On the Cost of Shifting Event Processing within Wireless Environments

Kirsten Terfloth, Katharina Hahn
Freie Universität Berlin
Takustr. 9
Berlin, Germany
terfloth|khahn|voisard@inf.fu-berlin.de

Agnès Voisard
Fraunhofer ISST and Freie Universität Berlin
Mollstr. 1
Berlin, Germany
agnes.voisard@isst.fraunhofer.de

ABSTRACT

With the emergence of wireless sensor networks, the issues of event recognition and processing have been partially shifted into the embedded domain. New processing capabilities on small devices allow for physically close event monitoring and fast filtering without having to set up a wired infrastructure beforehand. This opportunity for flexible deployments, local data storage and demand-driven event forwarding opens up new application areas for event-centric architectures.
However, the convenience of localized event processing comes at a cost, such as sparse resources or medium contention when relying on wireless communication. Several parameters have to be evaluated to decide whether pushing the application logic into a sensor network is worthwhile, or whether a conventional server-centered deployment is to be preferred. In this paper, we discuss parameters influencing an architectural decision and their interdependencies, illustrate our contribution with the help of an example and provide a generic cost model for estimating this decision.

1. INTRODUCTION

Many application areas such as airline baggage routing, business process management, monitoring of the health of patients in a hospital or even studies in particle physics rely on recognizing, filtering and processing events (see [4] for a detailed list of example applications). Classically, the corresponding systems either receive streams of data produced by sensors (e.g. GPS data, RFID readings, physical measurements) or streams of low-level events to further reason on actions to be taken. Architectures for complex event processing stretch from stream databases over rule engines to middleware solutions, and usually run on powerful machines or clusters. Within the past years, wireless sensor networks have become available to serve as a new tool for local event processing. An environmental or business site can be monitored by simply distributing sensor nodes to cooperatively determine the state of the investigated field. Therefore, each individual sensor node is equipped with a variety of application-specific sensors to sample its physical surrounding, a microcontroller to allow for further processing, some secondary storage and a transceiver to enable ad-hoc wireless communication with other nodes in the network. Furthermore, actuators may be mounted to a node to directly trigger physical actions. A reduction of the size of sensor nodes provides a less invasive technology but also harsh limitations in terms of resources such as processing power, memory and energy. Thus, efficiency in resource usage is of great importance. The usage of wireless sensor networks in an event processing context is very natural. With their ability to provide processing capabilities right at the sources of data generation, these embedded devices can take different roles in the event processing chain: Their tasks may span from acting simply as a filter on the incoming data streams of their sensors, over providing event recognition locally on each node up to enabling complex, distributed event processing involving spatio-temporal event pattern matching. The decision on which part of an event processing task should be shifted into the network or whether a central approach is much more feasible is not trivial. Many different parameters have to be taken into account for choosing an application specific architecture. Numerous papers address the minimization of dedicated costs such as energy spend per node on a routing algorithm or memory consumed by a proposed datastructure. A more abstract view on the information which parameters to consider and their corresponding impact on the overall cost has to be derived from literature. This paper therefore primarily addresses networking and hardware related costs to be considered when opting for a distributed event processing scenario, but does not incorporate monetary costs. The contribution of this paper is hence to identify and correlate parameters influencing this decision, evaluate their interdependencies and provide a qualitative model to estimate costs. Note that for the time being, we do neither consider the risk of failure of nodes, nor the probability of erroneous event reporting.
The remainder of the paper is organized as follows: In section 2, we present prominent Early-Warning Systems as representatives of event-processing architectures and work related to the discussion of cost parameters. A brief introduction of an exemplary application scenario with two opposite system configurations is discussed in section 3 to visualize the domain of the proposed cost model which is defined in the following section 4. Section 5 evaluates the utility of the model, before concluding the paper in section 6.

.

Dagstuhl Seminar Proceedings 07191
Event Processing
http://drops.dagstuhl.de/opus/volltexte/2007/1141

2. RELATED WORK

Reduction of costs in its various occurrences is the target to achieve for most research contributions. Therefore, the presentation of specific optimizations in this section is avoided in favor of focussing on three topics: After relating the contribution of this paper to previous work done in the event filtering domain, a variety of existing or past projects that can benefit from having a cost estimation model is pointed out. We also reference two projects of the networking domain that consider common tradeoffs in system and algorithmic design. In the past years, a significant amount of work has been carried out in the area of event filtering, and more particularly event selection and consumption (see e.g. [11]). In this paper, we study two reference infrastructures (i.e., centralized and decentralized) to support these basic event operators. As far as event definition is concerned, it is now common to distinguish atomic, simple event (e.g., the value of a temperature sensor at a certain time) from complex events, made of other (complex or atomic) events, see [7]). As stated in [1] the composition of events can be based on a casual, spatial, or temporal correlation. Here, we focus on spatial and temporal correlations and the two architectures that serve as a reference in the paper support complex event processing (CEP). A specific class of event processing systems dealing with data gathered from distributed data sources are Early-Warning Systems (EWS). These include i.e. the Tsunami Early Warning System presented in [5], a seismic EWS for nuclear power plants described in [10] or a wildfire warning system presented in [3] to name but a few. The devices utilized to take measurements range from seismic stations in the first cases to embedded sensor nodes in the latter, employing wireless connectivity either through satellite or radio communication towards a central processing entity. Another warning system that furthermore not only collects data, but also provides a feedback loop to control and re-adjusts its data sources is a warning system for hazardous weather conditions presented by Kurose et. al. in [6]. Data provided by low-power X-band radars concerning the lowest few kilometers of the earth's atmosphere is forwarded to a central command and control entity where event features are extracted and stored in a feature repository. Dependent on these features, end-user preferences and policies to identify areas of meteorological interest, a control loop optimizes the configuration of each radar and sends back this repositioning information to the participating nodes accordingly. This collaborative work of the data sources can be used to provide a better precision and sensitivity of the meteorological data for a monitored region in the atmosphere, thus a better resolution than can be obtained by a single radar. In the discussion on future challenges, the authors point out the necessity to evaluate distributed event detection in networks featuring scarce resources to still be able to provide maximal system utility.

Especially in the domain of embedded networked sensors where resource constraints force an application programmer to consider system behavior before the actual deployment, a number of publications illustrate tradeoffs between system parameters. In [9] a hash table approach for storing events inside a sensor network instead of streaming them towards a base station is suggested. The authors provide an estimation for the communication costs the network is exposed to for different types of storage and varying numbers of events and queries for events. They suggest a mechanism for load balancing among the nodes and thus illustrate the tradeoff between push- and pull-based data acquisition in sensor networks. A tradeoff between computation and communication has been identified in [8], proving communication to be the most energy intense operation on a node. A common technique to avoid energy wastage is to aggregate data before sending it, thus favor local processing on the nodes wherever possible.

3. SCENARIO DESCRIPTION

To illustrate the scope and application area of our cost estimation model and provide an idea of the impact it may have on planning future system architectures, two complementary system designs targeting the same application will serve as a sample visualization throughout this paper. This application aims at detecting and prediction of conditions leading to avalanches. The goal of this research area is to be able to predict precisely where and when the risk of an avalanche going downhill is high. Commonly, this risk is indicated with the help of different levels of danger that change under the influence of static and dynamic parameters of a certain, monitored area [2]. For an evaluation on local avalanche risk static and dynamic parameters have to be taken into account. Static parameters include different features of the ground such as steepness of its elevation, soil constitution and predominant flora. Dynamically changing weather conditions including amount, thickness and weight of snow, temperature and pressure, but also the history of these conditions furthermore add to the evaluation. A distributed system setup to detect events resulting in rising or lowering the risk indication level has to work in a robust, cost-efficient manner, otherwise the lifetime of the network may not cover a complete season. Furthermore, timely notification of people within the danger zone and possibly other subscribers is a crucial part of an event-warning system, thus event delivery has to be concerned accordingly. The two following scenario descriptions exemplify the two extreme ends of system design for avalanche risk detection to enable the illustration of tradeoffs between chosen parameters.

3.1 Configuration I - Centralized Event Processing

Configuration I, a classical centralized system design, features lightweight, embedded sensor nodes capable of taking all the measurements needed. These devices sample their environment regularly, tag the corresponding data points with a timestamp and their node id and forward them to a base station, possibly relying on a multi-hop path. The base station, a machine not constrained in terms of resources, calculates the current, localized risk levels depending on the newly received data and available spatio-temporal information, typically static parameters and event history. Subscribers will be notified directly by the base station in case the interpolated avalanche risk level is violating a pre-set threshold. This configuration is visualized in figure 1.

3.2 Configuration II - Distributed, In-Network Event Processing

Configuration II features a completely distributed system implementation as commonly suggested by the wireless sensor network community, see figure 2. The deployed sensor nodes are not only able to sample the environmental parameters in question, but are also aware of their spatial context

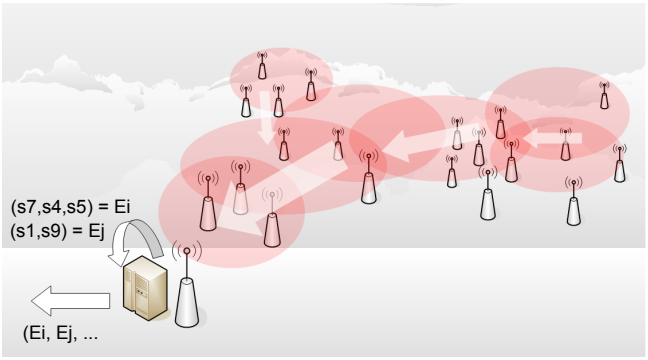

Figure 1: In a centralized configuration nodes stream raw data samples s_i towards a central entity for event recognition.

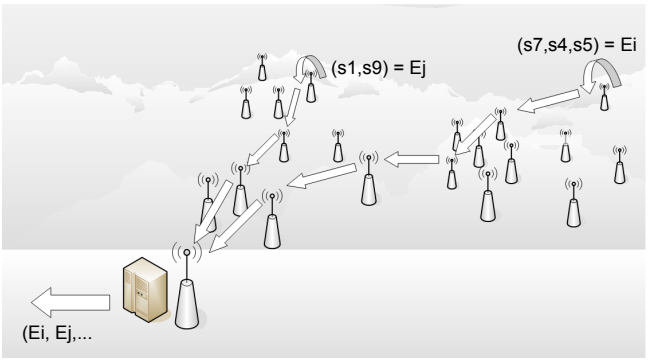

Figure 2: In a distributed configuration nodes detect events locally and send notifications to a base station.

and the temporal development of their sensed values. Data sampled is aggregated and processed locally on each sensor node, as opposed to sending raw data to a base station, and hence the task of detecting events is shifted from the central entity to the local instances of the network. An evaluation of the risk level that leads to a significant change will be reported to the base station upon detection, and a notification of the subscribers to this information may then be issued.

4. INFLUENTIAL COST PARAMETERS

Costs are commonly defined as a value or resource not available any more after using it to accomplish a specific task. When looking at costs for event processing, the basic entity to calculate a certain cost for is a single event E_i. Since specifying what exactly to take into account for a cost estimation in event processing is highly domain-dependent, we decided to focus on three main cost parameters which strongly influence the decision on which configuration to favor:

- The generated *network load*, denoted by $C_{network}(E_i)$, describes the number of packets which need to be transferred within the network for event recognition. We assume packets to be of a fixed size and the costs for routing to be only dependent on the number of hops traversed thus abstracting from a specific routing algorithm. This is feasible as it holds for the comparison of different system setups.

- Depending on the application domain, the *delay* of event recognition is important in order to be able to determine whether the requirements given by the application can be fulfilled. The delay of event recognition is denoted by $C_{delay}(E_i)$. It defines the time it takes to transfer the needed information to a base station.

- The *storage* needed in order to reliably detect and process events is indicated by $C_{storage}(E_i)$. This cost is an important factor to determine the resource settings of sensor nodes and therefore their physical dimension, but can be disregarded on the base station.

In the scope of this paper, we examine two dimensions of events which strongly influence the impact of the cost parameters on the total cost for a chosen configuration: The first distinction is drawn between local and distributed event detection. This distinction is straightforward as it considers the number of nodes which are involved within the process of event recognition. While in a local detection an event may be recognized by a single node, the data of several sensor nodes has to be aggregated for a distributed event. This may either be the case when the sensing range of a node cannot cover the complete area affected by an event or the accuracy of a single sensor involved is not sufficient to guarantee the precision demanded. On the other hand we consider the time-scale involved in the recognition of an event, namely the amount of previously sampled data that has to be taken into account to detect events. In case an event can be immediately derived upon the arrival of a data sample, we refer to this as an instantaneous event. History-sensitive events rely on previously sampled data items. Local acquisition of data samples on a sensor node affects the storage needed on each device. Both spatial and temporal event complexity are commonly subsumed under the concept of composite events, but have to be addressed separately in this context due to their differing impact on the investigated costs. According to these two dimensions of events, four different event classes are derived:

- EC_1 (local, instantaneous event detection): This class of events features the most basic event detection possible, where only one node is involved and no historical information has to be kept. An example of such an event is a notification if any sensor senses a temperature below zero degrees.

- EC_2 (distributed, instantaneous event detection): Events span over a spatial area, thus involve the participation of several nodes for detection, but memory has not to be provided. A representative of this event class is a notification upon the recognition of a temperature below zero for a certain area (e.g. a hill).

- EC_3 (local, history-sensitive event detection): Detection of events is only possible in regard to the temporal context, thus data storage is mandatory. An example of this event class is to notify a subscriber if a sensor node has detected a temperature below zero for the last five minutes.

- EC_4 (distributed, history-sensitive event detection): Events span over a spatial area and have a temporal context that has to be stored. This kind of events include notifications if the temperature has been below zero for the last five minutes within a certain area.

For the sake of simplicity, we assume the deployment interval for both configurations to be of a constant time interval I divided into e time units (or epochs). r denotes the sampling rate measured by the number of samples taken per epoch. The number of active sensors on a node is k. The operational mode of the nodes is always on, i.e. we do not consider possible sleep-times of nodes. The total number of nodes within each configuration is denoted by n.
In the following we examine the costs of each event class within the sketched application scenarios.

4.1 Cost of Centralized Event Processing

This classical architectural setup relies on a central entity that consumes raw data streams to apply event semantics. We consider the reporting rate of data items to equal the sampling rate and a data sample to fit into exactly one packet. As mentioned before, within this configuration storage is not a critical resource, thus its corresponding cost $C_{storage}(E_i)$ is neglected.

4.1.1 Network load

All sensor nodes of the network forward their data samples to the central entity according to the given sampling rate r of k different sensors within an epoch e. They also possibly relay samples of other nodes sent to a base station (BS) in a multi-hop network. The cost function $C_{network}(EC_i)$ in terms of packets transmitted for recognizing events of all event classes EC_i can therefore be described as

$$C_{network}(EC_i) = \sum_{j=0}^{n} (r * e * k) \sum_{m=0}^{n} \chi(N_j, N_m)$$

$$with\ \chi(N_j, N_m) := \begin{cases} 1 & N_m\ part\ of\ route\ N_j\ to\ BS \\ 0 & otherwise \end{cases}$$

4.1.2 Delay of event detection

If each node simply forwards its samples, the delay of the event detection is given by the time the network needs to forward the samples correlated to the event $T_{route}(N_j, BS)$ (and thus depends on the location of the node N_j in regard to the base station) and the time the base station needs to process the incoming samples $T_{p,BS}(EC_i)$. Hence the cost function $C_{delay}(EC_i)$ for calculating the delay is

$$C_{delay}(EC_i) = T_{route}(N_j, BS) + T_{p,BS}(EC_i)$$

4.1.3 Conclusion

In a completely centralized setting, the notion of an event is formed at the central entity. As a consequence, data samples of all data sources have to be routed to this entity for event detection, independent of both the number of nodes that possibly contribute to the detection (local vs. distributed) and the number of samples of each node that have to be taken into consideration (instantaneous vs. history-sensitive). Hence, the corresponding cost $C_{network}(EC_i)$ is constant for all event classes EC_i. Looking at the cost in terms of delay $C_{delay}(EC_i)$, the cost estimation can also be described with the same function for all event classes, but is dependent on the position of a node in the network, and the processing time necessary at the base station which will differ among event classes. An event raised by the reception of a data sample reported by a node close to the central entity will have a shorter delay than one raised on the opposite edge of a multi-hop network.
Remarkably, the costs for event processing in a centralized manner does neither depend on the event classes, nor on the frequency of the occurrence of events.

4.2 Cost of Distributed, In-Network Event Processing

Distributed event detection involves in-network processing which means that event semantics are applied on the node themselves. Instead of sending data items to a central entity, only the occurrence of an event without the corresponding raw data items are reported. For comparability reasons, we assume an event notification to fit into one packet. In the following, $f_{local}(E_i, N_j)$ denotes the frequency of the local occurrence of an event E_i at node N_j. $f_{spatial}(E_i, N_m)$ the occurrence of a spatially distributed event E_i at the master node N_m coordinating distributed event detection among nodes, where n_{min} is the minimum number of nodes participating in recognizing such a distributed event. Note that $f_{local}(E_i, N_j)$ is typically higher than $f_{spatial}(E_i, N_m)$ since the later depends on the agreement of several nodes. Furthermore, the size of the history window for history-sensitive event recognition is denoted by Δt, a fraction of the interval I, thus $\Delta t = l{*}e$ with $l \in N$.

4.2.1 Network load

A sensor node will report the local recognition of an event when a corresponding data item leading to meeting the event condition has been sampled, hence the cost in terms of packets transmitted for detecting a local, instantaneous event is simply

$$C_{network}(EC_1) = \sum_{j=0}^{n} f_{local}(E_i, N_j) * \sum_{m=0}^{n} \chi(N_j, N_m)$$

$$with\ \chi(N_j, N_m) := \begin{cases} 1 & N_m\ part\ of\ route\ N_j\ to\ N_m \\ 0 & otherwise \end{cases}$$

Distributed event recognition relies on message exchange of several nodes. Given a predefined clustering with a determined master node, the simplest algorithm to achieve the notion of a spatial event is for a node raising the local event to notify its master node. In case a predefined threshold number of local event occurrences are reported by the participating nodes within a time frame, i.e. an epoch, the master sends out an event notification to be routed to the base station. With M being the set of all master nodes within the network, the cost this collaborative event recognition imposes can therefore be described as:

$$C_{network}(EC_2) = \sum_{\forall m: N_m \in M} (f_{spatial}(E_i, N_m) * \sum_{j=0}^{n} \chi(N_m, N_j)$$

$$+ \sum_{\forall s: N_s \in S_m} f_{local}(E_i, N_s) * n_i * \sum_{q=0}^{n} \chi(N_s, N_q, N_m))$$

$$with\ \chi(N_i, N_j, N_k) := \begin{cases} 1 & N_j\ part\ of\ route\ N_i\ to\ N_k \\ 0 & otherwise \end{cases}$$

Note that the network load is not affected by history-sensitive

types of event since this information is processed locally on each node. Therefore, the associated cost functions for $EC3$ and $EC4$ are simply equal to their instantaneous counterparts:

$C_{network}(EC_3) = C_{network}(EC_1)$
$C_{network}(EC_4) = C_{network}(EC_2)$

4.2.2 Delay of event detection

The cost function to specify the delay of reporting a local event of event class $EC_1$is the sum of the time needed to process the data on the sensor node to raise this event and for routing the event notification to the base station.

$$C_{delay}(EC_1) = T_{p,local}(E_i) + T_{route}(N_j, BS)$$

Since distributed event reporting relies on the cooperation of several nodes, the respective cost function $C_{delay}(EC_2)$ for the delay reflects this in terms of additional time needed. Local node processing time, time spend for routing the event from the location of local event occurrence N_s to a master node N_m and from a master to the base station as well as the maximum time fraction a master node waits for incoming local event notifications including the processing time $T_{p,spatial}(N_m, E_i)$ sum up to the overall costs in delay.

$$C_{delay}(EC_2) = \\ T_{p,local}(N_s, E_i) + T_{route}(N_s, N_m) \\ + T_{p,spatial}(N_m, E_i) + T_{route}(N_m, BS)$$

Events involving the evaluation of history information on a node contribute to the overall complexity of the event detection scheme. This increase in complexity can be observed as a longer processing time depending on the size of history information considered per event $T_{P,mem}(E_i)$, as well as an increased demand for data storage addressed in the next subsection. Therefore, the delay for event classes EC_3 and EC_4 can be expressed as

$C_{delay}(EC_3) = C_{network}(EC_1) + T_{p,mem}(E_i)$
$C_{delay}(EC_4) = C_{network}(EC_2) + T_{p,mem}(E_i)$

4.2.3 Data Storage

While network load and delay are parameters of a cost function that have to be taken into consideration independent of the utilized class of devices chosen as data sources, storage and processing capability are only critical resources when deploying embedded devices. Within a centralized event processing configuration we assume all nodes to be able to fulfill the minimum requirements for providing store and forward mechanisms. In case a distributed, in-network event detection configuration is provided, the cost of storage has to be explicitly addressed. Note that the additional storage cost for event classes EC_1 and EC_2 are disregarded as they do not employ temporal context and the costs depicted are costs per node.

$$C_{storage}(EC_3) = l * k * r$$

For determination of the storage costs for events of EC_4, a distinction between master nodes and all others is necessary. In the later case, the corresponding cost equal those of EC_4, while master node have to provide additional resources for incoming local event messages.

$$C_{storage}(EC_4) = l * k * r + n_{min}$$

4.2.4 Conclusion

In a distributed event processing architecture used for localized event detection two major effects can be observed: First of all, the network load is primarily depending on the number of events locally detected for all event classes, hence is independent of the sampling rate. However, this advantage comes at the prize of a higher processing burden on the nodes, higher storage costs to enable the detection of temporally composite events and a lack of raw data at a central entity. Furthermore, the cost in terms of delay for event notification, unless a distributed detection schema is chosen, strongly relates the event complexity to the availability of local processing capabilities. As a consequence, the delay for routing is not the dominant factor for choosing a configuration, but rather the deployed hardware of the data sources.

5. COST MODEL UTILITY

The benefit of applying a cost estimation model as presented in the last section for choosing a specific network configuration for event processing can be best evaluated with the help of a use case. Considering the avalanche scenario as presented in 3, the idea is to clarify the interdependencies of the identified parameters by varying the operational circumstances of the application domain.. This study is far from being exhaustive, it rather tries to point out correlations as well as critical measures that heavily influence an infrastructural decision.

5.1 Data Rate vs. Event Rate

High data rates are usually needed when the occurrence of critical events (e.g. the rise of the risk level for an avalanche to a critical value) has to be detected with a minimal delay, or the temporal complexity of events demand for a high data granularity. On the other hand, the event rates are bounded by the data rate while the event rate does not influence the data rate. Note that from a data perspective, events can be seen as a semantical compression of data streams, so the increase of the event rate in consequence of an increase of the data rate will be orders of magnitude smaller.

A higher data rate will directly impact the network load within the centralized setting, as described in section 4.1.1. Highering the data rate will therefore lead to a factorial increase of the network costs of an event. Within the distributed, in-network configuration, a higher data rate does not influence the network costs whatsoever (unless it correlates with the event rate).

Whenever the event rate within the application scenario is raised (which is basically determined by the application domain), the distributed, in-network configuration has to accept higher network load. As long as the capabilities of the base station are not exceeded, a centralized approach is oblivious to a variance of the event rate.

5.2 Time-Critical Event Detection vs. Event Complexity

Early Warning systems demand for event reporting within a predefined, usually very tight time interval. When evaluating whether this interval can be met with a chosen configuration, two parameters dominate this analysis: the time needed for routing and the time needed for processing, independent of the considered event class or choice of scenario. Based on the fact that for a similar network layout, both the centralized and the distributed configuration will face the same conditions for routing a data packet to the base station, the critical resource to observe is the processing capability of a node. In case the event complexity increases in such a way, that in-network event detection time dominates the delay in an extent that is not tolerable, a centralized approach is to be favored. Note that there is of course a correlation of network load and delay as soon as network contention has a negative effect on the delivery of time-critical data, which has to be addressed when designing an appropriate routing scheme.

5.3 Energy Consumption vs. Temporal Event Complexity

When deploying a network of embedded sensor nodes, the available amount of energy per node directly influences network and thus application lifetime. Since communication is the most energy intense operation a node can execute, algorithms generally prefer processing over communication whenever applicable.

In a distributed setting, a high temporal event complexity demands for high storage capabilities on sensor nodes. Energy-wise these are to favor over a the centralized configuration as long as the energy spent on network load exceeds the corresponding costs in terms of energy for local event processing. Higher temporal event complexities (accounted for in number of sensitive historical samples) lead to a greater energy consumption for event processing considering a distributed approach, but due to the inherent data compression nature of events, this will highly cut the costs in network load.

6. CONCLUSIONS

Within this paper we discussed the shift of event processing from centralized systems to distributed sensor networks by means of the example of avalanche detection. Within this application scenario, we sketched two different system configurations which both allow for reasonable risk detection. In order to identify the advantages of each setting and thus the favorable solution, we defined four classes of events that strongly correlate with the benefits of each configuration. Considering those event classes, we introduced a generic cost-model which determines the dominant cost-measures when it comes to evaluating the different set-ups. Applying the cost-model to the introduced application scenario, we identified decisive conflicts of objectives that lead to an application-dependent decision.

7. REFERENCES

[1] A. Adi and O. Etzion. Amit - the situation manager. *The VLDB Journal*, 13(2):177–203, 2004.

[2] Canadian-Avalanche-Centre. International danger scale. http://www.avalanche.ca/default.aspx?DN=8,4,558,3,Documents, May 2007.

[3] D. Doolin and N. Sitar. Wireless sensors for wildfire monitoring. In *Proceedings of the SPIE Symposium on Smart Structures Materials/ NDE 2005*, March 2005.

[4] O. Etzion, M. Chandy, and R. v. Ammon. Proceedings of the dagstuhl seminar on event processing, seminar n 07191,06.05.2007-11.05.2007 (to appear). Dagstuhl Research Online Publication Server, 2007.

[5] GTZ-Potsdam. German indonesian tsunami early warning system. http://www.gitews.org/, May 2007.

[6] J. Kurose, E. Lyons, D. McLaughlin, D. Pepyne, B. Philips, D. Westbrook, and M. Zink. An end-user-responsive sensor network architecture for hazardous weather detection, prediction and response. In *Proceedings of the Asian Internet Engineering Conference (AINTEC 2006)*, pages 1–15, November 2006.

[7] C. Liebig, M. Cilia, and A. Buchmann. Event composition in time-dependent distributed systems. In *Proceedings of the 4th Intl. Conference on Cooperative Information Systems (CoopIS 1999)*, pages 70–78, Edinburgh, Scotland, Sept. 1999. IEEE Computer Society Press.

[8] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *Proceedings of the ACM SIGMOD (SIGMOD 2003)*, June 2003.

[9] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with GHT, a geographic hash table. *Mobile Networks and Applications (MONET)*, 8(4):427–442, 2003. Special Issue on Wireless Sensor Networks.

[10] M. Wieland, L. Griesser, and C. Kuendig. Seismic early warning system for a nuclear power plant. In *Proceedings of the 12th World Conference on Earthquake Engineering (WCEE 2000)*, February 2000.

[11] D. Zimmer and R. Unland. On the semantics of complex events in active database management systems. In *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, page 392, New York, N.-Y., USA, 1999. IEEE Computer Society.