scenario, we reuse the energy consumption (EC) measurements (collected from [3], [9], [15], [17], [19]), exhibited in Table I. It should be noted that among all data sources shown in this table, only *Camera* is tagged as stationary, while others consume the battery power of the target's smartphone. To normalize the energy consumption

TABLE I
SENSING CONFIGURATIONS AND THEIR CHARACTERISTICS

Name	Range (m)	EC (mW)	Accuracy(m)
BLE	70 - 100	426	1 - 3
RFID	1 - 12	375	0.1 - 2
WIFI	50 - 100	817	1 - 5
Camera	N/A	374	< 1
LTE	$>$ Km	1634	< 1

TABLE II
APPLIED QUERIES IN THE DYNAMIC QUALITY POLICY SCENARIO

Q#	Definition		Quality Policies	
	RFS	PR(m)	Q-metric	condition
Q1	60 : 185	70	Accu $<$ 2m	0 $<$ DTS $<$ 100
			Accu $<$ 5m	100 $<$ DTS $<$ 200
			Accu $<$ 10m	200 $<$ DTS $<$ 1000
Q2	310 : 80	80	Accu $<$ 3m	0 $<$ DTS $<$ 50
			Accu $<$ 7m	50 $<$ DTS $<$ 1000
Q3	190 : 240	50	Accu $<$ 2m	0 $<$ DTS $<$ 100
			Accu $<$ 5m	100 $<$ DTS $<$ 1000
Q4	370 : 200	60	Accu $<$ 4m	0 $<$ DTS $<$ 50
			Accu $<$ 10m	50 $<$ DTS $<$ 1000
Q5	210 : 140	35	Accu $<$ 2m	0 $<$ DTS $<$ 20
			Accu $<$ 3m	20 $<$ DTS $<$ 50
			Accu $<$ 5m	50 $<$ DTS $<$ 1000

of the Camera, we assume that its consumption is equal to the situation in which the smartphone is placed in *airplane* mode.

To investigate various query models with diverse locations for the red-flagged spot and the requested quality policies, we try the queries listed in Table II. Each of these queries is applied separately to our scenario. To estimate the availability of data sources based on their coverage, a route should be defined for the target. Therefore, a synthetic location dataset for a target route is utilized in our simulation that covers all zones in the predetermined area. In each query, the position of the red-flagged spot is denoted by RFS, and the prohibition radius is presented as PR. Consequently, if a moving target is located at a distance less than PR to the RFS point, an alarm should be generated. Moreover, the dynamic quality policy is depicted based on a pair of *Q-metric* and the corresponding *condition*. In the former, the required level of accuracy (Accu) is determined, while in the latter, the upper and lower bound values for the Distance To Spot (DTS) are specified.

The simulation output of applying dynamic quality policy over query processing is depicted in Figures 3 and 4. First, we discuss the performance in terms of consumed energy. One can observe from Figure 3 that the ODA's bars exceedingly getting taller with the increase in the number of queries. That means if an approach takes the highest accuracy all the time, it will shortly deplete the smartphone's battery completely. This confirms the motivation of our work to dynamically select the best sensing infrastructure. The reason for such dramatic consumption is the high amount of energy required to establish and maintain an LTE connection. On the other

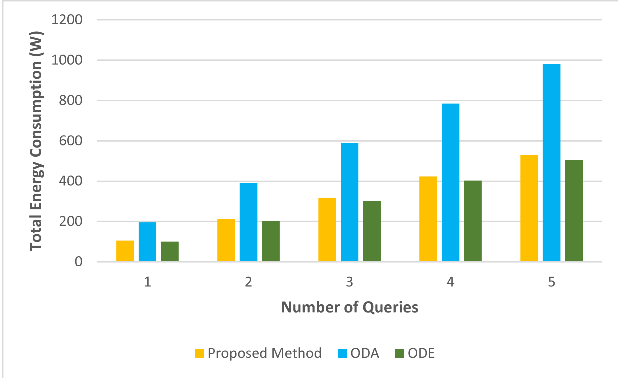

Fig. 3. Total energy consumed by active data sources for a different set of queries

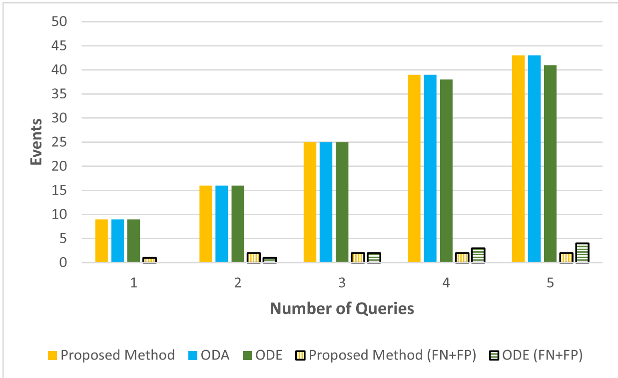

Fig. 4. Number of detected events and the summation of FP and FN for a different set of queries

hand, the consumption of quality-driven CEP and ODE is pretty similar, with a small excess in our results. Keeping this fact in mind that ODE is assumed as the optimal approach in terms of energy consumption, the obtained results ascertain that our quality-driven CEP performed slightly behind ODE, which authorizes us to claim that our solution is near-optimal in terms of energy consumption.

Two favorable comparison metrics in event processing systems are the number of False Negatives (FN) and False Positives (FP). In our example, an FN denotes a violation by entering a red-flagged area that occurred in the real world, but the event processing system could not catch it and trigger an alarm. Besides, an FP indicates a wrong violation has been detected in the system while in the reality it did not occur. Since both of these errors are feasible in our use case with small counts, we form a single number of their summation that makes the differences more distinguishable as illustrated in Figure 4. The ODA employs the best sensing deployment in terms of accuracy. That's why its results display the exact number of events that should be detected, without any number of FN or FP.

To achieve such accuracy in event detection, the system will swallow the battery capacity in no time. Although the number of detected events in others (i.e., ODE and quality-driven CEP) is almost the same as ODA, the growth of FNs + FPs makes the difference, especially by involving more queries. In other words, the enlargement in the number of FNs and FPs is more evident in the bars related to ODE, while this summation in our results remains the same after adding more queries. The imaginable reason for the better performance of quality-driven CEP is the lower detection capacity of data sources with the smallest energy consumption when the red-flagged spot is located in their closed vicinity.

Similar to FN and FP in stream processing, *precision* and *recall* are mostly employed to distinguish between classifiers in the Machine Learning field of study. *F-Score* is a comparison measure that combines these two metrics [20]. Therefore, it can be used in stream processing to compare mechanisms in terms of accuracy. Analyzing the F-Score shows an *ascending trend* in the reports which are approximately 0.947%, 0.941%, 0.961%, 0.975%, and 0.977% for set of 1,2,3,4 and 5 queries, respectively. While, the outcome for ODE as 1%, 0.969%, 0.961%, 0.962%, and 0.953% displays a *descending trend*. This proves the ability of quality-driven CEP to deal with involving more queries while maintaining the accuracy of event detection.

V. Conclusion and Outlook

In this research, we studied how to adaptively configure sensing deployments based on the monitored quality characteristics of detected events *and* their data sources. The reported results demonstrated that in quality-driven CEP adapting the sensing deployment performed near-optimal in both energy consumption and quality measured in form of FNs/FPs. In addition, the F-Score results proved the capabilities of quality-driven CEP to better deal with the quality requirements of consumers when the number of deployed queries increases.

In future work, we plan to study how to better support concurrent queries by introducing *priorities* as part of quality policy definitions, a common requirement in current event-driven applications. In addition, estimating and accounting for the switching overhead in selecting data sources is a central point that requires further understanding. Finally, we plan to extend our study by accounting also for more dynamic behavior by also considering the quality degradation of data sources over time and studying appropriate models for dynamic quality policies.

References

[1] Arafat, A.I., Akter, T., Ahammed, M.F., Ali, M.Y. and Nahid, A.A. A dataset for the internet of things based fish farm monitoring and notification system. Data in Brief, 33, p.106457, 2020.

[2] Cardellini, V., Lo Presti, F., Nardelli, M. and Russo, G.R. Runtime Adaptation of Data Stream Processing Systems: The State of the Art. ACM Computing Surveys (CSUR), 54(11s), pp.1-36, 2022.

[3] Corral, L., Georgiev, A.B., Sillitti, A. and Succi, G. A method for characterizing energy consumption in Android smartphones. In Proceedings of the 2nd International Workshop on Green and Sustainable Software (GREENS), pp.38-45, IEEE, 2013.

[4] Arcaini, P., Riccobene, E. and Scandurra, P. Modeling and analyzing MAPE-K feedback loops for self-adaptation. In Proceedings of 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, pp. 13-23, IEEE/ACM, 2015.

[5] Gao, Y., Diao, M. and Fujii, T. Sensor selection based on dempster-shafer evidence theory under collaborative spectrum sensing in cognitive radio sensor networks. In Proceedings of 90th Vehicular Technology Conference (VTC2019-Fall), pp. 1-7, IEEE, 2019.

[6] Halgamuge, M.N., Zukerman, M., Ramamohanarao, K. and Vu, H.L. An estimation of sensor energy consumption. Progress In Electromagnetics Research B 12, pp.259-295, 2009.

[7] Kolchinsky, I. and Schuster, A. Efficient adaptive detection of complex event patterns. arXiv preprint arXiv:1801.08588, 2018.

[8] Lee, S.K., Yoo, S., Jung, J., Kim, H. and Ryoo, J. Link-aware reconfigurable point-to-point video streaming for mobile devices. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 12(1), pp.1-25, 2015.

[9] Li, C.T., Cheng, J.C. and Chen, K., 2020. Top 10 technologies for indoor positioning on construction sites. Automation in Construction 118, p.103309, 2020.

[10] Luthra, M., Hennig, S., Razavi, K., Wang, L. and Koldehofe, B. Operator as a service: Stateful serverless complex event processing. In Proceedings of International Conference on Big Data (Big Data), pp.1964-1973, IEEE, 2020.

[11] Luthra, M. and Koldehofe, B. Progcep: A programming model for complex event processing over fog infrastructure. In Proceedings of the 2nd International Workshop on Distributed Fog Services Design, pp.7-12, ACM, 2019.

[12] Luthra, M., Koldehofe, B., Weisenburger, P., Salvaneschi, G. and Arif, R. TCEP: Adapting to dynamic user environments by enabling transitions between operator placement mechanisms. In Proceedings of the 12th International Conference on Distributed and Event-based Systems (DEBS), pp.136-147, ACM, 2018.

[13] Nardelli, M., Cardellini, V., Grassi, V. and Presti, F.L. Efficient operator placement for distributed data stream processing applications. IEEE Transactions on Parallel and Distributed Systems, 30(8), pp.1753-1767, 2019.

[14] Qin, C., Eichelberger, H. and Schmid, K. Enactment of adaptation in data stream processing with latency implications—a systematic literature review. Information and Software Technology 111, pp.1-21, 2019.

[15] R Riesebos. Smartphone-based real-time indoor positioning using BLE beacons. Master's thesis, 2021. http://fse.studenttheses.ub.rug.nl/id/eprint/25582

[16] Slo, A., Bhowmik, S. and Rothermel, K. hSPICE: state-aware event shedding in complex event processing. In Proceedings of the 14th International Conference on Distributed and Event-based Systems (DEBS), pp.109-120, ACM, 2020.

[17] Wang, Y., Han, S., Tian, Y., Xiu, C. and Yang, D. Is centimeter accuracy achievable for LTE-CSI fingerprint-based indoor positioning?, pp.75249-75255, IEEE Access 8, 2020.

[18] Weisenburger, P., Luthra, M., Koldehofe, B. and Salvaneschi, G. Quality-aware runtime adaptation in complex event processing. In Proceedings of 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), pp.140-151, IEEE/ACM, 2017.

[19] Zou, L., Javed, A. and Muntean, G.M. Smart mobile device power consumption measurement for video streaming in wireless environments: WiFi vs. LTE. In Proceedings of International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), pp.1-6, IEEE, 2017.

[20] Li, L., Zhong, B., Hutmacher Jr, C., Liang, Y., Horrey, W.J. and Xu, X. Detection of driver manual distraction via image-based hand and ear recognition. Accident Analysis & Prevention 137, p.105432, 2020.

[21] Slo, A., Bhowmik, S. and Rothermel, K. State-Aware Load Shedding from Input Event Streams in Complex Event Processing. IEEE Transactions on Big Data, 2020.