The Third SIGMOD PhD Workshop on Innovative Database Research
(IDAR2009), June 28, 2009, Providence, USA

Complex Event Detection in Pervasive Computing

Chunjie Zhou, Xiaofeng Meng
School of Information, Renmin University of China, Beijing, China
{lucyzcj, xfmeng} @ ruc.edu.cn

ABSTRACT

In pervasive computing environments, wide deployment of sensor devices has generated an unprecedented volume of atomic events. However, most applications such as healthcare, surveillance and facility management, as well as environmental monitoring require such events to be filtered and correlated for complex event detection. Therefore how to extract interesting, useful and complex events from low-level atomic events is becoming more and more important in daily life. Due to the increasing importance of complex event detection, this paper proposes a framework of Complex Event Detection and Operation (CEDO) in pervasive computing. It gives an event model and extends current detection by incorporating temporal and spatial settings of events and different levels of granularity for event representation. We first show research issues, related works, and main research problems in this area. Then our current research works and the preliminary results are introduced. Finally, the research plan of my PhD project is presented for discussion.

Keywords

pervasive computing, atomic events, complex events, complex event detection

1. INTRODUCTION

In pervasive computing environments, sensors are deployed in everything from IT networks to enterprise software systems and physical world devices (through RFID readers, bar code scanners, manufacturing equipment sensors, and others). As these systems continue to proliferate, they generate events at a growing rate. Usually there are thousands of data records in a normal sensor device, which make it difficult for operators to find exceptional events by checking every record. While operators should find the relative records timely when analyze the exceptional events afterwards. However, the traditional detection methods are short of intelligent analysis and the data records are unable to be indexed efficiently. People must search artificially according to the rough time interval, so the data analysis waste a lot of time and energy.

In order to solve these problems, the efficient method is to do intelligent analysis on events atomically, and extract the centralized and interesting events timely. So it can give an alarm in time and index data efficiently based on the stored event information. For a concrete example, in a retail store, a occurrences and non-occurrences of events, and impose temporal constraints over these events. When there is a scenario where an item was picked up from a shelf and then taken out of the store without being checked out, the system could give an alarm atomically. Today, significant improvements in operational business decisions await those organizations to capture and process these events into meaningful business insight.

At present many applications need to extract complex events (often user-specified) from these flows of low-level atomic events. Such applications include supply chain management, financial services [1], business activity monitoring, elder care [2], and various pervasive computing applications. In it, the applications of indoor environment include: 1) checking "Whether the patient has already been taken care of" which contains a series of checks "Did the patient take his medicine?" "Did he have his lunch?" "Was his symptom normal or not?" and so on. 2) The security system might decide whether to take some precautionary action by comparing the complex events at the same time of different days. The applications of outdoor environment include: in the airport, station, or district, it can be used to detect and follow the people, the vehicles, or other suspicious objects. It also can be used to judge whether there are exceptional actions of people or vehicles in the restricted area.

All the above applications require such events to be filtered and correlated for complex event detection and transformed to new events that reach a semantic level appropriate for end applications [3]. These requirements need to perform real time translation of data describing a physical world into information useful to end applications. So this paper proposes a framework of Complex Event Detection and Operation (CEDO), which provides a rich, declarative environment for the development of event processing applications that may process and act on thousands of events per second. CEDO will be integrated into standard middleware architectures and be embedded in any standard enterprise application. It can be deployed as a stand-alone offering on third party application severs or as an integrated service engine. The main contribution of the framework is to address modeling, representation, and detection of events, where the focus is to detect about events rather than about the changes in objects' states.

The remainder of the paper is organized as follows. Section 2 introduces the related works. The event model and our framework of CEDO are shown in section 3. Section 4 presents the research plan of my PhD project.

Proceedings of the Third SIGMOD PhD Workshop on Innovative Database Research (IDAR 2009), June 28, 2009, Providence, RI, USA

2. RELATED WORKS

In pervasive computing environments, events distribute in nodes scattered, some of which are mobile nodes. If we use a central node to detect atomic events and form complex event expressions, this node will become the bottleneck of event detection. Therefore, in order to detect events effectively, we should choose the appropriate detection method according to characteristics of pervasive computing environments and system requirements. The current existing complex event detection methods include: (1) based on the event tree [5] complex event detection; (2) based on the diagram detection method [6]; (3) based on automata [7] complex event detection; (4) based on Petri nets [4] complex event detection; (5) pipeline operation [8] detection methods. The above complex event detection methods all have their own advantages and weak points: GEM [5] considers the delay between events occurrence and detection, and handles events disorder by assigning the biggest tolerant delay. But it assumes there is a perfect global synchronous clock, which is unsuitable for no-centralized management and distributed systems of clock drift and loose coupling. Due to the lack of consideration of unpredictable delay, it cannot make breaking and mobile detection in mobile database efficiently. Snoop [6] only provides the simple time model, in which every event is regarded as a certain time point. Atomic events are based on definitions, while complex events are based on semantic. This method is suitable for centralized system or LAN. ODE [7] uses Finite Automata to express events, which can express real-world events intuitively, establish automata and detect complex events. But pure automata can neither detect parameter-events nor express event-disorder, so it cannot meet requirements of distributed systems. In addition, the above complex event detection methods don't consider uncertainty at all, which is the essential characteristic of pervasive computing environments.

As mentioned above, the existing complex event detection methods all cannot satisfy the requirements of pervasive computing environments. Therefore, based on the characteristics of pervasive computing environments, we summarize the current research works mainly from the three characteristics of complex event detection. Current researches on complex event detection generally include the following aspects: from the angle of event-type, describing the representation of complex events; from the angle of time, describing all kinds of sequential representations; from the precision degree of data, analyzing and handling the probabilistic data. Current research works emphasize particularly on different aspects.

According to the above three characteristics of complex event detection, the existing research works can be classified and summarized as in figure 1. In it, three axes correspond to three characteristics: time (time point and time interval), data (precise and uncertain), and events (atomic and complex). Three axes divide the space into eight quadrants (as shown in figure 1), and each quadrant corresponds to different attribute values of the three characteristics.

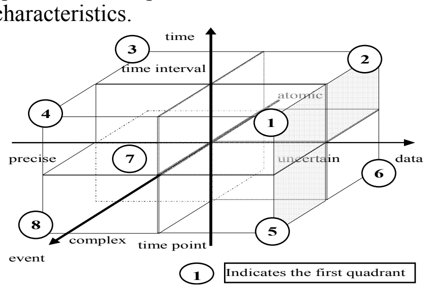

Figure 1 the summary of current research works

As in figure 1, the cube region in quadrant 7 stands for researches about precise atomic events at time point. At present, there are many research works about it [9]. The cube regions in quadrant 3, 4, 6 and 8 indicate separately the precise atomic events in time interval, the precise complex events in time interval, the uncertain atomic events at time point, and the precise complex events at time point. There are some research works about them [10, 8, 11, 12, 13, 14 and 20]. The cube regions in quadrant 1, 2, 5 are mainly about uncertain data, including the uncertain atomic events at time point, the uncertain complex events in time interval, and the uncertain complex events at time point. As far as we know, there is merely related research works about them. As shown in table 1, in quadrants 3, 4, 6 and 8, [9] is about time point, atomic events and precise data; [10, 8] is about time point, complex events and uncertain data; [11, 12 and 19] is about time interval, atomic events and precise data; [13] is about time interval, complex events and precise data; [14] is about uncertain data.

Table 1 the comparison of current research works

Research works	Time interval	Complex events	Uncertain data
[9]	No	No	No
[10,8]	No	Yes	No
[11, 12, 19]	Yes	No	No
[13]	Yes	Yes	No
[14]	No	No	Yes

3. FRAMEWORK

In this section, we present the framework of CEDO, and illustrate how this framework can be used to support application requirements. Before that, we give some preliminaries and an event model, which are the basis of the following.

3.1 Preliminaries

Events are defined as something that users are interested in. Events are happening all around us all the time. Detection of a person in a room, the firing of a CPU timer, and a Denial of Service (DoS) attack in a network are example events from various application domains. All events signify certain activities; however their complexities can be significantly different. For instance, the firing of a timer is instantaneous and simple to detect, whereas the detection of a DoS attack is an involved process that requires computation over many simpler events. So events can be divided into two types: atomic events and complex events [10]. In the following we will give definitions of them.

Definition 1 (atomic event): An atomic event is defined as a thing that happens instantaneously at a specific time point. It can be expressed as $E_{atomic_i} = Action < o_i, p_i, t_i >$. In it, o_i stands for certain object; p_i indicates some place, which is the current location of object o_i; t_i expresses a certain time point; *Action* means the activity of object o_i at time t_i in place p_i. An atomic event E_{atomic_i} corresponds to something in the physical world. For example, *Coffee ('Mary', 'Room 301', 10:00am)* is an atomic event, which means "Mary is getting coffee in Room 301 at 10:00am".

Definition 2 (complex event): A complex event often happens in a continuous time interval, which is assigned by users (called case 1) or abstract directly from atomic events (called case 2). It can be expressed as $E_{complex_i} =< Q_i, E_i, T_i >$. In it Q_i stands for a certain query, which is only useful in case 1 and be used to indicate the query condition, while in case 2, its value is null; E_i expresses the set of atomic events, which is $E_i = \{1 \le i \le n \mid E_{atomic_i}\}$. The atomic events in the set are connected and some relational operators exist among them (such as positive correlation, negative correlation, parallelism, serial, and so on); T_i means a time interval. Querying or abstracting on a series of correlated atomic events in a time interval T_i is the process of getting complex events. For example, "Mary is getting coffee" can be extracted from a series of atomic events "Mary is in her office", "Mary is in coffee room", "Mary is in her office", and so on. Complex events are generated by composing atomic or other complex events using a set of event detection operators.

3.2 Event Model

In this paper, we consider events in the context of spatio-temporal databases. As introduced in section 3.1, our model includes atomic events and complex events. Here we use a model like data cube which is a three- (or higher) dimensional array of values. The three dimensions are separately object, time and place. As in figure 2, a snapshot of the model taken at time *i* contains all objects' current positions. For simplicity, we call each such snapshot a *world* W and it can be expressed as $W = \{1 \le i \le n \,|< o_i, p_i >\}$ (the meaning of o_i and p_i can be found in section 3.1). A *stream* shows the same place where different people are in at different time. A *flow* of objects is a set of places in which the object is at distinct timestamps. An *event database* consists of several flows of objects in the time interval T. We call each such event database a complex event, and denote it as a sequence of sets of tuples: $E_{complex} = (E_1, \cdots, E_n, T_i)$ where $E_i = \{1 \le i \le n \,|\, E_{atomic_i}\}$ (see section 3.1). The start time, the end time, and the duration of complex events can be showed in the time dimension. In addition, the choice of granularity for time dimension is very important. When the granularity of time is too small (e.g., milliseconds), an event query such as "What did he do during the first three days of May, 2008?" lead to a massive amount of uncertain (3×24×60×60×1000) possibilities.

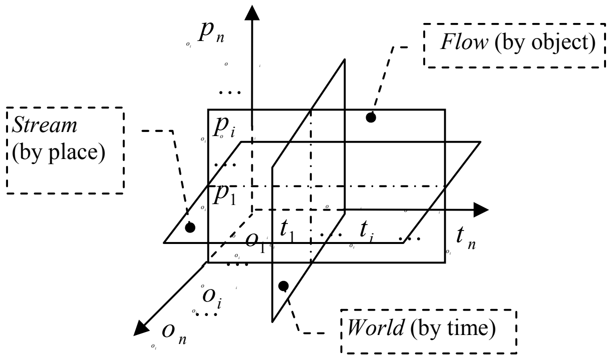

Figure 2 the event model

3.3 Complex Event Detection and Operation

Our framework has two sub-processes: complex event detection and complex event operation, which are the inverse procedures (as shown in figure 3). In the following, we will introduce every block of our framework.

The terminal layer

This is the source of raw data, including mobile devices, the smart phone, PDA, PC, and so on. Every created data has two elements: the data itself and a timestamp [15].

The application layer

The applications include healthcare, security monitoring, people tracker, and so on. These applications need high-level complex events oriented to clients.

Smart device bus

According to the two sub-processes, the smart device bus has two functions. In the process of complex events detection, the smart device bus is in charge of feeding raw data to CEDO; while in the process of complex events operation, it translates the message into the commands that can be recognized by a physical reader.

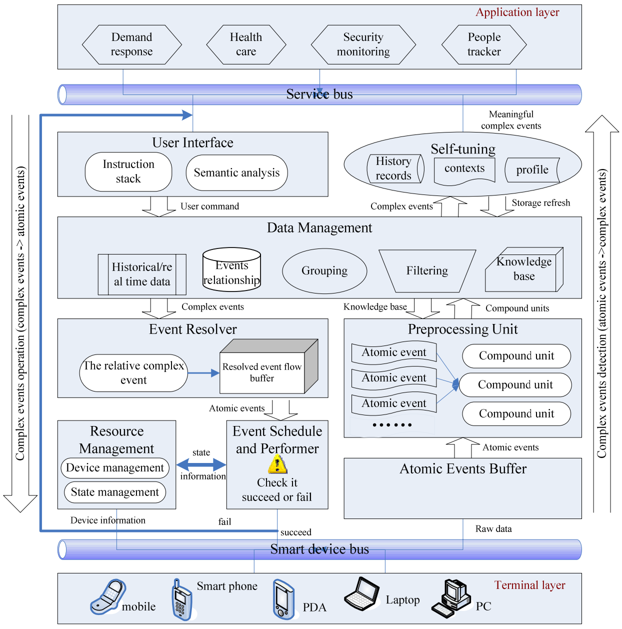

Figure 3 the framework of CEDO

3.3.1 Complex event detection

The goal of complex event detection is to enable information contained in the events flowing through all of the layers of the framework to be discovered, understood in terms of its impact on high-level management goals and business processes, and acted upon in real time (as shown in the right side of figure 3). Here we consider as an example a database that handles typical behaviors of occupants in a smart home. Table 2 shows raw data collected from physical devices.

Table 2 raw data stream (RDS)

RID	Obj	Time	Place	Probability
1	O1	6:40:00am	Kitchen	1.0
2	O1	6:40:01am	Bedroom	1.0
3	O1	6:50:00am	Kitchen	1.0
4	O1	7:00:20am	Kitchen	1.0
5	O1	7:10:00am	Dining-hall	1.0
6	O1	7:20:35am	Dining-hall	1.0

Ps: RID (Record ID), Obj (Object).

Atomic events buffer

Many types of applications generate data streams as opposed to data sets. Managing and processing data for these types of applications involves building buffer storage and forming atomic events with a strong temporal focus. Atomic events outputted from the buffer have context, that is, timing (when it happened, both in absolute terms and relative to other events), sequence, and linking relationships to other events.

In the above example, the raw data are inputted to "atomic events buffer" through "smart device bus". After a certain time interval, some atomic events are outputted from "atomic event buffer", which are shown in Table 3.

Table 3 atomic events (AE)

RID	Obj	Time	Place	Probability
1	O1	6:40:00am	Kitchen	1.0

2	O1	6:40:01am	Bedroom	1.0
3	O1	6:40:02am~ 7:09:59am	Kitchen	< 1.0
4	O1	7:10:00am~ 7:29:59am	Dining-hall	< 1.0

Preprocessing unit

CEDO looks at events in the context of other events rather than in isolation. So the preprocessing unit classifies large volume of atomic events into compound units, according to the incorporating temporal and spatial settings of the incoming events. In order to get meaningful compound units, a pattern matching capability is typically included in the preprocessing unit.

The above atomic events are inputted into "preprocessing unit" intermittently, and be classified into compound unit. This process often needs some extra information which is stored in "database management". In this example, the probability of record 2 is reduced based on the rules of spatial information (shown in table 6). O1 may be cooking or washing in the kitchen (based on the knowledge base in table 5), so we integrate the two atomic events into one compound unit. Because O1 went to the dining-room after that, we guess O1 was more likely cooking in the kitchen and the probability of record 1 is set 0.7 (as shown in table 4).

Table 4 compound units (CU)

CU	RID	Obj	Time duration	Place	Action	Probability
1	1	O1	6:40:00am~ 7:09:59am	Kitchen	cooking	0.75
	2	O1	7:10:00am~ 7:29:59am	Dining hall	eating	
2	3	O1	6:40:00am~ 7:09:59am	Kitchen	washing	0.24
	4	O1	7:10:00am~ 7:29:59am	Dining hall	eating	

Data management

The data management infrastructure of CEDO supports the notion of streams of structured data records together with stored relations. Many modern applications require long-running queries over continuous unbounded sets of data [18]. So there are two kinds of event records stored in the database. One is called "real-time event record", which is the processing of events as they arrive; the other is called "historical event record", which is the use of a sophisticated and optimized storage mechanism. CEDO is really an integration of historical event record coupled with real-time event record. A knowledge base also should be stored in the data management, which includes the extra information, such as the spatial location information, and the possible actions in certain place. Relations identify the relationships between incoming atomic events in CEDO. In CEDO the instantaneous relation is used to denote a relation in the traditional bag-of-tuples sense, and relation to denote a time-varying bag of tuples.

As in the above example, both data records and relationships are stored in the "database management" as shown in table 6. In addition, there should be a knowledge base used to store extra information (shown in table 5).

Table 5 the knowledge base in database management

Place	Time interval	Action	Probability
Kitchen	6:00am-7:30am or 11:00am-12:30am or 18:00pm-19:30pm	cooking	0.75
		washing	0.25
Kitchen	7:30am-8:00am or 12:30am-13:00pm or 19:30pm-20:30pm	cooking	0.75
		washing	0.25

Table 6 rules in database management

Place1	Place2	Distance	Cost time
Kitchen	Bedroom	20 meters	15 seconds

Table 7 complex events chain (CEC)

CE Identity/Name	Sequence	Atomic Event	Probability
Dinner	1	Cooking	0.75
	2	Eating	
	3	Washing	

Self-tuning

Filtering the complex events outputted from the data management according to the people' profile, the context, the historical records and so on. So the complex events reported to end users must be meaningful. The final results of self-tuning are also stored in the database.

In the above example, through "self-tuning", complex events in table 7 will become more and more precise, which are useful to end applications. The complex events chain can be shown intuitively in figure 4.

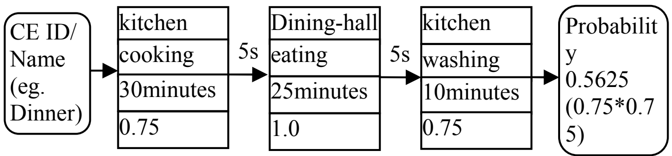

Figure 4 the complex event chain

3.3.2 Complex event operation

The goal of complex event operation is to resolve complex events into a set of corresponding atomic events, which can be recognized and performed by physical devices through semantic analysis (as shown in the left side of figure 3).

User interface

CEDO provides a graphical user interface (GUI). It differs from traditional graphical user interface technologies in that they are designed to display and manipulate time-based information typically found in event processing systems.

Event resolver

Resolve the complex event into a set of relative atomic events, which are stored in a buffer temporarily. This is the inverse process of complex event detection.

Resource management

It includes device management and state management. Device management schedules physical devices and decides which device should be used. State management defines state situations of physical devices. Each physical device has four statuses: undefined, unrequest, request and active. "Undefined" means this kind of physical device does not exist; "unrequest" indicates there is such physical device but no event requests it; "request" implies there is at least one event that wants to use this physical device; "active" signifies this device is being used now.

Event schedule and performer

Decide which one in the resolved atomic events buffer should be performed first and check whether the performance conditions of the relative event are satisfied. If all relative physical devices are available, then the event performance succeeds and sends instructions to corresponding devices. Otherwise, it fails and returns the failing reason to the user.

4. RESEARCH WORKS

In my PhD project, I plan to focus on a few key problems in

CEDO. To make our work based on a strong foundation, we would like to fully implement CEDO and build the experimental platform of complex event detection in pervasive computing based on our framework. This section will introduce our main work on event probability, event disorder, and event relationship in CEDO.

1) Event probability

Uncertainty is one of the most important challenges of complex event detection (such as RFID data). However, there are many reasons for producing probabilistic data. For example: 1) conflicting readings, e.g. Alice is read by two adjacent antennas, what is her true location? [17]; 2) missed readings, e.g. readers commonly detect only about 60%-70% of tags in their vicinity [17]; 3) granularity mismatch, an application queries about offices, but the system only provides information about sensors.

Complex event detection on probabilistic data can be divided into two types: local uncertainty detection and global uncertainty detection. When event detection only concern with the uncertainty of the tuple / object itself, and are independent from other objects/ tuples, we call them local uncertainty detection. Let's think the example of getting coffee in section 3.1, when and where does Mary want to get coffee, and the time duration of getting coffee are all uncertain. As shown in table 8 (a), the time when Mary gets coffee may be at 10:15 am or at 9:55am; the place where she gets coffee may be in No.1 cafe or in No.2 café; the time duration may be 15 minutes or 17 minutes. But these factors are only based on Mary's own willing, and independent of others', so it is called local uncertainty.

On the other hand, when the event detection must consider the uncertainty of combinations of objects / tuples, we call such detection global event detection. We still take getting coffee as an example. Suppose that Mary likes to get coffee together with Joe, then the time, place and duration of getting coffee are not only based on Mary's own willing, but be decided by many uncertain factors, such as Joe's willing and the actions of others. As shown in table 8(b), in July, due to the influence of Joe, the time that Mary got coffee is earlier and the duration are shorter than in March. But their coffee time and duration in different dates are still uncertain. Generally, when whether an object / tuple satisfies a detection condition depends on other objects or tuples not involved in the same generation rule, global uncertainty has to be considered. Semantically, we have to examine the possible worlds one by one and count the probability that a combination of objects / tuples is an answer.

Table 8(a) the event local uncertainty

Date	Name	Coffee time	Café ID	Time duration
March 1st	Mary	10:15am	1	15 minutes
March 2nd	Mary	9:55am	2	17 minutes
March 1st	Joe	8:55am	1	10 minutes
March 2nd	Joe	9:05am	1	8 minutes

Table 8(b) the event global uncertainty

Date	Name	Coffee time	Café ID	Time duration
July 1st	Mary	9:35am	1	12 minutes
July 2nd	Mary	9:25am	1	10 minutes
July 1st	Joe	9:35am	1	12 minutes
July 2nd	Joe	9:25am	1	10 minutes

Most of the current researches on complex event detection suppose events are precise, however, they are imprecise in many real applications. Probability is the essential problem in pervasive computing, even in complex event detection. For example, in the application scenario of "smart home", the sensor data, the behavior pattern and customers of the occupants are all probabilistic. So how to extract meaningful and precise information according to these imprecise data is a challenge problem. Probability has become a hot research problem in resent years, but there are still many problems should be further researched in probabilistic complex event detection, such as the probabilistic sensor data, the local uncertainty detection and global uncertainty detection. With the appearance of large volume of probabilistic events, the probabilistic complex event detection will become more and more important and demand prompt solution.

2) Event disorder

The tuples in an event flow may or may not be in order by some desired attribute of those tuples. When such an ordering exists, some operations become easier and can be performed without the need for arbitrary storage; however, when this ordering is violated, this is called "event disorder." Poset processing consists of performing operations on a set of tuples that may not be related by a total ordering. Any partially ordered set of tuples can be processed in arbitrary ways within an event flow processing system by storing those tuples and retrieving as needed to match desired patterns. Most of current researches suppose events are ordering, that is to say, they don't consider the concurrent and overlapping events. However, in many real applications this assumption is unacceptable. Take the healthcare in section 1 for example; the atomic events (such as toothbrushing and taking temperature) may happen currently in the process of detecting complex events (eg. healthcare). In addition, atomic events and complex events in this process are possible disorder because of different habits of different people. Future research on complex event detection must take disorder events into account.

3) Event relationship

Current researches on complex event detection usually suppose events are isolate, but actually they have a thousand and one links. So in complex event detection, we must consider the relationships of the same object in different times, the interaction of different objects, and the factors of identity, position, and so on. Here we use an example in [14] to explain. The location of Joe at T=7 and T=8 are separately shown in Figure 5(a) and 5(b). [14] use sampled distributions produced by the particle filter to express the location probability. Each particle represents a guess about Joe's location and the locations are uncertain. Figure 5(c) shows the location of Joe and Sue are connected, that is to say, we can guess approximately the location of Sue according to Joe's location. In figure 5(c) the probabilities of Joe in H1 and in O2 are both 0.4 at T=7. If we know Sue is the secretary of Joe and they are almost together. We can guess Joe was more likely to be in O2 at T=7 based on the probability that Sue was in O2 at T=7 is 0.6. Figure 5(d) shows the location of Joe at T=7 and at T=8 are also connected. If Joe was in O2 at T=7, he was more likely to be still in O2 at T=8. However, current researches don't consider these connective factors.

In addition, most current research works only consider converting atomic events to complex events, few studies convert complex events to more complex ones. The input of the latter is the output results of the former, so the former research is an important step of more complex event detection. However, with wide applications in real world, more complex event detectionwill become increasingly important. Take the health-care in section 1 as an example, checking "Whether the patient has already been taken care of" contains a series of checks "Did the patient take his medicine?" "Did he have his lunch?" "Was his symptom normal

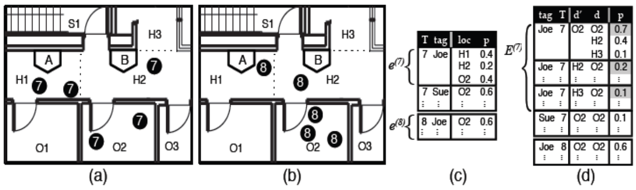

Figure 5 the relationships of events

or not?" and so on. In this example, we can regard the whole process of health-care as a more complex event. The actions involved in it can be atomic events, or complex events. For example, checking "Did the patient take his medicine?" is a complex event, because it includes the following atomic events: "pick up a cup of water ", "take up the medicine bottle", and "take water". While checking "whether the reading of blood pressure and body temperature is normal or not" are atomic events. We use figure 6 to express the ranked events intuitively.

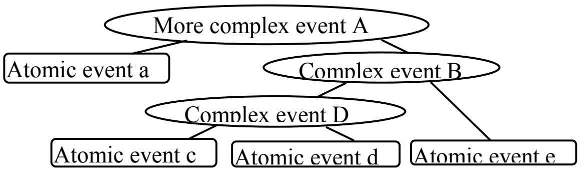

Figure 6 the ranked complex events

The specific solution approaches for the various challenges listed are our future research works, and maybe we will consider tree-based complex events.

5. CONCLUSION

This paper presents the sketch of my research plan on PhD project. Firstly, we summarize the current research status in this area. Then our framework of complex event detection and operation in pervasive computing is introduced. It gives an event model and extends current detection by incorporating temporal and spatial settings of events and different levels of granularity for event representation. Based on this framework, the unprecedented volume of atomic events can be filtered and correlated to get interesting, useful and complex events. In conclusion, the main works of my PhD project are to build a complex event detection system and to address several key issues in this area.

6. ACKNOWLEDGMENTS

This research was partially supported by the grants from the Natural Science Foundation of China (No: 60833005, 60573091); China 863 High-Tech Program (No: 2007AA01Z155); PhD Foundation Program of Education Ministry (No: 200800020002).

7. REFERENCES

[1] Demers, A. J., Gehrke, J., and Hong, M. 2006. Towards expressive publish/subscribe systems. In EDBT, pages 627-644.

[2] Liao, L., Patterson, D. J., Fox, D., and Kautz, H. A. 2007. Learning and inferring transportation routines. Artif. Intell, 171(5-6):311-331.

[3] Khoussainova, N., Balazinska, M., Suciu, D. 2008. Probabilistic Event Extraction from RFID Data. Proceedings of IEEE 24th International Conference on Data Engineering (ICDE 2008). 1480-1482

[4] Gatziu, S., Dittrich, K. R. 1993. Events in an active object-oriented database system. In Proceeding of the 1st Int'l Conference on Rules in Database Systems. 23-39

[5] Samani, M., Sloman, M., Gen, M. 1997. A generalized event monitoring language for distributed systems. IEEE / IOP/BCS Distributed Systems Engineering. 96-108

[6] Chakravarthy, S., Rasad, V. K., and Anwar, E. 1993. Anatomy of a composite event detector. Technical Report UF-CIS-TR-93-039[R]. Gainesville: University of Florida

[7] Gehani, N. H., Jagadish, H. V., Shmueli, O. 1992. Event specification in an active object oriented database. Proceedings .of the ACM SIGMOD International Conference on Management of Data. ACM Press. 81-90

[8] Wu, E., Diao, Y., Rizvi, S. 2006. High-performance complex event processing over streams. In SIGMOD06: Proceedings of the 2006 ACM SIGMOD. 407-418

[9] Agrawal, J., Diao, Y., Gyllstrom, D., and Immerman, N. 2008. Efficient Pattern Matching over Event Streams. International Conference on Management of Data Proceedings of the 2008 ACM SIGMOD international conference on Management of data. Vancouver, Canada. 147-160

[10] Akdere, M., Cetintemel, U., and Tatbul, N. 2008. Plan-based Complex Event Detection across Distributed Sources. Proceedings of the VLDB Endowment.Vol.1,Issue 1: 66-77

[11] Papapetrou, P., Kollios, G., Sclaroff, S., and Gunopulos, D. 2005. Discovering Frequent Arrangements of Temporal Intervals. ICDM Proceedings of the Fifth IEEE International Conference on Data Mining IEEE. 354 - 361

[12] Wu, S. Y., and Chen, Y. L. 2007. Mining Nonambiguous Temporal Patterns for Interval-based events. IEEE Transactions on Knowledge and Data Engineering. Volume 19, Issue 6: 742-758

[13] Patel, D., Hsu, W., and Lee, M. L. 2008. Mining Relationships Among Interval-based Events for Classification. Proceedings of International Conference on Management of Data Proceedings of the 2008 ACM SIGMOD international conference on Management of data. 393-404

[14] Re, C., Letchner, J., Balazinksa, M., and Suciu, D. 2008. Event Queries on Correlated Probabilistic Streams. Proceedings of International Conference on Management of Data Proceedings of the 2008 ACM SIGMOD international conference on Management of data. 715-728

[15] Complex Event Processing in the Real World, an Oracle White Paper, September 2007

[16] Romero, M., and Rodriguez, M. A. 2007. A Graph-Oriented Model and Query Language for Events. SeCoGIS 2007-International Workshop on Semantic and Conceptual Issues in Geographic Information Systems. 358-367

[17] Jeffery, S. 2006. Adaptive cleaning for RFID data streams. In Proc. of the 32nd VLDB Conference.

[18] Chen, Q., Li, Z. H., Liu, H. L. 2008. Optimizing Complex Event Processing over RFID Data Streams. Proceedings of IEEE 24th International Conference on Data Engineering (ICDE 2008). 1442-1444

[19] Kam, P. S., Ada, W. F. 2000. Discovering Temporal Patterns for Interval-based events. Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery.