(Reactive + Proactive Behavior) ⇒ Situation Awareness in Sensor Networks

Goce Trajcevski and Peter Scheuermann
Dept. of Electrical Engineering and Computer Science
Northwestern University
Evanston, Il 60208
{goce, peters}@eecs.northwestern.edu

1. Introduction and Motivation

In order to fully exploit all the potentials of various heterogeneous resources in sensor network applications in a context-aware manner, coordinated research efforts need to be pursued in multiple technical areas including networking, information management, control, and spatio-temporal data/events management. At the heart of the motivation for our research is the observation that there is a lack of a *unified paradigm* that will enable dynamic balancing between the Quality of Data (QoD) assurances and the coordination of the resource-utilization, not only as *reaction* to an occurrence of a particular event, but also with a *proactive* impact on how the future tracking and data gathering should be managed. These are crucial capabilities that are needed for implementing any system that relies on automated situation awareness. Our work is focusing on adaptive and environment-aware integrated management of spatio-temporal queries and events in heterogeneous sensor networks.

In lieu of *the Network is the Database* perspective [2], some recent works have addressed the goal of *declarative networking* which essentially aims at providing high-level declarative languages for managing the behavior of the network (e.g., [9]). In particular, the DSN (Declarative Sensor Networks) paradigm combines a language, compiler and run-time system for high-level specification and deployment of large variety of applications [4]. The language – *Snlog*, is a variant of *Datalog* [8] with two notable extensions: "@"-symbol for denoting that the type of the variable is a *node*, and "#"- denoting a *link*. In the mid-1990s, deductive databases and active databases (ADb) [5] co-developed, both having a similar goal: providing expressive languages to enable users to easily specify *what* the system needs to do, without getting into the details of *how*. However, ADb leaned more towards the specification of the *reactive behavior* of the systems and arguments were made that the deductive and active databases can/should be viewed as two end-points of the spectrum of the family of rule-based languages in databases [10], along with attempts for their integration [3]. Although declarative languages are emerging in sensor networks settings, the *triggers*, which are the basic mechanism for specifying the reactive behavior in ADb, have not yet found their counterpart. In this position paper, we focus on efficient management of two types of information pertaining to mobile entities: (1) *location-in-time* and (2) *context*, and we combine the behavioral and semantic-knowledge with the optimal resource utilization. We will consider the collaborative behavior of two types of sensors used for detecting the information of interest: (1) low-cost and low-energy acoustic and seismic (vibration-based) sensors, and (2) multiple video sensors. Towards the goal of situation awareness, we present a methodology that, in reaction to a particular event in a given state of the system, will proactively adjust the resources' exploitation, so that the highest Quality of Data (QoD) level is assured in balance with some parameters of interest, e.g., networks connectivity and lifetime.

2. Evolving Triggers and Our Research

To motivate the need for specifying (and managing) the reactive behavior in sensor networks, and to illustrate the basic rationale of our goals, observe the scenario depicted in Figure 1 and consider the following request: **Req1:** *"When a moving object has been continuously moving towards the region of interest for at least 4 min., If the Marines units are further than 2 miles from the region, notify the closest infantry unit.* ***Subsequently:*** *when that object has been continuously within 100 yards from the boundary of the region more then 2 min., if the cameras have identified him as a person carrying a bag, notify the closest patrol vehicle and guide it towards the mobile object.* **Req1** has all the elements of the traditional

triggers from the ECA (Event-Condition-Action) paradigm, however, there are a few important observations that need to be made: (1) The event ("when") is *composite* and its *primitive* constituents are distributed *(location, time)* values obtained by the tracking sensors. Still, the detection(s) of the composite event(s) can be done in a manner that does not require forwarding every detection of an individual instance of the primitive event to the sink. (2) The condition ("If") can be viewed like a remote but *instantaneous* query, to be executed upon detection of the composite event. However, in reality, it is a *continuous* one [1], because the distance of the Marines units from the region of interest changes over time. In addition, the value of the composite event (*moving towards*) also changes over the time. (3) **Req1** contains a portion ("**Subsequently**") which, in a sense is a nested *"child-trigger"*. It may be tempting to declare it as a separate trigger, however, in that case it may defeat the purpose of being *enabled* in a correlated-manner with its *"parent"* trigger which, in turn, may cause an unnecessary waste of network resources. (4) The network itself needs the capability to react to various stimuli, i.e., processing some other query like, for example ***Q1*** in Figure 1, may enforce re-routing of the notifications towards the sink – aside from the possible mobility of the sink itself. Figure 2 illustrates an SQL-like syntax of a trigger corresponding to **Req1**, along with a possible time-line of the evolution of the environment. Actually, this is an example of triggers conforming to the $(ECA)^2$ (Evolving and Context-Aware Event Condition Action) paradigm that was introduced in [7]. We also showed how such triggers can be used to pro-actively influence the behavior of the system. However, our main assumption was that they execute on top of ORDBMS equipped with CQ (Continuous Queries) processing capabilities [1] in streaming environments. Sensor Networks pose unique challenges in terms of their load balancing and efficient resource utilization. Based on our earlier experiences for translating triggers and workflow-descriptions [6] specified in high-level languages into logic programs, one of the goals that we are currently focusing upon is:

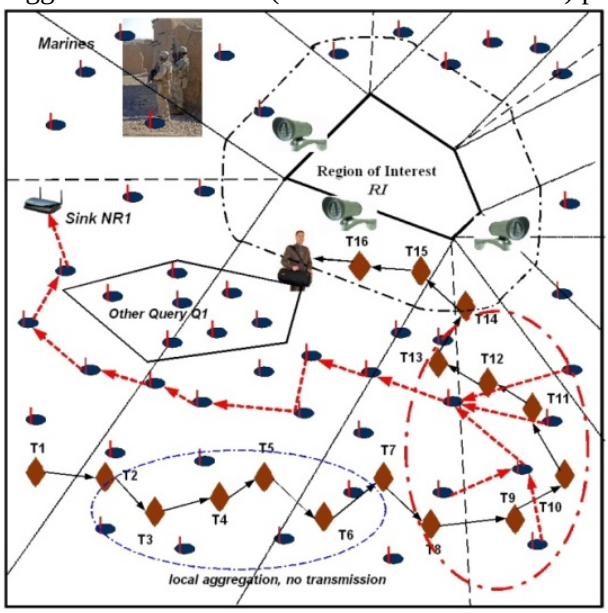

Figure 1. Reactive + Proactive Behavior

```
ON E_MovingTowards(O,'RI',4,T)
IF Distance(Marine,'RI',X,T1) AND T1 >= T AND X > 2miles
                Within_Event(E_MovingTowards(O,'RI',T1))
           AND MinDistance(Infantry,'RI',Y,T1)
Alert Y
    Span:
     Consumed_by_Parent = no; Consume_Parent = no;
     ON E_Distance(O,'RI',2,T2)
     IF CameraInfo(C,U,V,W) AND U = 'Persons' AND V = 'Carry' and W = 'Bag'
                    AND T2 >= T1 AND MinDistance(Patrol,C,Z,T2)
     Alert (Z,C)
```

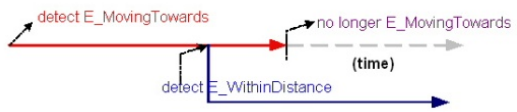

Figure 2: $(ECA)^2$ Trigger for **Req1**

G1: Develop high-level linguistic constructs for declarative specification of $(ECA)^2$ triggers and incorporate them on top of the TinySQL framework, along with procedures that will automatically translate these specifications into logic programs in the spirit of SNlog.

As an illustration of the need for context-awareness, recall that that due to the processing of an already-existing query ***Q1***, the routing of the notification (regarding the detection of the *moving_towards* event) to the sink, has to bypass the sensors in the region of relevance for ***Q1***. In the light of observation (2) above, a straightforward interpretation of **Req1**, as specified in the example-trigger in Figure 2, has an obvious drawback. Namely, the evaluation of the *condition* may be literally interpreted as if *individual*

instantaneous queries about the status of the Marines units, are transmitted anytime a detection of the primitive *(location,time)* update (re)confirms the composite event *continuosly_moving_towards*. However, this is undesirable from the perspective of wasting communication resources – an effect present in any distributed environment, but the negative effects are amplified in sensor networks settings. To cope with this, in [7] we introduced the concept of the *Meta-Trigger*, which is a module that takes a locally-specified $(ECA)^2$ trigger and, after parsing it, decomposes it into *local components* and *distributed components* that are installed into the remote sites, thus promoting *push-like* behavior instead of the *pull* based one. In this spirit, another goal that we are currently pursuing is:

G2: Develop local (in-node) triggers that will:

- Consider the load of the individual node and respond to events that represent new request for that particular node in a manner that will balance the QoD requirements of the new request with the existing ones;
- Collaborate with the Meta-Trigger module in a manner that will take into account the minimization of the communication overheads , while generating the distributed components of the locally-specified triggers;
- Be aware of their "local-ignorance", in the sense of recognizing that a particular request is not part of their pre-programmed list of tasks and, instead of dropping the packets with such request, raise exception-notifications to the respective sinks.

Lastly, as an umbrella spanning through this particular research project, we have the general goal:

G3: Develop novel constructs for specifying composite event-predicates for which the constituent primitive events are correlated to non-local spatial and spatio-temporal entities, along with distributed algorithms for their efficient processing.

References

1. S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein , W. Hong, S. Krishnamurthy, S. Madden, F. Reiss and M. A. Shah, "TelegraphCQ: Continuous Dataflow Processing", S*IGMOD Conference, 2003.*
2. W. Fung, D. Sun and J. Gehrke, "COUGAR: the network is the database", *SIGMOD Conference*, 2002.
3. J. V. Harrison and S. W. Dietrich, "Integrating Active and Deductive Rules", *RiDS Workshop, 1993.* pp. 288-305, 1993.
4. B. T. Loo, T. Condie, M. N. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe and I. Stoica, "Declarative networking: language, execution and optimization", *SIGMOD Conference*, 2006.
5. N. W. Paton, *Active Rules in Database Systems*, 1999.
6. G. Trajcevski, C. Baral and J. Lobo, "Formalizing and Reasoning About the Requirements Specifications of Workflow Systems", *Int. J. Cooperative Inf. Syst.*, Vol. 10, No. 4, 2001.
7. G. Trajcevski, P. Scheuermann, O. Ghica, A. Hinze and A. Voisard, "Evolving Triggers for Dynamic Environments", *EDBT*, 2006.
8. J.D. Ullman, "Bottom-Up Beats Top-Down for Datalog", *PODS*,1989.
9. M. Welsh and G. Mainland, "Programming Sensor Networks Using Abstract Regions", *NSDI*, 2004.
10. J. Widom, "Deductive and Active Databases: Two Paradigms or Ends of a Spectrum?", *RiDS Workshop*, 1993.