Context-Dependent Event Detection in Sensor Networks

Scarlet Schwiderski-Grosche
Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
scarlet.schwiderski-grosche@rhul.ac.uk

ABSTRACT

Event-based systems are well suited for application in sensor networks. Compared to the traditional application domains of event-based systems however sensor networks impose a number of challenging requirements. In this short paper, we present an architecture designed to address these requirements: the CoDED platform for context-dependent event detection.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communications

General Terms

Design, Theory

Keywords

Composite event detection, (mobile) sensor networks, context-awareness

1. OVERVIEW

In sensor networks, event-based systems represent a powerful mechanism for monitoring real-world phenomena while providing for efficient processing and communications (see [1]). As opposed to traditional DEBS (Distributed Event-Based Systems) however sensor networks create a highly dynamic application environment. On the one hand, the network topology changes continuously due to node mobility and node lifetime. On the other hand, the observed physical environment experiences constant and often unforeseeable change. The requirements on composite event detection are therefore different from traditional DEBS. Related work has neglected the dynamics of the application environment (see e.g. [1], [2], [3]). A new approach is required in which composite event detection is made adaptive.

In (mobile) sensor networks[1], the following general characteristics apply:

There is no established infrastructure or centralised administration. Consequently, composite event detection needs to be fully distributed and autonomous.

Every node acts as router and forwards packets to the next hop. Hence, neighbouring nodes need to interact to detect global composite events.

Sporadic connectivity and node mobility result in a constantly changing network topology. Hence, composite event detection needs to adapt to ever-changing contexts. Since nodes may be unavailable, composite event detection needs to be redundant.

Link and node capabilities vary. In order to reduce energy consumption, wireless communication and processing overhead need to be minimised. Therefore, event processing should be done at the source and should be restricted to those events that are relevant in the current context.

Adaptivity of composite event detection constitutes a major challenge. In order to avoid unnecessary processing and communications, events should only be monitored in their relevant context. This represents a major difference to related work where events are filtered after their detection (see e.g. [4]). Consider an Alpine sensor network where mountaineers are equipped with a personal health monitor that measures, among other things, heart rate, location, speed, altitude and time. If augmented with a distance sensor and an acceleration sensor, such device could act as an avalanche beacon as proposed in [5]. Hence, this type of mobile node serves as personal health monitor in one context and as avalanche beacon in another. In case of an avalanche emergency, all resources should be directed to processing events in the "avalanche context".

In this short paper, we develop an architecture for context-dependent event detection: *the CoDED (Context Dependent Event Detection) platform.* Here, composite event detection is fully distributed, with primitive events occurring throughout the network and nodes collaborating to detect system-wide or *global* composite events. The novelty of CoDED lies in the fact that composite event detection is context-dependent: events are only monitored when their "context is active". CoDED consists of the following three layers:

Context layer: Here, a number of *default monitoring contexts* are specified in simple propositional logic (PL); a context is active while its PL formula evaluates to **true**, inactive otherwise. Moreover, we propose a *fallback context* when no default monitoring context applies and an *initialisation context* that determines which contexts are currently active.

Event detection layer: This layer comprises one *CED*

[1]Here, we are particularly interested in sensor networks employing mobile nodes.

(CompositeEventDetection) engine for each context on the context layer where respective ECA rules are implemented.
Event signalling layer: A continuous stream of events is signalled at this layer. Primitive events originate locally. Moreover, primitive and composite events are signalled from CoDED platforms on the local and on remote nodes.

CoDED is not aimed at general context detection but rather at the detection of specific monitoring contexts. For example, the nodes of a sensor network may be designed for an outdoors scenario with monitoring contexts $outdoors \wedge hot \wedge dry$ and $outdoors \wedge cold$ and $outdoors \wedge wet$. Depending on the values of context variables, a monitoring context is either active or inactive. Choosing a logic-based approach for context specification is in line with existing work on context-awareness (see e.g. [6]).

Due to the dynamic application environment, nodes may experience unforeseen circumstances that are not covered by the default monitoring contexts. For example, our outdoors scenario sensor may be exposed to a sandstorm. A *fallback context* is consequently needed to cater for basic monitoring needs.

The design of CoDED requires one context to be always active, the *initialisation context*. This context implements event detection mechanisms that realise context detection. For each context variable in the context space (e.g. *outdoors, dry, hot*), the initialisation context implements one or more ECA rules that implement its detection. A change in value (from **true** to **false** or vice versa) causes the affected contexts to be reevaluated. For example, in the outdoors desert scenario the context variable *outdoors* occurs in contexts $outdoors \wedge hot \wedge dry$ and $outdoors \wedge cold$ and $outdoors \wedge wet$. Hence, when *outdoors* changes all three contexts need to be reevaluated. Depending on the outcome, context changes may occur. Using event detection mechanisms to derive high-level contexts is not new. In [7], Tan et al. investigate event-driven context interpretation to derive high-level contexts.

For each context on the context layer, a set of relevant ECA rules is specified. Composite event detectors are constructed from the event expressions given in the E-parts of ECA rules. Conditions are given in the C-parts and checked when the respective E-part events have been detected. The A-parts specify how to react when the condition is valid. Each set of ECA rules is implemented on the event detection layer of CoDED and forms the context's so-called *CED (CompositeEventDetection) engine*.

At runtime, the stream of events is fed into the CED engines of all active contexts. Detected composite events are processed locally or broadcast to neighbouring nodes where they join the event stream on the event signalling layer. The runtime behaviour of CoDED raises a number of interesting questions:

When a context becomes deactivated, processing stops and partially detected events remain in the context's CED engine. We refer to this problem as *context retention*. The runtime behaviour at subsequent context activations depends on the solution of the context retention problem. We propose to introduce *event lifetime* to deal with context retention. For each type of primitive event in a CED engine's composite event detectors, a lifetime is assigned. The lifetime can either be a *relative lifetime*, relating to the time of deactivation of the current context, or an *absolute lifetime*.

Another interesting problem is caused by *context fluttering*. Due to inaccurate sensor readings, context changes may occur in a flutter, causing an affected context to be activated and deactivated erroneously in close succession. Let us assume that the *hot* context is active when temperature events signal a temperature of 25 degrees or above. Due to inaccurate sensor readings, the measured temperature may jitter between 24 and 25 degrees, meaning that the *hot* context flutters. A first challenge is to detect context fluttering. A second challenge is to deal with it. We propose to implement *hysteresis* to deal with context fluttering. Hysteresis is "the dependence of the state of a system on the history of its state". Using hysteresis, the *hot* context is activated when a temperature event signals a temperature of $25 + h$ and deactivated at $25 - h$, where h is the hysteresis parameter. The value of h needs to be bigger than a sensor's error of measurement and is application specific.

Network delays cause another problem in CoDED. In sensor networks, it is not feasible to defer composite event detection and wait for all delayed events from remote sites. Hence, events cannot be consumed in the order of their occurrence. Rather, they must be consumed in the order of their arrival. In CoDED, network delays may result in the arrival of events whose timestamp corresponds to an earlier context activation; in the meanwhile the relevant context has been deactivated. For CoDED, we propose *high-importance* event types which are always evaluated. Late events of other event types are discarded.

2. CONCLUSIONS

Further work includes the formal reasoning about event occurrences and context changes as well as optimisation of context layer design vs. ECA rule splitting. Moreover, the platform will be implemented using the Java-based ad hoc network simulator JiST/SWANS.

3. REFERENCES

[1] K. Römer and F. Mattern. Event-based Systems for Detecting Real-World States with Sensor Networks: A Critical Analysis. In *DEST Workshop on Signal Processing in Wireless Sensor Networks at ISSNIP*, pages 389–395, December 2004.

[2] S. Li, S. Son, and J. Stankovic. Event detection services using data service middleware in distributed sensor networks. In *IPSN'03*, Apr 2003.

[3] A V U Phani Kumar, A.M. Reddy V, and D. Janakiram. Distributed collaboration for event detection in wireless sensor networks. In *MPAC '05*, 2005.

[4] L. Fiege, M. Mezini, G. Mühl, and A.P. Buchmann. Engineering event-based systems with scopes. In *ECOOP'02*, 2002.

[5] F. Michahelles, P. Matter, A. Schmidt, and B. Schiele. Applying wearable sensors to avalanche rescue. *Computers and Graphics*, 27(6), 2003.

[6] A. Ranganathan and R.H. Campbell. An infrastructure for context-awareness based on first order logic. *Personal and Ubiquitous Computing*, 2003.

[7] J. G. Tan, D. Zhang, X. Wang, and H. S. Cheng. Enhancing semantic spaces with event-driven context interpretation. In *Pervasive'05*, 2005.