Spatio-Temporal Event Model for Cyber-Physical Systems

Ying Tan, Mehmet C. Vuran, Steve Goddard
Computer Science and Engineering
University of Nebraska–Lincoln, Lincoln, NE 66588
{*yingtan, mcvuran, goddard*}*@cse.unl.edu*

Abstract

The emerging Cyber-Physical Systems (CPSs) are envisioned to integrate computation, communication and control with the physical world. Therefore, CPS requires close-interactions between the cyber and physical worlds both in time and space. These interactions are usually governed by events*, which occur in the physical world and should autonomously be reflected in the cyber-world, and* actions*, which are taken by the CPS as a result of detection of events and certain decision mechanisms. Both event detection and action decision operations should be performed accurately and timely to guarantee temporal and spatial correctness. This calls for a flexible architecture and task representation framework to analyze CP operations. In this paper, we explore the temporal and spatial properties of events, define a novel CPS architecture, and develop a layered spatio-temporal event model for CPS. The event is represented as a function of attribute-based, temporal, and spatial event conditions. Moreover, logical operators are used to combine different types of event conditions to capture composite events. To the best of our knowledge, this is the first event model that captures the heterogeneous characteristics of CPS for formal temporal and spatial analysis.*

1. Introduction

Cyber-Physical Systems (CPSs) are integrations of computation, communication, and control with the physical world [1]. More specifically, a CPS is envisioned to be a heterogeneous *system of systems*, which consists of computing devices and embedded systems including distributed sensors and actuators. These components are inter-connected together in a large-scale and execute autonomous tasks to link the cyber world and the physical world. Generally, these tasks involve close-interactions between the two worlds and a certain change in one world should be reflected in the other world in a time-sensitive and/or spatial-sensitive manner. On the other hand, CPS applications and users may not be interested in every change in the physical world. Instead, certain conditions are of interest, according to which certain predefined operations are executed by the CPS. In our framework, we refer to the conditions of interest as *events* and the desired predefined operations following the detection of an event as *actions*. As a result, any CPS task can be represented as an "*Event-Action*" relation.

Developing an event model for CPSs is a challenge due to several reasons: First, the abstraction of an event is highly dependent on the component, whereas the CPS components and their architecture have not been formally defined. For example, the abstraction of an event *"user A is nearby window B for the last 30 minutes"* by a sensor mote can be defined as the range measurement of the user A according to window B. On the other hand, the abstraction of the same event by a sink node can be the location of user A because the sink node may have received several range measurements from different sensor motes and the user location can be calculated. Second, it is clear that the event specification mechanism for CPSs must also support spatio-temporal events. More specially, a CPS event might be defined as a combination of attributes, temporal, and spatial information. Consequently, spatial properties should be incorporated into event definitions in addition to temporal and attribute-based properties. To best of our knowledge, no prior attempts have been done for CPSs yet. Finally, since the abstraction of the same event by different CPS components may be different, the event model must be flexible to integrate different events over time and space while keeping the information regarding the original physical event intact. Consequently, a common *frame-of-reference* to the heterogeneous entities in CPSs can be provided.

Based on these observations, in this paper, we introduce a *spatio-temporal event model* to capture the close interactions between the physical and cyber worlds in CPSs. More specifically, a hierarchical CPS architecture and the hardware components are defined. Accordingly, the event model relies on a hierarchical layered structure, which extends the event spatio-temporal relations to capture the complex relationships in CPSs. Accordingly, formal temporal and spatial analysis of the CPS can be performed using this generic framework.

The rest of the paper is organized as follows: In Section 2, an overview of the recent work on event modeling in various contexts is provided. In Section 3, the cyber-physical system architecture and its components are described. Accordingly, formal definitions of event types in the physical world and the cyber-world are given in Section 4. Furthermore, the event model for the proposed CPS architecture is described in Section 5. Finally, the paper is concluded and future research topics are highlighted in Section 6.

2. Related Work

The concept of an event has been investigated in different contexts so far. The most related work are from the active database community. The concept of *Event-Condition-Action (ECA)* model is introduced in [2], in which *event* specifies the signal that triggers the evaluation of the *condition* and if true, causes an action to be carried out. In the ECA model, the *event* is generated by a single event source (e.g. system clock or incoming user query) at a point in time. Extensions to the ECA model [3] [4] [5] introduce a set of *event operators* to compose events so that more complex scenarios, namely the composite event, can be described. For example, using *conjunction* event operator represents the logical relation when both events have occurred in any order, while using *sequence* operator represents the temporal relation when one event occurs before the other one. While the above models consider the occurrence time of an event as a time point, SnoopIB [6] considers the occurrence time of an event as a time interval. Accordingly, the event "a light is on for the last 30 minutes" can be defined using time *intervals*. A CPS event model requires support for both temporal event types: punctual and interval events.

Recent advance in run-time verification community tends to adopt an event-based approach to describe the interested properties of a running program [7] [8]. More specifically, events are the observable states of the monitored program during the execution time. Moreover, the interested program properties (e.g. safety or liveness) are the temporal occurrence patterns of the events. For example, Java-MAC [9] [10] adopts the Linear Temporal Logic (LTL) for Java program run-time monitoring, where *events* occur instantaneously during system execution and *conditions* represent information that hold for a duration of time. Compared to our work, there are two major distinctions: First, the events in run-time verification domain are limited within one program and one hardware device, whereas in CPSs, events are generated from multiple programs and heterogeneous components. Second, the temporal relationships for punctual and interval temporal event types are not fully addressed in any of the existing temporal logic. We aim to support both punctual and interval events through our CPS event model.

The real-time community aims to add timing constraints on the top of the ECA model. For example, the Real-time Logic (RTL)-based event model has been proposed with point- and interval-based timing constraints in [11] and [12], respectively. More specifically, the timing constraints in RTL-based event model defines the time point-based temporal relationships among the occurrence time of events. However, since interval-based events are not supported in RTL-based event model, the interval-based temporal relationships such as "$During$, $Overlap$" are not addressed.

Events have also been investigated in the middleware community, where the focus is how event can be used in an asynchronous distributed environment. Unlike the centralized systems, in a distributed environment, it may only be possible to archive partial ordering or restricted total ordering for the events. This poses additional challenges for event-based middleware [13]. On the other hand, in the event (stream) processing community, it is assumed that the events have already been ordered by a third party when received by the event processing device [14]. Moreover, the problem of exploring the received event sequence patterns is investigated and mechanisms are developed to rapidly detect these event patterns [15]. While all these researches are important for the realization of CPSs, however, they are beyond the scope of this paper.

The event models discussed so far only incorporate the temporal properties of events without any spatial properties. The most related work for spatio-temporal analysis are from spatio-temporal database community [16]. Generally, the spatio-temporal database aims to extend the existing Spatial Information Systems (SIS) to include time in order to better describe the dynamic environment. Under the SIS framework, the spatial relationships between different real-life entities have been studied. For example, Egenhofer et.al., proposed the topological relationships of objects in 2-Dimensional space [17]. On the other hand, the role of time in SIS is tracing the lineage of spatial objects and their attributes. Therefore, the temporal relationships are not completely addressed in the context of spatio-temporal database.

In summary, supporting the spatio-temporal event is one basic requirement for a CPS event model. Specially, the punctual and interval events as well as the temporal relationships among them should be considered for completeness. On the other hand, the CPS event model should also capture the heterogeneity in the hardware components in CPSs. To best of our knowledge, none of the existing event models fully meet these requirements.

3. CPS Architecture and Components

As discussed in the Section 1, the development of an event model for CPSs necessitates the definition of a CPS architecture and its components. In this section, we extend our previous work in [18] to a hierarchical CPS architecture, where the following components are considered:

Sensor (SR) and Actuator (AR): Sensors and actuators provide the interface between the physical and cyber worlds. A sensor is a device that measures a physical phenomenon, e.g., room temperature, and converts physical phenomena into information, which contains the attributes, sampling timestamp, and/or spacestamp. In general, one type of sensor

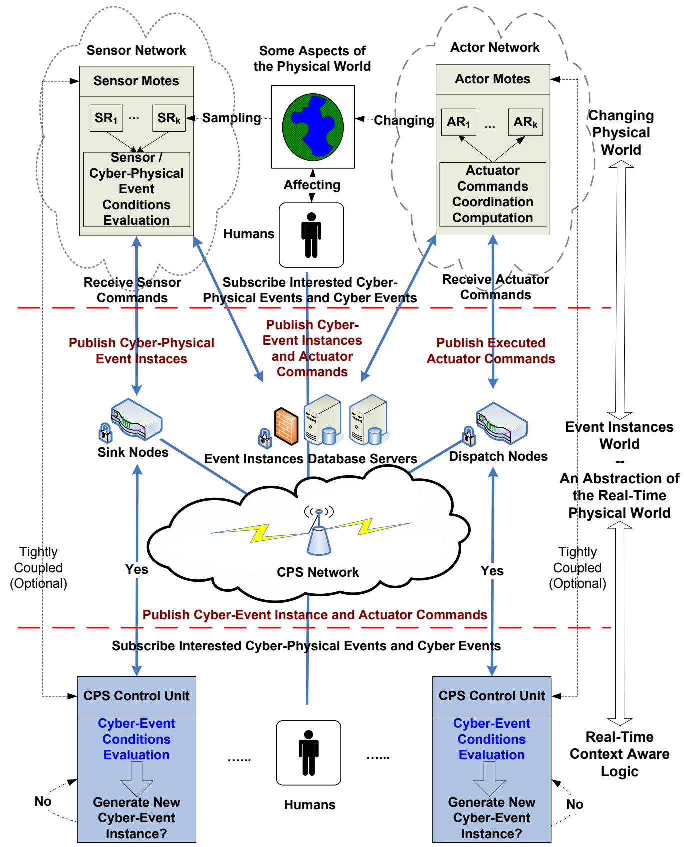

Figure 1. Architecture of Cyber-Physical Systems

is associated with a single physical phenomenon or property. An actuator, on the other hand, is a device that is able to change attributes of a physical object, e.g., move a chair, or physical phenomena.

Sensor and Actor Mote: A sensor (actor) mote usually contains one or more types of sensors (actuators), in addition to a micro controller unit (MCU), and an optional transceiver. Physical observations from each sensor are sent to the sensor mote for further processing and the transceiver is used to send the processed data to other sensor motes [19]. Similarly, action commands sent by CPSs are evaluated to control actuators at the actor mote. Sensor and actor motes can also serve as repeaters to relay and aggregate packets from other motes to form a sensor and actor network [20].

Sink and Dispatch Node: A sink node is a special sensor mote, which receives and aggregates the data received from a set of sensor motes. Similarly, a dispatch node disseminates the action commands to multiple actor nodes. Both nodes serve as a gateway to connect a sensor and actor network to the rest of the CPS network.

CPS Control Unit (CCU): A CCU is an event-driven control unit connected to the CPS network. It receives cyber-physical events from the sink nodes and cyber-events from other CCUs and processes them according to certain rules and generates cyber-events. Moreover, at this level, actions are associated with certain cyber-events.

Database Server: The database server is a distributed data logging service for the event instances. The event instances that circulate inside the CPS network are automatically transferred to the database server after a certain time for later retrieval.

CPS Network: The CPS Network connects the sensor network, actor network, CPS control units, and database servers through wired and/or wireless communication techniques.

4. Spatio-Temporal CPS Event Concept

In this section, we introduce the spatio-temporal CPS event related concepts. In general, the term "event" has been used in two distinct contexts in the literature. The first relates to the physical world occurrences while the second involves representations of those occurrences in a computer system. However, either of the two "event"s can not be transformed from one to another directly. In this section, the general definitions of an *event*, *event condition* and *event instance* are provided. Moreover, the properties of different events and their classifications are discussed. Since temporal and spatial properties of an event are essential in CPSs, we first introduce the time and spatial models next.

Time Model: The time model we use in this paper is similar to the time model used in language Snoop [21]. In a digital system, the notation of time is always discrete and has limited precision. Therefore, time is also considered as discrete collection of time points in our time model.

Spatial Model: The spatial model we use in this paper is a standard 2-dimensional Cartesian coordinate system, in which an ordered pair (x, y) indicates a specific location point and a function $y = f(x)$ indicates a specific location field (polytope).

4.1. Spatio-Temporal Event Definitions

Definition 4.1. Spatio-Temporal Event: is the occurrence of interest, which describes the state of one or more objects either in the cyber-world or the physical world according to attributes, time, and location. We denote a generic event as:

$$\mathcal{E}id \quad \{t^o_{\mathcal{E}id}, l^o_{\mathcal{E}id}, V_{\mathcal{E}id}\} \tag{4.1}$$

where $\mathcal{E}$ is the event type identifier, which denotes the type of the event as discussed in Section 5, id is the event ID, $t^o_{\mathcal{E}id}$ is the event occurrence time, $l^o_{\mathcal{E}id}$ is the event occurrence location, and $V_{\mathcal{E}id}$ is the set of event occurrence attributes. According to the occurrence time, an event can be further classified as a *Punctual Event* or *Interval Event* (refer to Section 4.2), while according to the occurrence location, the event can be either a *Point Event* or a *Field Event* (refer to Section 4.2).

Definition 4.2. Event Condition: Each event is defined as a combination of one or more event conditions, which are constraints in terms of attributes, time, and location. Accordingly, 3 types of event conditions are considered: *attribute-based*, *temporal*, and *spatial* event conditions. In addition, 3 types of operators are used to represent and specify these event conditions: relational operator ($\mathcal{OP_R}$), which defines the attribute-based constraints, temporal operator ($\mathcal{OP_T}$), which defines the temporal event constraints, and spatial operator ($\mathcal{OP_S}$), which defines the spatial event constraints. The details of the event conditions are listed below.

Attribute-based event conditions are defined using *relational operators* $\mathcal{OP_R}$ such as "*Greater*, *Equal*, *Less*". In general, an attribute-based event condition is represented as:

$$g_v[V_1, V_2, ..., V_n] \quad \mathcal{OP_R} \quad C \tag{4.2}$$

where g_v is an aggregation function, e.g., Average, Max, Add, which takes the attribute of n entities, $\mathcal{OP_R}$ is a relational operator, and C is a numerical constant. An entity in CPS can be a physical observation or an event instance. For example, the attributed-based event condition "The average attribute of physical observation x and y is *Greater* than C" can be represented as $Average(V_x, V_y) > C$.

Temporal event conditions are defined according to *temporal operators* $\mathcal{OP_T}$ such as "*Before*, *After*, *During*, *Begin*, *End*". In general, a temporal event condition can be represented as:

$$g_t[t_1, t_2, ..., t_n] \quad \mathcal{OP_T} \quad C_t \tag{4.3}$$

where g_t is an aggregation function which takes the time (occurrence time, estimated occurrence time and so on) of n entities, $\mathcal{OP_T}$ is a temporal operator, and C_t is a time constant (either a point-based or an interval-based time). Similarly, an entity in a CPS can be either a physical observation or an event instance. For example, the temporal event condition "every event instance of event x must occur AFTER 5 time units *Before* event y" can be represented as $t^o_{\boldsymbol{E}(\boldsymbol{CCU}1,\boldsymbol{E}x,i)} + 5 \quad Before \quad t^o_{\boldsymbol{E}(\boldsymbol{CCU}1,\boldsymbol{E}y,i)}$.

Spatial event conditions are defined using *spatial operators* $\mathcal{OP_S}$ such as "*Inside*, *Outside*, *Joint*". In general, a spatial event condition can be represented as:

$$g_s[l_1, l_2, l_3, ...] \quad \mathcal{OP_S} \quad C_s \tag{4.4}$$

where g_s is an aggregation function, which takes the location of n entities, $\mathcal{OP_S}$ is a spatial operator, and C_s is a location constant (either a point or a field). An entity in CPS can be a physical observation or an event instance. For example, the spatial event condition "every event instance of event x must occur *Inside* event y" can be represented as $l^o_{\boldsymbol{E}(\boldsymbol{CCU}1,\boldsymbol{E}x,i)} \quad Inside \quad l^o_{\boldsymbol{E}(\boldsymbol{CCU}1,\boldsymbol{E}y,i)}$.

Using the 3 types of event operators, a composite event condition can be defined using *logical operators* $\mathcal{OP_L}$ such as "*AND*, *OR*, *NOT*". In general, an event condition can be represented as:

$$\left\{ \mathcal{E}id, \begin{array}{cccccccc} (g_v^1 & \mathcal{OP_L} & g_v^2 & ... & \mathcal{OP_L} & g_v^i) & \mathcal{OP_L} \\ (g_t^1 & \mathcal{OP_L} & g_t^2 & ... & \mathcal{OP_L} & g_t^j) & \mathcal{OP_L} \\ (g_s^1 & \mathcal{OP_L} & g_s^2 & ... & \mathcal{OP_L} & g_s^k) & \end{array} \right\} \tag{4.5}$$

where $\mathcal{E}id$ is the event identifier, g_v^i, g_t^j, and g_s^k represent attribute-based, temporal, and spatial event conditions, respectively, and $\mathcal{OP_L}$ is a logical operator. For example, a spatio-temporal sensor event condition $\boldsymbol{S}1$ "every instance of physical observation x occurs before physical observation y and the distance between location of x and the location of y is less than 5 meters (assume x, y are from the sensor motes $\boldsymbol{MT}1$ and $\boldsymbol{MT}2$, respectively)" can be represented as

$$\{\boldsymbol{S}1, (t^o_{\boldsymbol{O}(\boldsymbol{MT}1,\boldsymbol{SR}x,i)} \ Before \ t^o_{\boldsymbol{O}(\boldsymbol{MT}2,\boldsymbol{SR}y,i)}) \quad \wedge \quad (g_{distance}(l^o_{\boldsymbol{O}(\boldsymbol{MT}1,\boldsymbol{SR}x,i)}, l^o_{\boldsymbol{O}(\boldsymbol{MT}2,\boldsymbol{SR}y,i)}) < 5)\}$$

The event conditions are processed by *observers*, which generate event instances that can be shared system-wide. In the following, we define the concept of an *observer*.

Definition 4.3. Observer: An observer is a device or a human that is able to collect data, evaluate these data based on event conditions, and output the according *event instance* if the event conditions are met. For example, a sensor mote is an observer, which can use a sampled data as an input, process this data based on some predefined event conditions, and create an event instance accordingly. On the other hand, a particular sensor on a sensor mote is not an observer although it is capable of measuring a physical quantity and transforming to signals with attribute, time and/or space information. Since it is not capable of processing this captured data based on the event conditions, so it is not considered an observer.

Definition 4.4. Event Instance: The event instance is the result of an evaluation of a certain observer according to event conditions. Accordingly, an event instance is defined as a 3-tuple as follows:

$$\mathcal{E}(OBid, \mathcal{E}id, i) \tag{4.6}$$

where $OBid$ identifies the observer, $\mathcal{E}id$ is the event identifier, and i is the sequence number. Since an observer is associated with each event instance, in addition to the 3 properties of an event, the event instance is characterized by 3 additional properties related to the observer as follows:

$$\{t^g_{\mathcal{E}(OBid,\mathcal{E}id,i)}, l^g_{\mathcal{E}(OBid,\mathcal{E}id,i)}, t^{eo}_{\mathcal{E}(OBid,\mathcal{E}id,i)}, l^{eo}_{\mathcal{E}(OBid,\mathcal{E}id,i)}, V_{\mathcal{E}(OBid,\mathcal{E}id,i)}, \rho_{\mathcal{E}(OBid,\mathcal{E}id,i)}\} \tag{4.7}$$

where $t^g_{\mathcal{E}id(OBid,\mathcal{E}id,i)}$ and $l^g_{\mathcal{E}id(OBid,\mathcal{E}id,i)}$ are the time and location when the observer generates the event instance.

$t^{eo}_{\mathcal{E}id(OBid,\mathcal{E}id,i)}$, $l^{eo}_{\mathcal{E}id(OBid,\mathcal{E}id,i)}$ and $V_{\mathcal{E}id(OBid,\mathcal{E}id,i)}$ are the estimated event occurrence time, location and attribute from the view of the observer. Finally, $\rho_{\mathcal{E}(OBid,\mathcal{E}id,i)}$ is the confidence level of observer regarding the generated event instance.

4.2. Event Classification

According to the temporal and spatial properties, an event can be classified into different classes. In this section we define the concept of temporal event and spatial event as follows:

Temporal Event: The temporal event properties are related to the (estimated) occurrence time of the event. Based on whether this time is a point or an interval in time, the event can be classified into two categories as *Punctual Event* or *Interval Event*.

Punctual Event (E) refers to the case where the occurrence time of an event is a time point. For punctual physical event, it represents any change in attributes, temporal or spatial status of physical objects or phenomena at certain *time point*. Obviously, if an event is considered physically "punctual", the representation of this event in cyber world should also be punctual, i.e. the estimated occurrence time of the cyber event is a time point. For example, event *"user A is nearby window B"* can be considered as a punctual physical event, which we define a nearby window B area and once the user A is detected entering into this area, a punctual cyber event instance is generated.

Interval Event $(\underline{E})$ refers to the case where the occurrence time of an event is a time interval marked by starting and ending time points. For interval physical event, it represents any attributes or spatial status of physical objects or phenomena unchanged for a period of time. For example, event *"user A is nearby window B"* can also be considered as an interval physical event, where the event starts once the user is detected entering into the area and ends once the user is detected leaving this area. Clearly, the difference between the punctual event and the interval event depends on the end-user definition.

Based on these definitions, the temporal relationships between two events can be extended to 3 types: punctual event with punctual event, e.g., *Before, After*, punctual event with interval event, e.g., *During, Meet* and interval event with interval event, e.g., *Overlap*.

Spatial Event: The spatial event properties are related to the (estimated) occurrence location of the event. Based on whether this is a point or a field in location, the event can be classified into two categories as *Point Event* or *Field Event* [22].

Point Event (PE) refers to the case where the occurrence location of an event is a location point (x, y). In physical world, a point event could be a change in the properties of a stationary physical object, e.g., turn on/off a light. Moreover, for an event instance, a point event refers to the case where the estimated occurrence location is a point.

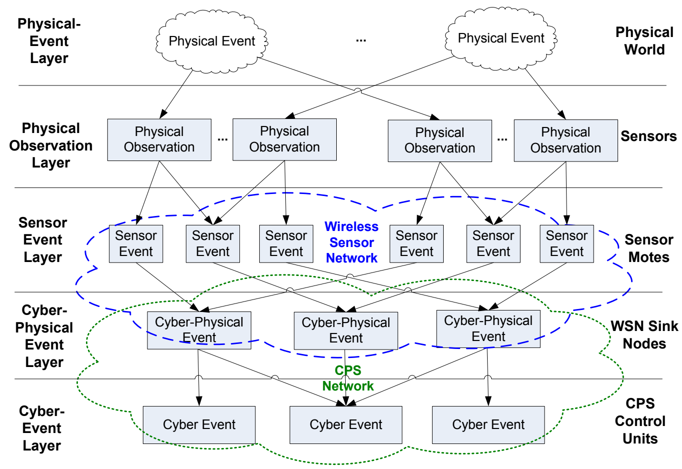

Figure 2. CPS hardware and event model hierarchy

Field Event (FE) refers to the case where the occurrence location of an event is a polytope defined by a function $y = f(x, y)$. In physical world, a field event refers to a physical phenomena, which occurs in an area, e.g., a forest fire or a moving physical object. For an event instance, a field event refers to the case where the estimated occurrence location is an area. Essentially, a field occurrence location is made of at least 2 or more point events.

Based on these definitions, the spatial relationships between two events can be extended to 3 types: point event with point event, e.g., *Equal to*, point event with field event, e.g., *Inside, Outside* and field event with field event, e.g., *Joint*.

5. CPS Event Model

Based on the CPS architecture defined in Section 3 and the discussion on the event concept and its properties in Section 4, in this section, we define the CPS event model. Following the inherent layered structure of the CPS architecture, a layered CPS event model is developed as shown in Figure 2. Accordingly, each aspect of CPS and the associated events can be modeled in a hierarchical manner.

Physical Event: Physical event models the occurrence of the end-user interest in the physical world and can be any change in attribute, temporal or spatial status of one or more physical objects or physical phenomena. A physical event is characterized as follows:

$$\boldsymbol{P}id\,\{t^o_{\boldsymbol{P}id}, l^o_{\boldsymbol{P}id}, V_{\boldsymbol{P}id}\} \qquad (5.1)$$

where $\boldsymbol{P}id$ is the physical event identifier, $\boldsymbol{P}$ denotes a physical event and symbol id is the event identifier, $t^o_{\boldsymbol{P}id}$, $l^o_{\boldsymbol{P}id}$, and $V_{\boldsymbol{P}id}$ are the occurrence time, location, and attributes of the physical event, respectively. Physical events represent real occurrences in the physical world and hence, reside at the *physical event layer*, where the event properties can be measured/sampled by the sensors installed on sensor motes.

Physical Observation: Physical events are captured through a physical observation, which is a snapshot of attribute, temporal, or spatial status of the target physical event. A physical observation is characterized as:

$$\boldsymbol{O}(\boldsymbol{MT}id, \boldsymbol{SR}id, i)\,\{t^o_{\boldsymbol{O}}, l^o_{\boldsymbol{O}}, V_{\boldsymbol{O}}\} \tag{5.2}$$

where $\boldsymbol{O}(\boldsymbol{MT}id, \boldsymbol{SR}id, i)$ is the physical observation identifier denoting it is made by sensor $\boldsymbol{SR}id$ installed on sensor mote $\boldsymbol{MT}id$, and that it is the ith observation. The physical observation has 3 properties, where $t^o_{\boldsymbol{O}(\boldsymbol{MT}id,\boldsymbol{SR}id,i)}$, $l^o_{\boldsymbol{O}(\boldsymbol{MT}id,\boldsymbol{SR}id,i)}$ and $v_{\boldsymbol{O}(\boldsymbol{MT}id,\boldsymbol{SR}id,i)}$ are the physical observation occurrence time, location and attributes, respectively.

Sensor Event: Sensor motes serve as the first level of observers in the CPS event model, where one or more physical observations can be exploited to generate a sensor event instance based on sensor event conditions. A sensor event instance is represented as:

$$\boldsymbol{S}(\boldsymbol{MT}id, \boldsymbol{S}id, i)\,\{t^g_{\boldsymbol{S}}, l^g_{\boldsymbol{S}}, t^{eo}_{\boldsymbol{S}}, l^{eo}_{\boldsymbol{S}}, V_{\boldsymbol{S}}, \rho_{\boldsymbol{S}}\} \tag{5.3}$$

where $\boldsymbol{MT}id$ is the sensor mote that generates the event based on sensor event ID, $\boldsymbol{S}id$ at the ith event instance. Moreover, a 6-tuple property set is used, where $t^g_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$ and $l^g_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$ are the time and location related to the event instance and the sensor mote, respectively, $T^{eo}_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$, $l^{eo}_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$ and $V_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$ are the estimated event occurrence time, location and attribute according to sensor mote, and $\rho_{\boldsymbol{S}(\boldsymbol{MT}id,\boldsymbol{S}id,i)}$ is the confidence level of the sensor mote regarding the sensor event instance.

Cyber-Physical Event: The WSN sink node serves as the second level of observer in the CPS event model. More specifically, sink nodes collect the sensor event instances from other sensor motes as input observations and generate cyber-physical event instances based on the cyber-physical event conditions. A cyber-physical event instance is represented similar to a sensor event as follows:

$$\boldsymbol{CP}(\boldsymbol{MT}id, \boldsymbol{CP}id, i)\,\{t^g_{\boldsymbol{CP}}, l^g_{\boldsymbol{CP}}, t^{eo}_{\boldsymbol{CP}}, l^{eo}_{\boldsymbol{CP}}, V_{\boldsymbol{CP}}, \rho_{\boldsymbol{CP}}\} \tag{5.4}$$

where $\boldsymbol{MT}id$ is the sensor mote , i.e, the sink node, that generates the event based on the cyber-physical event ID $\boldsymbol{CP}id$. Moreover, $t^g_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$ and $l^g_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$ are the time and location when the sink node generates the event instance, $T^{eo}_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$, $l^{eo}_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$ and $V_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$ are the estimated event occurrence time, location and attributes, and $\rho_{\boldsymbol{CP}(\boldsymbol{MT}id,\boldsymbol{CP}id,i)}$ is the confidence level of the sink node.

Cyber-Event: The CPS control unit (CCU) serves as the highest level of observer in CPS event model. A CCU may combine cyber-physical event instances from sink nodes and other CCUs as input observations to generate the cyber event instances based on cyber event conditions. Accordingly, a cyber event instance is characterized as a cyber event instance ID and 6-tuple event instance properties:

$$\boldsymbol{E}(\boldsymbol{CCU}id, \boldsymbol{E}id, i)\,\{t^g_{\boldsymbol{E}}, l^g_{\boldsymbol{E}}, t^{eo}_{\boldsymbol{E}}, l^{eo}_{\boldsymbol{E}}, V_{\boldsymbol{E}}, \rho_{\boldsymbol{E}}\} \tag{5.5}$$

where $\boldsymbol{E}(\boldsymbol{CCU}id, \boldsymbol{E}id, i)$ is the cyber event instance of the CPS control unit $\boldsymbol{CCU}id$ based on cyber event ID $\boldsymbol{E}id$, $t^o_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$ and $l^o_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$ are the time and location when the CCU generates the event instance, $t^{eo}_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$, $l^{eo}_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$ and $V_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$ are the estimated event occurrence time, location and attributes, respectively, and $\rho_{\boldsymbol{E}(\boldsymbol{CCU}id,\boldsymbol{E}id,i)}$ is the confidence level of the CCU regarding the particular cyber event instance.

6. Conclusion and Future Work

In this paper, a CPS architecture along with an novel event model for CPS are developed. Similar to the hierarchical structure of the CPS architecture, the CPS event model is also separated into several layers, where physical event, physical observation, sensor event, cyber-physical event and cyber-event are defined. In addition, an event is represented as one or more attribute-based, temporal and/or spatial event conditions using the associated operators. The logical operators AND, OR, NOT are then used to generate composite event conditions. This event representation method significantly extends the event temporal relations and additional event spatial relations to capture the complex relationships in a CPS. Furthermore, since information regarding the event occurrence time and location are kept intact, formal temporal and spatial analysis of the cyber-physical systems can be performed using this generic framework.

The future work includes a formal temporal analysis of *Event Detection Latency (EDL)* based on the proposed framework and building an end-to-end latency model for CPSs. Moreover, we will investigate the event condition evaluation at different CPS components.

References

[1] E. Lee, "Cyber physical systems: Design challenges," *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, pp. 363–369, May 2008.

[2] D. McCarthy and U. Dayal, "The architecture of an active database management system," *SIGMOD Rec.*, vol. 18, no. 2, pp. 215–224, 1989.

[3] N. Gehani, H. Jagadish, and O. Shmueli, “Composite event specification in active databases: Model & implementation,” *VLDB ’92: Proceedings of the 18th International Conference on Very Large Data Bases*, pp. 327–338, 1992.

[4] S. Gatziu and K. Dittrich, “Detecting composite events in active database systems using petri nets,” *RIDS’94: Proceedings 4th International Workshop on Active Database Systems*, pp. 2–9, 1994.

[5] S. Chakravarthy, “Sentinel: an object-oriented dbms with event-based rules,” *SIGMOD Rec.*, vol. 26, no. 2, pp. 572–575, 1997.

[6] R. Adaikkalavan and S. Chakravarthy, “Snoopib: interval-based event specification and detection for active databases,” *Data Knowl. Eng.*, vol. 59, no. 1, pp. 139–165, 2006.

[7] I. Lee and O. Sokolsky, “Semantics of high-level event definition,” in *First Workshop on Event-based Semantics*, 2007.

[8] H. Barringer, A. Goldberg, K. Havelund, and K. Sen, “Rule-based runtime verification,” *5th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI’04)*, vol. 2937, pp. 277–306, Jan. 2004.

[9] M. Kim and M. V. etc., “Formally specified monitoring of temporal properties,” *Real-Time Systems, 1999. Proceedings of the 11th Euromicro Conference on*, pp. 114–122, 1999.

[10] M. Kim and S. K. etc., “Java-mac: a rigorous run-time assurance tool for java programs,” *Formal Methods in System Design*, vol. 24, no. 2, pp. 129–155, 2004.

[11] A. Mok, P. Konana, and G. L. etc., “Specifying timing constraints and composite events: an application in the design of electronic brokerages,” *IEEE Transactions on Software Engineering*, vol. 30, no. 12, pp. 841–858, 2004.

[12] A. Mok, C.-G. Lee, H. Woo, and P. Konana, “The monitoring of timing constraints on time intervals,” *RTSS ’02: Proceedings of the 23rd IEEE International Real-Time Systems Symposium*, pp. 191–200, 2002.

[13] R. Meier and V. Cahill, “Taxonomy of Distributed Event-Based Programming Systems,” *The Computer Journal*, vol. 48, no. 5, pp. 602–626, 2005.

[14] T. J. Owens, “Survey of event processing,” AFRL/RIED, Tech. Rep. AFRL-RI-RS-TIM-2007-16, Dec. 2007, in-House Technical Memorandum.

[15] E. Wu, Y. Diao, and S. Rizvi, “High-performance complex event processing over streams,” *SIGMOD ’06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 407–418, 2006.

[16] T. Abraham and J. F. Roddick, “Survey of spatio-temporal databases,” *Geoinformatica*, vol. 3, no. 1, pp. 61–99, 1999.

[17] M. J. Egenhofer and K. K. Al-Taha, “Reasoning about gradual changes of topological relationships,” *Proceedings of the International Conference GIS*, pp. 196–219, 1992.

[18] Y. Tan, S. Goddard, and L. C. Pérez, “A prototype architecture for cyber-physical systems,” *SIGBED Review, Special Issue on the RTSS Forum on Deeply Embedded Real-Time Computing*, vol. 5, no. 1, 2008.

[19] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Computer Networks (Elsevier) Journal*, vol. 38, no. 4, pp. 393–422, March 2002.

[20] I. F. Akyildiz and I. H. Kasimoglu, “Wireless sensor and actor networks: Research challenges,” *Ad Hoc Networks (Elsevier)*, vol. 2, no. 4, pp. 351–367, October 2004.

[21] S. Chakravarthy and D. Mishra, “Snoop: an expressive event specification language for active databases,” *Data Knowl. Eng.*, vol. 14, no. 1, pp. 1–26, 1994.

[22] M. C. Vuran and O. B. Akan, “Spatio-temporal characteristics of point and field sources in wireless sensor networks,,” *Proc. IEEE ICC ’06*, June 2006.