Event Broker Grids with Filtering, Aggregation, and Correlation for Wireless Sensor Data

Eiko Yoneki

University of Cambridge Computer Laboratory,
Cambridge CB3 0FD, United Kingdom
eiko.yoneki@cl.cam.ac.uk

Abstract. A significant increase in real world event monitoring capability with wireless sensor networks brought a new challenge to ubiquitous computing. To manage high volume and faulty sensor data, it requires more sophisticated event filtering, aggregation and correlation over time and space in heterogeneous network environments. Event management will be a multi-step operation from event sources to final subscribers, combining information collected by wireless devices into higher-level information or knowledge. At the same time, the subscriber's interest has to be efficiently propagated to event sources. We describe an event broker grid approach based on service-oriented architecture to deal with this evolution, focusing on the coordination of event filtering, aggregation and correlation function residing in event broker grids. An experimental prototype in the simulation environment with Active BAT system is presented.

1 Introduction

Recent progress in ubiquitous computing with a dramatic increase of event monitoring capabilities by Wireless Sensor Networks (WSNs) is significant. Sensors can detect atomic pieces of information, and the data gathered from different devices produce information that has never been obtained before. Combining regionally sensed data from different locations may spawn further useful information. An important issue is to filter, correlate, and manage the sensed data at the right time and place when they flow over heterogeneous network environments. Thus, an integrated event correlation service over time and space is crucial in such environments.

Event correlation services are becoming important for constructing reactive distributed applications. It takes place as part of applications, event notification services or workflow coordinators. In event-based middleware systems such as event broker grids, an event correlation service allows consumers to subscribe to patterns of events. This provides an additional dimension of data management, improvement of scalability and performance in distributed systems. Particularly in wireless networks, it helps to simplify the application logic and to reduce its complexity by middleware services. It is not easy to provide reliable and useful data among the massive information from WSNs. Mining new information from sensed data is one issue, while propagating queries over WSNs is a different issue. Combination of both approaches will enhance data quality, including users' intentions such as receiving, providing, and passing data. At the same time, data should be managed as openly as possible.

Middleware's Task: Middleware for sensor networks can be defined as software that provides data aggregation and management mechanisms, adapting to the target application's need, where data are collected from sensor networks. This functionality must be well integrated within the scheme of ubiquitous computing. The middleware should offer an open platform for users to seamlessly utilize various resources in physically interacting environments, unlike the traditional closed network setting for specific applications. As a part of this approach, mobile devices will play an important role and will be used for collecting sensor data over ad hoc networks, conveying it to Internet backbone nodes. Mobile devices can be deployed in remote locations without a network infrastructure, but they are more resource constrained, and a detectable/implementable event detection mechanism is required.

The trend of system architecture to support such platforms is towards service broker grids based on service management. When designing middleware for sensor networks, heterogeneity of information over global distributed systems must be considered. The sensed information by the devices is aggregated and combined into higher-level information or knowledge and may be used as context. The publish/subscribe paradigm becomes powerful in such environments. For example, a publisher broker node can act as a gateway from a WSN, performing data aggregation and distributing filtered data to other networks based on contents. Event broker nodes that offer data aggregation services can efficiently coordinate data flow. Especially with the distributed event-based middleware over peer-to-peer (P2P) overlay network environments, the construction of event broker grids will extend the seamless messaging capability over scalable heterogeneous network environments. Event Correlation will be a multi-step operation from event sources to final subscribers, combining information collected by wireless devices into higher-level information or knowledge.

There has been much effort for in-network data aggregation such as TinyDB [8]. However, a mainstream of deployments of sensor networks is to collect all the data from the sensor networks and to store them in database. Data analysis is preceded from the data in the database. We propose a distributed middleware architecture integrating global systems to support high volume sensor data. We prototype our proposed system in a simulation environment with real world data produced by the Active BAT system [5].

This paper continues as follows: section 2 describes middleware architecture, section 3 discusses event filtering/aggregation/correlation, section 4 reports an experimental prototype with the Active BAT system, section 5 describes related works and it concludes with section 6.

2 Middleware Architecture

Service Oriented Architecture (SOA) is a well-proven concept for distributed computing environments. It decomposes applications, data, and middleware into reusable services that can be flexibly combined in a loosely coupled manner. SOA maintains agents that act as software services performing well-defined operations. This paradigm allows users to focus on the operational description of the service. All services have an addressable interface and communication via standard protocols and data formats (i.e., messages). SOA can deal with aspects of heterogeneity, mobility and adaptation, and offers seamless integration of wired and wireless environments. Generic service elements are context model, trust and privacy, mobile data management, configuration, service discovery, event notification. The following are key issues addressed in our design.

- Support for service discovery mechanisms (e.g., new and sporadical services) for ad hoc networks.
- Support for an adaptive abstract communication model (i.e., event-based communication for asynchronous communication).

Peer-to-peer networks and grids offer promising paradigms for developing efficient distributed systems and applications. We integrate the Open Services Gateway Initiative (OSGi) [10] on the application layer. OSGi is open to almost any protocol, transport or device layers. The three key aspects of the OSGi mission are multiple services, wide area networks, and local networks and devices. Key benefits of the OSGi are platform and application independent. In other words, the OSGi specifies an open, independent technology, which can link diverse devices in local home network. The central component of the OSGi specification effort is the services gateway. The services gateway enables, consolidates, and manages voice, data, Internet, and multimedia communications to and from the home, office and other locations. We have developed a generic reference architecture applicable to any ubiquitous computing space. The middleware contains separate physical, sensor components, event-broker, service, service management, and an open application interface. We are in progress of implementing the reference architecture.

2.1 Service Semantics

We define service semantics in addition to the service definition so that services can be coordinated. Model real world is a collection of objects, where objects maintain their state using sensor data. Queries and subscriptions are examples of objects that are mapped to the service objects, and thus mapped to the sensors. This approach gives flexibility to services that will develop and evolve.

2.2 Layer Functionality

We describe the brief functionality of each layer below (see also Fig. 1).

Physical Layer: This layer consists of various sensors and actuators.

Sensor Component Layer: A sensor component layer can communicate with a wide variety of devices, sensors, actuators, and gateways and represent them

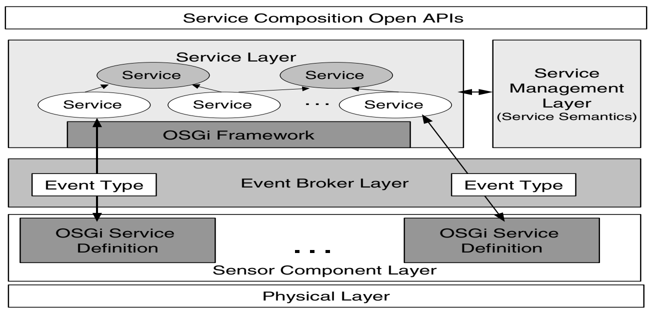

Fig. 1. Middleware Architecture with Wireless Sensor Data

to the rest of the middleware in a uniform way. A sensor component converts any sensor or actuator in the physical layer to a service that can be programmed or composed into other services. Users can thus define services without having to understand the physical world. Decoupling sensors and actuators from sensor platforms ensures openness and makes it possible to introduce new technology as it becomes available.

Event Broker Layer: This layer is a communication layer between Sensor component layer and Service layer. It supports asynchronous communication by publish/subscribe paradigm. Event filtering, aggregation, and the correlation service is currently a part of this layer.

Service Layer: This layer contains the Open Services Gateway Initiative (OSGi) framework, which maintains leases of activated services. Basic services represent the physical world through sensor platforms, which store service bundle definitions for any sensor or actuator represented in the OSGi framework. A sensor component registers itself with the service layer by sending its OSGi service definition. Application developers create composite services by Service Management Layer's functions to search existing services and using other services to compose new OSGi services. Canned services, which may be usefully globally, could create a standard library.

A context is represented as an OSGi service composition, where the context could be obtained. The context engine is responsible for detecting and managing states.

Service Management Layer: This layer contains ontology of the various services offered and the appliances and devices connected to the system. Service advertisement and discovery use service definitions and semantics to register or discover a service. Service definitions are tightly related to the event types used for communication in Event Broker Layer including composite formats. The reasoning engine determines whether certain composite services are available.

Application Interface: An application interface provides open interfaces for applications to manage services including managing contexts.

3 Event Correlation/Filtering/Aggregation

Event Correlation will be essential when the data is produced in a WSN and multi-step operation from event sources to final subscribers, which may reside in the Internet. Combined data collected by wireless devices into higher-level information or knowledge must be supported. In [12], we introduced a generic composite event semantics, which combines traditional event composition and a generic concept of data aggregation in wireless sensor networks. The main focus is on supporting time and space related issues such as temporal ordering, duplication handling, and interval-based semantics, especially for wireless network environments. We presented event correlation semantics defining precise and complex temporal relationships among correlated events.

Event correlation is sometimes deployed as part of applications, event notification services, or as a framework as part of middleware. Definition and detection of composite events especially over distributed environments vary, and equally named event composition operators do not necessarily have the same semantics, while similar semantics might be expressed using different operators. Moreover, the exact semantic description of these operators is rarely explained. Thus, we define the following schema to classify existing operators: *conjunction, disjunction, sequence, concurrency, negation, iteration, and selection.* Considering the analyzed systems, it becomes clear that to simply consider the operators is not sufficient to convey the full semantic meaning. Each system offers parameters, which further define/change the operator's semantics. The problem is that the majority of the system reflects parameters within the implementations.

Many event-based middleware offer a content-based filtering, which allows subscribers to use flexible querying languages to declare their interests with respect to event contents. The query can apply to different event types. On the other hand, the event correlation addresses the relation among event instances of different event types. Filtering and correlation share many properties. WSN has led to new issues to be addressed in event correlation. Traditional networking research has approached data aggregation as an application specific technique that can be used to reduce the network traffic.

WSN Data aggregation in-network operation has brought a new paradigm to summarize current sensor values in some or all of a sensor network. TinyDB [8] is an inquiry processing system for the sensor network and takes a data centric approach, where each node keeps the data, and those nodes execute retrieval and aggregation (in-network aggregation) with on-demand based operation to deliver the data to external applications. TinyLIME [3] is enhancing LIME (Linda In Mobile Environments) to operate on TinyOS. In TinyLIME, tuple space subdivided as well as LIME is maintained on each sensor node, and coordinated tuple space is formed when connecting with the base station within one hop. It works as middleware by offering this abstracted interface to the application. The current form of TinyLIME does not provide any data aggregation function, and only a data filtering function based on Linda/LIME operation is provided at the base station node. On the other hand, TinyDB supports data aggregation function via SQL query, but redundancy/duplication handling is not clear from available

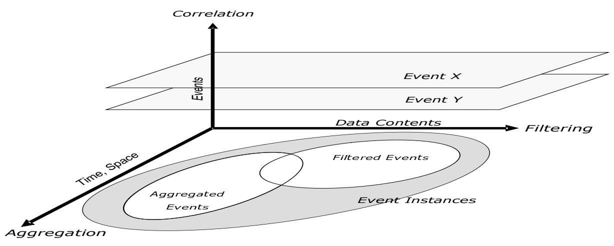

Fig. 2. Event Filtering, Aggregation and Correlation

documents. The coordination of nodes within WSN is different from other wireless ad hoc networks. The group of nodes acts as a single unit of processors in many cases, and for a single phenomenon, multiple events may be produced to avoid the loss of event instances, which is a different concept from traditional duplication of events. Aggregation has three stages: local, neighbour, and global. Fig. 2 highlights the relation among aggregation, filtering and correlation. Middleware research for WSN has been recently active, but most research focuses on in-network operation for specific applications. We provide a global view of event correlation over whole distributed systems from the correlation semantics point.

4 Prototype with Active BAT System

Sentient computing is a type of ubiquitous computing which uses sensors to perceive its environment and react accordingly. Sensors are used to construct a world model, which allows location-aware or context-aware applications. One research prototype of a sentient computing system has been the work at AT&T Research in the 1990s and the research continues at the University of Cambridge as the Active BAT system [5]. It is a low-power, wireless indoor location system accurate up to 3 cm. It uses an ultrasound time-of-light trilateration1 technique to provide more accurate physical positioning. Users and objects carry Active BAT tags. In response to a request that the controller sends via short-range radio, a BAT emits an ultrasonic pulse to a grid of ceiling-mounted receivers. At the same time the controller sends the radio frequency request packet, it also sends a synchronized reset signal to the ceiling sensors using a wired serial network. Each ceiling sensor measures the time interval from reset to ultrasonic pulse arrival and computes its distance from the BAT. The local controller then forwards the distance measurements to a central controller, which performs the trilateration computation. Statistical pruning eliminates erroneous sensor measurements caused by a ceiling sensor detecting a reflected ultrasound pulse, instead of one that travelled along the direct path from the BAT to the sensor.

SPIRIT (SPatially Indexed Resource Identification and Tracking) [5] provides a platform for maintaining spatial context based on raw location information derived from the Active BAT location system. It uses CORBA to access information and spatial indexing to deliver high-level events such as 'Alice has entered the kitchen' to listening context aware applications. SPIRIT models the physical world in a bottom up manner, translating absolute location events for objects into relative location events, associating a set of spaces with a given object and calculating containment and overlap relationships among such spaces, by means of a scalable spatial indexing algorithm.

4.1 Prototype

The current Active BAT system employs a centralized architecture, and all data are gathered in the database, where computational power is cheap. The Active BAT system, as described, is expensive to implement in that it requires large installations, and a centralized structure. The centralized structure allows for easy computation and implementation, since all distance estimates can be quickly delegated to a place where computational power is cheap. Moreover, the active mobile architecture facilitates the collection of multiple simultaneous distance samples at the fixed nodes, which can produce more accurate position estimates relative to a passive mobile architecture.

It is inherently scalable both in terms of sensor data acquisition and management as well as software components. However, when constructing real-time mobile ad hoc communications with resource constrained devices, a distributed coordination must be supported, so that mobile device users can promptly subscribe to certain information.

We simulate all rooms and corridors hold gateway nodes (see the location map Fig. 3), which are capable to participate in event broker grids. The software design follows the service-oriented architecture described in Section 2. Thus, each local gateway node performs event filtering and correlation. Each local node registers the service that associates states with abstractions such as ”ID10 in the room SN07”. These states are decomposed to the units executable by the event broker grid, where event correlation and aggregation and filtering are operated. The details of high-level language for service composition and event type definition are still under development. The used data is taken on March 22nd in 2005. The total number of events is around 200,000, and a sample of event data is shown in Fig. 4. This shows BAT data after the location of the user is calculated, which consists of timestamp, user, area, coordination (X, Y, Z) and orientation. The receiver on the ceiling produces more than two times of data than this.

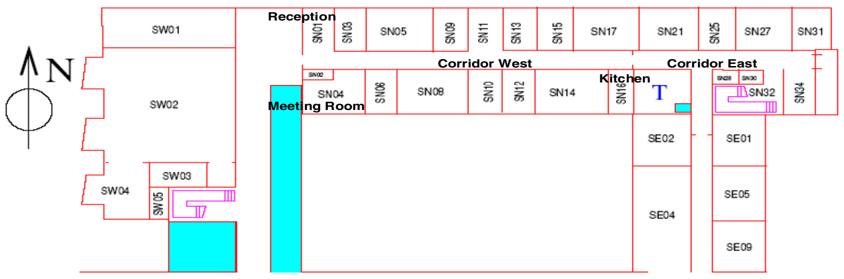

Fig. 3. Active BAT Location Map

```
30408.802618,10,SN09,1002.335327,1033.320801,1.261441,-22.443605
30408.856115,10,SN09,1002.520386,1033.289429,1.251856,-20.335649
30409.063099,10,SN09,1002.533203,1033.279297,1.285185,-20.326197
30409.112594,10,SN09,1002.732910,1033.234863,1.270585,-22.712467
30409.315079,10,SN09,1002.921448,1033.175903,1.271525,-54.598316
30409.370575,10,SN09,1002.994690,1033.126587,1.283121,-56.499645
30409.564561,10,SN09,1003.170227,1033.044556,1.298443,-52.581676
```

Fig. 4. Active BAT Events

4.2 Experiments

We performed several event correlations, and among those, we show the use of durable events below. Fig. 5 depicts the number of events over the local gateway nodes without durable events and Fig. 6 shows the same operation with durable event composition. During this experiment, thirteen BAT holders participated, which are shown ID1 through ID13. The result shows a dramatic reduction of event occurrences through the use of durable events.

Fig. 7 and Fig. 8 also depict the power of durable events composition over user ID 10 and 13 over the timeline (24 hours).

4.3 Temporal Ordering in Active BAT System

The applications derived from Active BAT have high accuracy and real-time tracking requirements. Problems of time synchronization and coordination amongst beacons are easily resolved, because these systems are wired and have a centralized controller. In the Active BAT system, the timestamp is derived from a Global Clock Generator (GCG), which is a hardware clock that sends 'ticks' to every component of the system over a serial link. When a location is computed, the location message is timestamped using the GCG. In general, GCG delay is in the order of microseconds, and the slowest part of the system is the bit that waits for ultrasound to propagate (speed of sound) after a position is requested but before a position can be calculated. This delay is used to measure

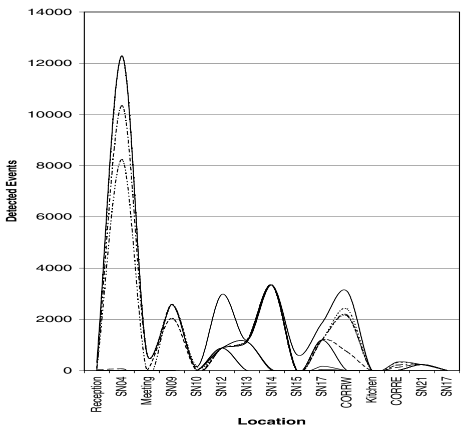

Fig. 5. All Sensed Events

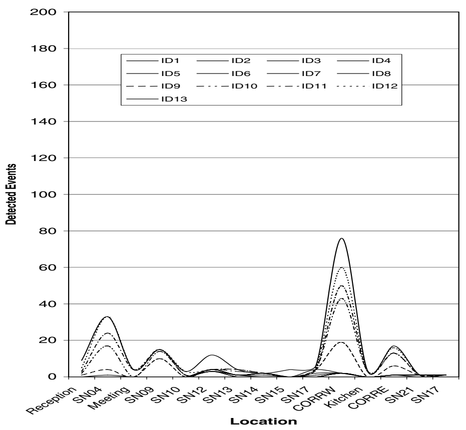

Fig. 6. Durable Events

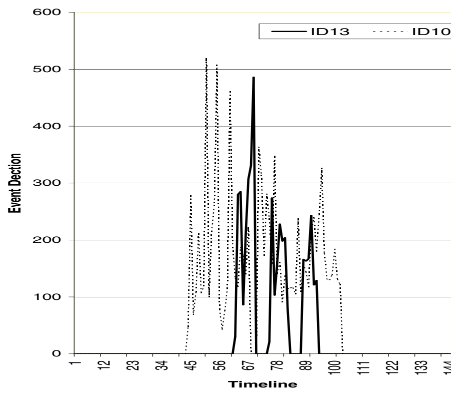

Fig. 7. Events of ID13 and ID10

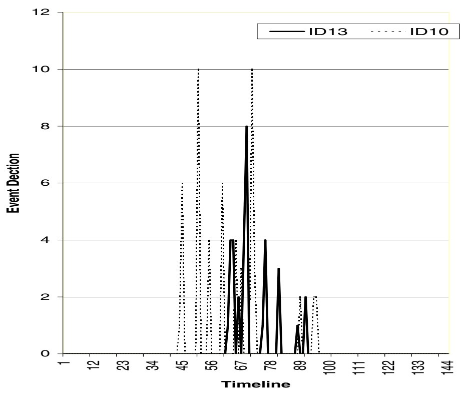

Fig. 8. Durable Events of ID13 and ID10

the distance between the BAT and the receiver at the ceiling. Once the location is calculated, the message then has to travel up to SPIRIT (order of milliseconds), and the event will be generated. However, no reliable information on that delay is considered. In Active BAT system, time synchronization is controlled in a centralized manner, and a trigger to collect BAT information is also used to synchronize all the clocks in the system. The current experiment assumes that all timestamps are properly synchronized. The implementation of temporal ordering mechanism described in [12] is in progress.

5 Related Works

Much composite event detection work has been done in active database research. SAMOS [4] uses Petri nets, in which event occurrences are associated with a number of parameter value pairs. Early language for composite event follows the Event-Condition-Action (ECA) model and resembles database query algebras with an expressive syntax. Snoop [2] is an event specification language for active databases, which informally defines event contexts. The transition from a centralized to a distributed system let to a new challenge to deal with time. Snoop presents an event-based model for specifying timing constraints to be monitored and to process both asynchronous and synchronous monitoring of real-time constraints. Reference [7] proposes an approach that uses occurrence time of various event instances for time constraint specification. GEM [9] allows additional conditions, including timing constraints to combine with event operators for composite event specification. In event-based middleware, publish/subscribe can provide subscription for the composite event instead of multiple primitive events, and this reduces the communication within the system and potentially gives a higher overall efficiency, which is addressed by [11]. Hayton et al. [6] on composite events in the Cambridge Event Architecture [1] describe an object-oriented system with an event algebra that is implemented by nested pushdown FSA to handle parameterized events. For related work on WSN data aggregation, see Section 3.

6 Conclusions and Future Work

The recent evolution of ubiquitous computing with a dramatic increase of event monitoring capabilities by wireless devices and sensors requires sophisticated middleware. We focus on specific aspects for supporting service broker grids including the data and communication models. The network environments will be more heterogeneous than ever, and open, P2P based networking environments will become common. In this paper, we provide a prototype of such a grid system over sensor networks and show simulated experimental results based on real data from the Active BAT system. Event broker nodes that offer data aggregation services can efficiently coordinate data flow. Event broker grids should seamlessly disseminate relevant information generated by deeply embedded controllers to all interested entities in the global network, regardless of specific network characteristics, leading to a solution to construct large distributed systems over dynamic hybrid network environments. We are working on a complete implementation including various timestamping environments and parallel/hierarchical composition.

Acknowledgment. This research is funded by EPSRC (Engineering and Physical Sciences Research Council) under grant GR/557303.

References

1. Bacon, J. et al. Generic Support for Distributed Applications. *IEEE Computer*, 68-77, 2000.
2. Chakravarthy, S. et al. Snoop: An expressive event specification language for active databases. *Data Knowledge Engineering*, 14(1), 1996.
3. Curino, C. et al. TinyLIME: Bridging Mobile and Sensor Networks through Middleware. *Proc. PerCom*, 2005.
4. Gatziu, S. et al. Detecting Composite Events in Active Database Systems Using Petri Nets. *Proc. RIDE-AIDS*, 1994.
5. Harter, A. et al. The Anatomy of a Context-Aware Application. *Proc. MobiCom*, 1999.
6. Hayton, R. et al. OASIS: An Open architecture for Secure Inter-working Services. *PhD thesis, Univ.of Cambridge*, 1996.
7. Liu, G. et al. A Unified Approach for Specifying Timing Constraints and Composite Events in Active Real-Time Database Systems. *Proc. IReal-Time Technology and Applications Symposium*, 1998.
8. Madden, S. et al. TAG: A tiny aggregation service for ad-hoc sensor networks. *Proc. of Operating Systems Design and Implementation*, 2002.
9. Mansouri-Samani, M. et al. GEM: A Generalized Event Monitoring Language for Distributed systems. *IEE/IOP/BCS Distributed systems Engineering Journal*, 4(2), 1997.
10. OSGi, http://www.osgi.org.
11. Pietzuch, P. Shand, B. and Bacon, J. Composite Event Detection as a Generic Middleware Extension. *IEEE Network Magazine, Special Issue on Middleware Technologies for Future Communication Networks*, 2004.
12. Yoneki, E. and Bacon, J. Unified Semantics of Event Correlation over Time and Space in Hybrid Network Environments. *Proc. CoopIS*, 2005.