OntoEvent: An Ontology-Based Event Description Language for Semantic Complex Event Processing

Meng Ma[1(✉)], Ping Wang[2,4], Jun Yang[3], and Chao Li[4]

[1] School of EECS, Peking University, Beijing 100871, China
mameng@pku.edu.cn
[2] National Engineering Research Center for Software Engineering,
Peking University, Beijing 100871, China
pwang@pku.edu.cn
[3] School of Electronic and Computer Engineering, Peking University, Shenzhen 518055, China
yangjvn@pku.edu.cn
[4] School of Software and Microelectronics, Peking University, Beijing 102600, China
li.chao@pku.edu.cn

Abstract. In this paper, we propose an ontology-based event description language, OntoEvent, for semantic event modeling in CEP system. In OntoEvent language, complex events are modeled and described in the form of event ontology. We propose the concept of nature language event constructor and define them by synonym set of WordNet database to describe logical and temporal event relationship in OntoEvent. We demonstrate event ontology examples in application domain of smart home, and our analysis shows that OntoEvent is of rich expressiveness compared to other related languages.

Keywords: Complex event processing · Ontology · Event description language · Semantic · WordNet · Natural language processing

1 Introduction

Complex event processing (CEP) [1] technique is one of the important cornerstones to connect cyber and physical world. Semantic event modeling and processing helps CEP systems to establish a transparent, machine-understandable knowledge discovering process. Some typical semantic complex event processing (SCEP) systems such as SCEPter [2], OECEP [3] and DyKnow [4] are proposed. However, a fundamental problem for SCEP is that no general-purpose semantic model for event construction has been proposed. In this paper, we propose an ontology-based event description language, OntoEvent, for semantic event model in CEP systems. In OntoEvent semantic model, we divide the concepts into two levels: general concepts and domain-specific instances, which promises the OntoEvent can be dynamic extended for different application usage. Complex events are modeled and described in the form of event ontology based on nature language event constructors. Our analysis and preliminary implementation proves that OntoEvent is of rich expressiveness and interoperability between different event description languages. The rest of this paper is organized as follows: Section 2 presents

J. Li and Y. Sun (Eds.): WAIM 2015, LNCS 9098, pp. 448–451, 2015.
DOI: 10.1007/978-3-319-21042-1_38

the semantic models and elaborates the event description language. We demonstrate an event example and discuss the language expressiveness in Section 3. Section 4 concludes the paper and discusses our future works.

2 OntoEvent Language

The semantic model of OntoEvent language defines and describe the concepts and their inter-relationship in complex event processing. It is a two-level framework which includes a set of general upper ontology and interfaces for different application domain ontology extensions. All the concepts in OntoEvent semantic model are grouped into four domains: *Entity*, *Dimension*, *Activity*, and *Service*. The semantic model of OntoEvent is established and implemented in OWL [5] ontology languages. Based on this semantic model, we present a novel ontology-based event description language, OntoEvent. Different from other event description language which we have surveyed in Section 2, OntoEvent language defines complex event as an ontology by its event component and their relationship. The component of a complex event includes primitive event, event constructor, event pattern, attribute, sliding window size and response action. The OntoEvent language is defined as follow:

OntoEvent Language Model. The OntoEvent language model is denoted as $E = (e, c, p, a)$, in which e is the event set, o is the event constructor set, $p = \{hasComponent, hasAttribute, hasSource, hasData, hasWindow, hasAction\}$ is the ontology property set and a stands for the attribute/data set. The pattern, primitive event source, attribute constraints, event window and response action of a certain complex event is defined by the following triples:

Pattern. $\subseteq \{e \cup c\} \times \{hasComponent\} \times \{e \cup c\}$, defines how the event is constructed by primitive events and event constructors.

Source. $\subseteq e \times \{hasSource\} \times a$, defines the primitive event source and indicates their corresponding attributes.

Constraints. $\subseteq \{e \cup c\} \times \{hasData, hasAttribute\} \times a$, defines the attribute constraints of primitive of event constructor.

Window. $\subseteq e \times \{hasWindow\} \times a$, defines the overall processing time limitation on the input event stream.

Action. $\subseteq e \times \{hasAction\} \times a$, defines the response action for the complex event when it is detected.

In OntoEvent language, we introduce the notion of nature language event constructor to express the logic and temporal relationship in event pattern. These constructors are the "keywords" for pattern description in natural English language. In order to support synonym keyword in event description, we introduce WordNet [6], which is a lexical semantic database groups English words into sets of synonyms. With the help of WordNet, we can avoid repetition of similar keyword definitions. Besides, natural or semi-natural description from end-users can be processed by natural language processing (NLP) and mapped into their equivalent constructors. In Table 1, we present and explain some typical representative constructors and their equivalent WordNet synset.

Table 1. Typical Nature Language Event Constructors

Representative Constructor	Equivalent Synset in WordNet		Constructor Semantics
	Type	Synonymy	
and	(*adv*)	*meanwhile, meantime, in the meantime*	Conjunction of two event/constructor components.
or	(*adv*)	*either*	Disjunction of two event/constructor components.
not	(*adv*)	*no*	Negation of event/constructor component.
then	(*adv*)	*subsequently, later, afterwards, after*	Sequence of two event/constructor components, representing the two event occurs sequentially.
distance	(*n*)	*distance, space*	Describing the temporal distance of two event/constructor components.
within	(*adv*)	*within, in*	Describing that event/constructor component occurs within the certain time interval.
keep	(*v*)	*retain, continue, keep, keep on*	Describing that one or more event/constructor component occurs and no counter instance occurs within a certain time interval.

3 Application Scenario and Event Example

We illustrate our proposed event description language using event examples in smart home application scenario. Smart home refers to a home environments that are enabled for co-operation of smart objects and systems for ubiquitous interactions [7]. In this environment, we leverage devices such as location sensor, body sensor and other sensing devices to generate primitive event streams.

Natural Language Event Description. This pattern set the status of user as *sleeping* if user enters the bedroom first and **then** heartbeat rates **keep** lower than 70 **within** 30 seconds, while the temporal **distance** between these two events is [600, 1200].

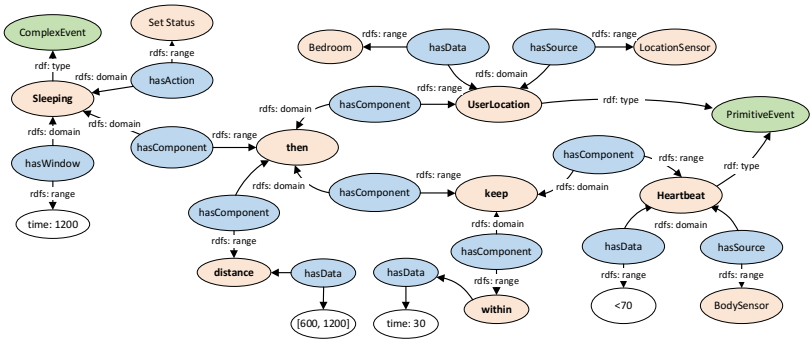

Fig. 1. Event Ontology Example

The event ontology for this event example is depicted in Figure 1. This event describes a temporal-constrained event sequence and iteration consists of two primitive event *UserLocation* and *HeartBeat.* The source of primitive event are declared by attribute *hasSource*. The event constraints are defined by *hasData*. To describe this complex event *Sleeping*, we leverages the constructor *then*, *keep*, *within* and *distance*

to depict the event logic. Specifically, we use constructor *distance* to describe the temporal distance between two event components of constructor *then*. We use *keep* to describe the event repetition of *HeartBeat* when the reading is lower than 70 and declared the time scope of 30 seconds by *within* constructor.

OntoEvent has rich expressiveness in logical and temporal aspects. Especially, it provides a set of synonym constructor to express the event iteration or status maintenance. The expressiveness of OntoEvent can be extended by defining more event constructors. Because of the essence of ontology, the event ontology can be defined and demonstrated in a user-friendly visualized paradigm. Therefore, as an ontology-based language, it is designed to be a ***middle-language*** to provide interoperability among different event processing systems by language rewriting techniques. Natural language event descriptions can be tranformed into event ontology by NLP. Besides, existing non-semantic event descriptions in existing CEP systems can be transformed into OntoEvent language by event rewriting.

4 Future Works

In our other works, an ontology-based event processing engine have been preliminarily implemented based on OntoEvent language. In this event processing model, we transform event ontology into non-deterministic finite automata-based detection model. A multi-target detection model is established according to the inference relationship between events which is derived from defined event ontologies. In the future work, we plan to establish a natural event query processing approach with the help of NLP, to map natural language query into equivalent event ontology. By this means, ontology-based CEP system can provide a user-friendly interface in complex event querying.

References

1. Cugola, G., Margara, A.: Processing flows of information: From data stream to complex event processing. ACM Computing Surveys (CSUR) **44**, 15 (2012)
2. Zhou, Q., Simmhan, Y., Prasanna, V.: SCEPter: Semantic complex event processing over end-to-end data flows. Technical Report 12-926, Computer Science Department, University of Southern California (2012)
3. Binnewies, S., Stantic, B.: OECEP: enriching complex event processing with domain knowledge from ontologies. In: Proceedings of the Fifth Balkan Conference in Informatics, pp. 20–25. ACM, Novi Sad (2012)
4. De Leng, D., Heintz, F.: Towards on-demand semantic event processing for stream reasoning. In: 17th International Conference on Information Fusion (FUSION), pp. 1–8 (2014)
5. McGuinness, D.L., Van Harmelen, F.: OWL web ontology language overview. W3C recommendation, February 10, 2004
6. Miller, G.A.: WordNet: a lexical database for English. Communications of the ACM **38**, 39–41 (1995)
7. Helal, S., Mann, W., El-Zabadani, H., King, J., Kaddoura, Y., Jansen, E.: The gator tech smart house: A programmable pervasive space. Computer **38**, 50–60 (2005)