

# GEM: Generic Event Service Middleware for Wireless Sensor Networks\*

Binjia Jiao Sang H. Son John A. Stankovic  
University of Virginia  
Charlottesville, Virginia 22903 USA

## ABSTRACT

Most wireless sensor network (WSN) applications are event-based and their special features demand a new paradigm for event services. This paper presents a framework for a generic event service middleware (GEM). GEM provides an integrated service package for WSN applications so that users can specify events accurately, export specified events to the network, and initiate in-mote middleware to provide multi-level event detection. A sensor network event description language (SNEDL), designed specifically for WSN events, is a part of the GEM architecture. Built upon Petri-Nets, SNEDL can also be used as an offline analysis tool. A case study is presented to demonstrate the usefulness of SNEDL. To make the sensor motes understand SNEDL specified events, GEM encodes events into a mote-understood format (called event DNA). GEM also contains an in-mote detection module to read DNAs for event recognition. When communication is necessary for higher level events, a detection module transfers a lightweight structure (called RNA) to support collaborative decisions.

## 1 INTRODUCTION

As a next generation computing platform, wireless sensor networks (WSNs) are being applied to more and more areas for surveillance and monitoring purposes [15]. In these applications, WSNs are expected to detect interesting events specified by the users, and/or respond to these events. An essential event service package for WSNs is to provide a platform for users to specify the events of interest, then to transform them into the data format that instructs motes how to detect the events and/or how to react to them. Higher level events such as group level and global level events can be recognized by aggregating multiple local events. Specifications for these higher level events are also necessary for an integrated event service. For some WSN applications, events can be simple since they only depend on the value of a particular sensor reading without spatial-temporal constraints. These types of events are usually called atomic events [10]. Some events (complex events) have complicated temporal and spatial constraints and correlations and thus are difficult to specify. One of the critical functions of an event service is accurately specifying events, especially complex and high level events, and interpreting event semantics into formats recognizable by motes. In most WSN literature, researchers use SQL or SQL-like languages to describe the events [3, 6, 10, 11]. However, SQL-like semantics are not always suitable for sensor networks because of the lack of collaborative decision making and other necessary functionalities.

The inadequacy of SQL languages in specifying WSN events comes from some of the essential characteristics of events in sensor networks :

- Data dependency and correlation among different types of sensors in heterogeneous WSNs are hard for SQL to capture.

- WSNs are distributed, concurrent and asynchronous, while meaningful events usually require spatial and temporal composition of ad-hoc sensor readings. SQL is awkward in collaborative decision making and representing event triggering [5].
- Each individual mote is unreliable, and it results in non-determinism in event detection. To tolerate such non-determinism, a probabilistic model should be considered in defining events. SQL has no explicit support in this regard.
- Most WSN applications need to form groups for data aggregation and event detection. Mote-level events, group level events and global level events can form a hierarchy of events. SQL-like languages do not naturally present a hierarchical model for event structure.

The Petri Net is well accepted as a model to tackle a system which has distributed, concurrent, asynchronous and non-deterministic features. Inspired but not limited by Petri Nets, we develop an event description language, SNEDL which has the following contributions:

- To the best of our knowledge, the first even description language specially designed for wireless sensor network applications.
- It tackles the challenges of sensor network events, namely composing complex events with temporal and spatial constraints and correlations.
- Based on Petri Nets, it not only provides symbolic language based specification, but also provides a visual picture of the event structure.
- SNEDL is a hierarchical approach with different levels of abstractions for differentiated specification purposes.
- In addition to being a description language, SNEDL can also be used as an analysis tool for studying event system properties by SNEDL net simulation.

SNEDL enables users to specify events, and important design decisions can be made according to event specifications. For instance, with event size specified, group size can be determined if groups are formed based on geography.

---

\* This work is supported in part by NSF CCR-0329609.

The next step after specification is to convert the SNEDL event specification into data formats for motes in the network, to tell each node what exactly they are expected to detect. Since the process of recognizing the specified events in GEM is like a DNA transcript procedure, we call the data format, converted from SNEDL specification and stored in motes' data memory, event-DNA. Each event DNA is an encoded representation of a SNEDL event, and all event DNAs are encoded by the same rule. There is an embedded event detection middleware in motes' program memory, which can read different event-DNAs and act accordingly.

The overall architecture of GEM is shown in Figure 1. In the figure, the SNEDL IDE (Integrated Development Environment) is an offline package which resides on a PC, from which specified event semantics can be encoded and exported to a base mote. The base node installs a deployment module that deploys all event DNAs onto their corresponding motes. For example, if an event DNA represents a mote-level event, it will be deployed onto every mote, while if an event DNA represents a group level event, only a group leader will have a copy. In reality, such a base mote is usually a laptop, carried around the network to communicate with motes for initialization purposes. For dynamic leader schemes, the deployment module needs to interact with group module to determine what to deploy. Then inside each mote, imported event DNA is stored in data memory and an event detection module is embedded in program memory. The detection module achieves its goal by reading event-DNA and calling other lower primitives in the program section such as reading sensor values, obtaining time, location information and other data from other modules in the program section. The GEM framework contains the following modules:

- SNEDL environment which includes both a specification module and an analysis module. It also encodes event specification into event-DNA and exports them to the base node.
- Base node contains encoded event DNA and has a deployment module in charge of transferring event-DNA to motes in the field. The deployment module provides communication, in which messages are event DNAs.
- At the mote level, data memory contains event DNAs and an event detection module resides in program memory. The event detection module uses a token vector (called RNA) to communicate with other motes for collaborative detection for higher level events.

This paper is organized as follows: Section 2 describes the SNEDL language in detail, concentrating on its specification and analysis capabilities, followed by a case study in Section 3. Section 4 presents a DNA-based approach used in the event detection module. Related works are discussed in Section 5. Section 6 presents the implementation of GEM and future work, and Section 7 concludes the paper.

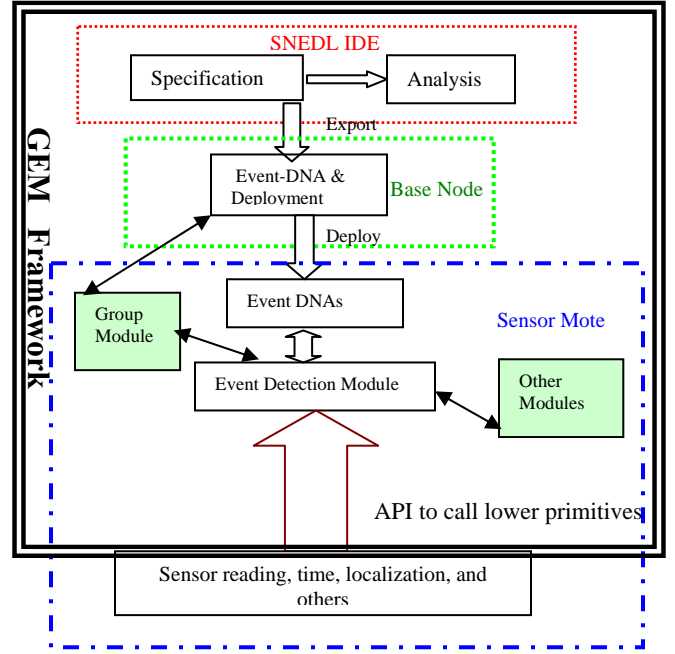


Figure 1: Architecture of GEM

## 2 SNEDL

Petri Nets are the base model for SNEDL due to its ability to tackle concurrent and non-deterministic distributed systems. A basic Petri Net consists of places (circles), transitions (bars), directed arcs and tokens (dots inside places) as in Figure 2. Tokens model instances / objects, and places represent the states in which the objects can be in. Arcs represent the path in which tokens travel through; they also represent changes between states. Transitions are used to check conditions (token attributes) and take actions such as manipulating passing tokens. Because events in WSN are in fact spatiotemporal combinations of sensor readings and/or status messages, SNEDL has the following features:

- Tokens travel through places and arcs, and they are much like messages communicated in WSN. Therefore, sensor readings and messages are naturally modeled using tokens. We extend Petri Net tokens to include attributes unique in WSN such as power level and sensing range.
- In SNEDL, arcs indicate a communication path and we impose a timer function on arcs to indicate the valid interval for a sensor reading/message.
- Transitions represent conditions and actions - a mote's computing and processing of readings / messages. It tests if passing tokens satisfy the guard functions and take actions (change tokens) if they do. In this way, transitions naturally model the event detection process in WSN, namely testing if readings / messages satisfy the spatiotemporal conditions of events, and creating a new message to indicate whether a state has been reached.

Refer to [8] for other specifics and a detailed example of SNEDL.

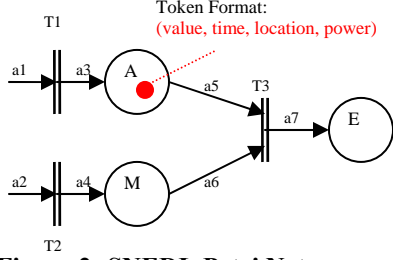


Figure 2: SNEDL Petri Net

## 2.1 SNEDL Semantics

### SNEDL Petri Net Definition

A SNEDL representation of an event is given as a 10-tuple structure

$S = (P, T, A, \lambda, \delta, \theta, H, L, Q, M)$  where

- $P$  is the set of all places, which includes places for sensor level events and those for higher level events.
- $T$  is the set of all transitions, and transitions are represented as rectangle bars in the structural diagram.
- $A$  is the set of arcs/flows, which are represented as arrows in the diagram. Note that  $A$  contains set of pre-arcs (incoming arcs to a transition)  $I$ , and set of post-arcs (outgoing arcs from a transition)  $O$ .

To this point, the definition is exactly an ordinary Petri Net. SNEDL extends this basic Petri Net model to integrate temporal, spatial and probability features.

- $\lambda$  is the probability / weight function for the arcs  $\lambda: A \rightarrow [0,1]$ . For example, assume  $f$  is a post arc from transition  $T$  to state  $B$ , then  $\lambda(f) = p$  means that after  $T$  fired, with probability  $p$  the token enters state  $B$ . As for a pre-arc, it can be viewed that a token puts a weight  $p$  on its capacity  $c$ , the capacity changes into  $c \cdot p$  when it goes through this arc. With this function, SNEDL takes non-determinism into account such that some events can be specified using a probabilistic model.
- $\delta$  is a time guard function for transitions,  $\delta: T \rightarrow (r1, r2)$ , where  $r1 \leq r2$  are real. It means a transition can only fire during a given time interval. For example  $\delta(T) = (a1, a2)$  means transition  $T$  can only be fired during interval  $(a1, a2)$ . This function is used for defining temporal constraints.
- $\theta$  is the persistency guard for arcs. For a pre-arc of a transition,  $\theta: I \rightarrow R^+$ , it means the arc can only hold the token to participate in a transition for a certain amount of time before this token expires. This models valid intervals for sensed data. For example, lightning data can only be valid for 1 second, while the existence of some chemical may last for 1 day. Mapping the semantics onto a sensor network, a  $\theta$  value for a pre arc refers to the valid interval for the tokens

in the place, which stands for a sensor event or a higher level event. For an outgoing arc (post-arc) of a transition,  $\theta: O \rightarrow R^+$ , means how long it takes for this event to happen, which is the delay of communicating a message.

- $H$  is the threshold function for places only,  $H: P \rightarrow R$ . For example  $H(p) = c$  means that if a token with a certain capacity wants to enter  $p$ , and the capacity is over  $c$ , then this token can reach place  $p$ , else it cannot. It can be used to model the threshold of sensor readings.
- $L$  is the spatial guard function for transitions,  $L: T \rightarrow R^+$ . This function is only one particular spatial guard function to guarantee that the sensed data are within the event radius. However, in SNEDL, we can also generalize spatial guard functions to support more complex spatial constructs. At this point, we focus on using  $L$  to model valid event size. For example, in Figure 2, we define  $L(T3) = r$ , then it means only if sensor readings from  $A$  (acoustic) and readings from  $M$  (magnetic) are localized within radius  $r$ , they can be called event  $E$ . In other words, radius of  $E$  is  $r$ .
- $Q$  is a quorum / frequency function on transitions to filter out false alarms by requiring a quorum of positive readings. For example in Figure 2,  $Q(T3, A) = (3, 10)$  means that tokens coming from  $A$  can only trigger transition  $T3$  if they appear 3 times in a row within a window of 10 time units. Here  $A$  represents the token type associated with place  $A$ .
- $M$  is a token manipulating function on post-arcs which explicitly defines attribute values for newly generated tokens. The new attribute values may depend on the values of old tokens. In Figure 2, transition  $T3$  has two pre-arcs with two types of tokens  $A$  and  $M$ . When tokens from  $A$  and  $M$  come to  $T3$  and fire  $T3$ , they will get merged into new tokens and sent to post-arcs of  $T3$ .  $M(a7)$  represents the definition for the new token that will travel through  $a7$  to  $E$ . For instance,  $M(a7) = (v(A,M); t(A,M); l(A,M); p(A,M))$  defines the new token attribute values, where  $v, t, l, p$  are functions on manipulating old tokens' attributes. With the token mapping function  $M$ , all the messages are explicitly defined in SNEDL.

### Token Path and Firing Rules

We outline the path of a token in SNEDL as below:

- A token is initially generated upon receiving sampling data from a sensor and encapsulated with the type (sensor event), time (when the sample taken), capacity (the value of the sensor reading) and the location (where the sensor is, and what the sensing range is), as shown by transitions  $T1, T2$  and  $T3$  in Figure 2.
- Tokens in places go immediately through the arcs to get ready to fire transitions, given that firing of a transition should respect all the guard functions.

- A token disappears if the arc carrying it exceeds its persistence value (defined by  $\theta$ ) before the transition can be fired.
- When a token goes through a pre-arc, its sensor reading value is multiplied by the weight on the arc (defined by  $\lambda$ ). In this way, we can give different weights for different readings to distinguish their importance levels in collaborative decision making.
- When a token goes through a post-arc, it enters the destination place of this post-arc according to the probability defined on this post-arc. In this way, non-determinism of individual nodes can be modeled. If all probabilities are defined as one, then no non-determinism is considered.
- A token can only enter a place if the token's sensor reading value is over the place's threshold value (defined by  $H$ ).

The token generation/destruction process is done through transition firing processes. Transition firing rules and the token processing procedure are presented as follows:

- A transition  $T$  can be fired if and only if
  - each of its pre-arcs has a positive token,
  - $T$  happens at a time interval satisfying  $\delta$ .
- During firing, a transition  $T$  does the following on the tokens carried by its pre-arcs:
  - Generates a new token for each of the post-arcs according to mapping function  $M$ .

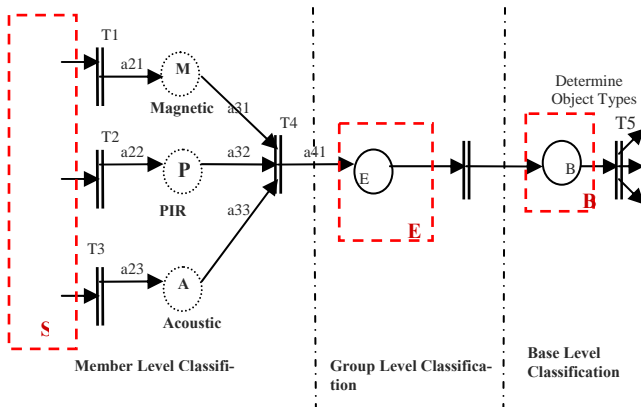


Figure 3: Hierarchical Event Specification

## 2.2 SNEDL Analysis

SNEDL provides a generic specification environment for users to define how an event is interpreted in a WSN, and users also need to define system properties they need to analyze. SNEDL has analysis capability because it is not only a specification tool, but also provides a simulation engine.

After an event is specified as a SNEDL Petri Net, inserting stream of tokens will result in different event patterns. These streams of tokens represent ad-hoc sensor readings across the network. Since a SNEDL event specified in such a way simulates how nodes interpret readings/messages into

events, the analysis process is actually studying the impact of a specific system design on event patterns and other system properties.

As shown in Figure 4, ad-hoc sensor readings across the network are represented by tokens in the sample space. According to their timestamps, tokens are dispatched in streams into the SNEDL event system specified. To study a certain system property such as timeliness of event detection, statistics functions are built on a component such as a transition to monitor attributes of interest of all the tokens.

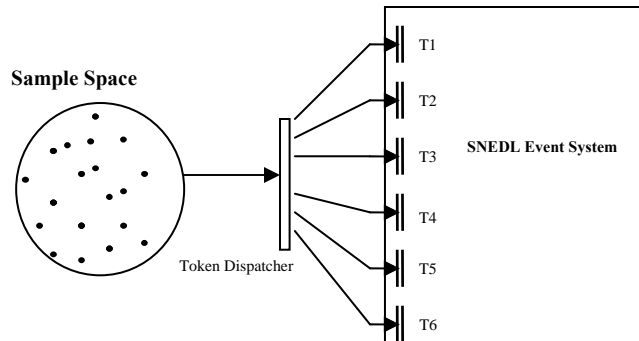


Figure 4: SNEDL Analysis Workload Generation

There is a lot of flexibility in SNEDL analysis. For example, if we design the token dispatcher to dispatch tokens according to their locations, then we can analyze geographic system properties.

## 3 CASE STUDY

To illustrate how SNEDL can be used in a real WSN application, we present a case study of a surveillance application in sensor networks. DARPA (Defense Advanced Research Projects Agency) sponsored the NEST project [12]. In this project a large surveillance system has been implemented and tested, in which the main purpose is to monitor a large area for objects of interests and classify them. Due to space limitation, we only provide a brief outline in this paper. More detailed specification and analysis procedures are provided in [8].

### 3.1 Hierarchical Specification

There are three types of sensors – acoustic, magnetic, and PIR (motion) – in the network sampling environments. The major task of this application is to distinguish three types of events – a person, a person with weapons, and a vehicle – from ad-hoc sensor readings. The system designers have developed a 3-tier classification approach to identify a particular event. As shown in Figure 3, the three tiers are mote-level, group-level and base-level event classification. At the mote level, readings from M (magnetic sensors), P (PIR sensors), and A (acoustic sensors) are processed at transition T4, where temporal and spatial conditions are checked and new messages are generated for next level (group level) processing. SNEDL is a hierarchical approach, which means that each component (indicated as dashed box) can be ex-

tended to a sub SNEDL net to provide more design details. For instance, extending sensing activity component S in Figure 2 dashed box discloses a SNEDL structure as shown in Figure 5.

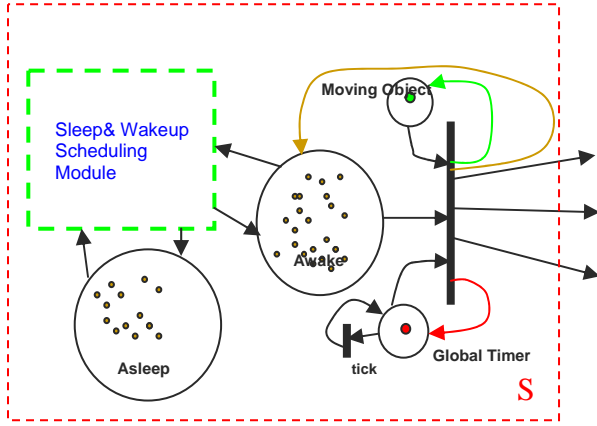


Figure 5: Hierarchical Approach – Extending Component S

In addition, hierarchies can be nested, such as the scheduling module in Figure 6. It can be viewed as a transition in S, yet it can be extended to a sub-net disclosing more details in how a scheduling module is implemented.

### 3.2 System Property Analysis

In this case study, there are two system properties of interest: maximum speed detectable and system lifespan. Once a system property is defined using the parameters in a SNEDL Petri Net, it can be evaluated by running the simulation engine and statistics collector.

**Definition:** The maximum speed detectable (MSD) is defined as the highest speed of a moving object  $v$  such that from time  $t1$  to  $t2$ , we have at least  $n$  number of distinguished location reports collected, where  $n$  is a function of  $t1$ ,  $t2$  and average sensing range  $r$ .

With this definition, we can evaluate the maximum speed detectable using the following steps:

- Specify NEST event system architecture using SNEDL.
- Insert token streams representing different sensor readings when an object is moving at different speeds, and run the simulation engine to see if over  $n$  messages have been received.
- Increase object speed until it does not satisfy the definition to identify the maximum detectable speed.

To analyze the system lifespan, a power level attribute is modeled into tokens, such that when tokens travel through arcs and transitions, their power level can be reduced accordingly. Statistics on power levels will be collected, and based on definition of system lifetime, the data can be used to determine how alive a system is. Users can define system lifetime differently based on different applications.

## 4 DNA-BASED EVENT DETECTION

As discussed in Section 1, SNEDL specified events are encoded into a data format to be stored in motes. This type of data format is called event DNA, and they are normally deployed during the initialization phase. In detecting group-level events or global-level events, status about a mote level event is needed from each mote. However, if we transfer a marked event DNA (a SNEDL Petri Net with tokens), it will be too heavyweight for WSN with a constrained packet size. To address this issue, we designed a light-weight structure (called RNA or token vector) representing the event status on a mote, and transfer this type of light-weight structure across the network. This process is very similar to the biological DNA/RNA transcript process. A SNEDL specification is static in the sense that only the event structure is specified. The token vector is a dynamic structure which records the changing status of the event described by the DNA. For example, as in Figure 2, symbol representation of the event structure can be given as:

- *Components:*  $P = \{A, M, E\}$ ,  $A = \{a1, a2, a3, a4, a5, a6, a7\}$ ,  $T = \{T1, T2, T3\}$
- *Token Format:* e.g.  $A\{soundvalue, time, location\}$
- *ArcPlace Relations:*  $a3-A$ ,  $a4-M$ ,  $a7-E$ ;  $A-a5$ ,  $M-a6$ .
- *ArcTransition Relations:*  $a1||T1$ ,  $a2||T2$ ,  $a5||T3$ ,  $a6||T3$ ;  $T1||a3$ ,  $T2||a4$ ,  $T3||a7$ .
- *Guard functions:* e.g.  $T1\{L, \lambda, \delta, L, Q\}$ ,  $a3\{\theta, M\}$ ,  $A\{H\}$

The token vector is a vector of tokens for each place in the event structure, indicating the marking of the event structure. Using the same example as in Figure 2, the token vector will look like:

- *Token Vector:*  $(A:1-\{value, time, location, power\} M:0-\{E:0-\{}$ )

If a place has a token, it indicates that there is a positive sensor reading or a message and a “1” will mark the corresponding place in the vector with the values for all token attributes. Otherwise it is marked by “0” without any token attribute values.

This DNA-based event recognition method has substantial advantages in handling event aggregation, privacy maintenance and other features. This method is ongoing research which will be refined in the future work.

## 5 RELATED WORK

From the event description language perspective, there has been little work on specifically designing a description language for specifying events in sensor networks. One relevant work can be found in [14], in which an object-oriented model was proposed to represent sensor network events. However, the approach lacks supporting data interactions and collaborative decision making.

In databases and other conventional areas, a lot of formalized approaches are used for event descriptions and compositions purposes. A survey on formal methods for specifica-

tion and analysis [1] shows that most of the popular approaches are based on the theoretical models such as finite state machine, timed automata, process algebra and Petri Nets [13]. SNEDL is based on a basic place / transition Petri Net with extensions adapted to WSN characteristics. Other types of Petri Nets such as Color Petri Nets [7] and Timed Petri Nets [4] have difficulty in supporting these features. In addition, SNEDL is also designed as a compatible part of the GEM event service system, which establishes its uniqueness in these aspects. SNEDL can also be used to analyze system properties and its analysis capability is inherited from Petri Nets. A related work on analysis can be found in [2], where authors use hybrid automata to analyze system lifetime of a sensor network.

Although a lot of interesting work has been done in sensor networks, to the best of our knowledge, no prior research directly focuses on providing event services as an integrated framework like GEM for wireless sensor networks.

## 6 IMPLEMENTATION

GEM is ongoing research work; currently SNEDL semantics and analysis design have been finished. A prototype of the SNEDL tool for specification purposes has been built using GME [9] as illustrated in Figure 6. A full scale tool supporting other functions such as analysis and simulation engine is being implemented using Java. Other modules such as the DNA-based event recognition, deployment module are currently being refined for efficient implementation.

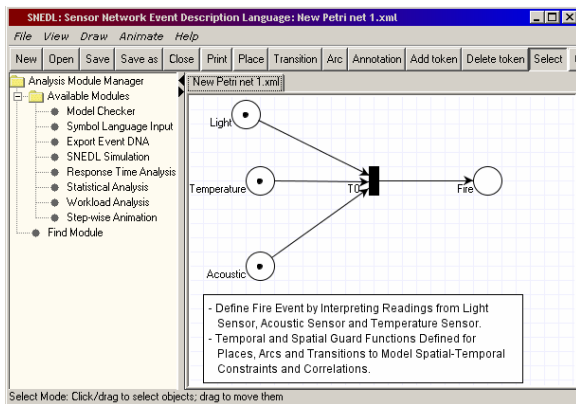


Figure 6: SNEDL Specification Tool

## 7 CONCLUSION

Most wireless sensor network (WSN) applications are event-based and their special features demand a new paradigm for event services. This paper presents a framework of generic event service middleware, called GEM. GEM aims to provide an integrated service package for WSN applications so that users can specify events precisely, export specified events to the network, and initiate in-mote middleware to do multi-level event detection. A sensor network event description language (SNEDL) is presented as a part of GEM archi-

ture, which is developed specifically for WSN events. Built upon Petri-Nets, SNEDL can also be used as an offline analysis tool. A brief case study is presented to demonstrate the usage of SNEDL. To make the sensor motes understand SNEDL specified events, GEM encode events into mote-understood format (event DNA). GEM also contains an in-mote detection module to read DNAs for event recognition. When communication is necessary for higher level events, detection module transfers a lightweight structure (called RNA) for collaborative decisions.

## REFERENCES

- [1] Fulvio Babich and Lia Deotto. "Formal Methods for Specification and Analysis of Communication Protocols," *IEEE Communications Surveys & Tutorials*, 2002.
- [2] S. Coleri, et al "Lifetime Analysis of a Sensor Network with Hybrid Automata Modeling", *Proceedings of 1<sup>st</sup> ACM International Workshop on Wireless Sensor Network and Applications*, Atlanta, Georgia, USA.
- [3] Cougar Project. [www.cs.cornell.edu/database/cougar](http://www.cs.cornell.edu/database/cougar).
- [4] F. DiCesare, et al. "Practice of Petri Nets in Manufacturing." Chapman & Hall, 1993.
- [5] Michael Franklin. "Declarative Interfaces to Sensor Networks", *Presentation at NSF Sensor Workshop*, Los Angeles, CA, Feb, 2004.
- [6] R. Govindan, et al. "The Sensor Network as a Database." Technical Report 02-771, Computer Science Department, University of Southern California, Sept 2002.
- [7] K. Jensen. "Colored Petri Nets and the Invariant-Method". *Theoretical Computer Science*, 14:317-336, 1981.
- [8] B. Jiao, S. Son, and J. Stankovic "SNEDL: Sensor Network Event Description Language", *University of Virginia, CSTR9400*. [www.cs.virginia.edu/~bj3r/snedl.pdf](http://www.cs.virginia.edu/~bj3r/snedl.pdf)
- [9] A.Ledeczki, et al. "Metaprogrammable Toolkit for Model-Integrated Computing." *In Proceedings of the IEEE ECBS'99 Conference*, 1999. 53
- [10] S. Li, S. H. Son, and J. A. Stankovic, "Event Detection Services Using Data Service Middleware in Distributed Sensor Networks," *2<sup>nd</sup> International Workshop on Information Processing in Sensor Networks (IPSN'03)*, Palo Alto, CA, April 2003.
- [11] S. Madden, et al "The Design of an Acquisitional Query Processor for Sensor Networks," *In Proceedings of ACM SIGMOD*, San Diego, CA, June 2003.
- [12] NEST Project [www.cs.virginia.edu/~control](http://www.cs.virginia.edu/~control)
- [13] C. A. Petri. "Kommunikation Mit Automaten." *Dissertation, Technische Universitat Darmstadt*, 1962.
- [14] M. Worboys, "Event-based Models of Geosensor Networks," *Invited Talk, First Workshop on Geo Sensor Networks*, Portland, Maine, Oct, 2003.
- [15] F. Zhao, et al. "Wireless Sensor Network: An Information Processing Approach", *Morgan Kaufmann Networking Series*, Jul 2004.