DEVS-Based Modeling and Simulation of Wireless Sensor Network

Songyuan Gu, Chen Wu, and Yuanyuan Qin(✉)

China Academy of Electronics and Information Technology, Beijing, China
gusongyuan614@163.com, wuchen5251@163.com, qinyuan98@163.com

Abstract. A modeling and simulation method for wireless sensor network (WSN) using discrete event system specification (DEVS) is proposed in this paper considering that the existing simulators cannot fully satisfy the requirements of WSN simulation. The method is for multi-layer and multi-aspect modeling which contains the component layer in sensor nodes, the sensor node layer, the wireless sensor net-work layer and the external environment where a WSN is exposed, and modules in lower layer are integrated into superior models through coupling relation and model reuse. Minimum Hop Count (MHC) protocol is chosen as the routing protocol of the WSN in the simulation experiment. It is finally demonstrated through the performance analysis of WSN using MHC that the proposed DEVS-based modeling and simulation method is feasible for WSN simulation.

Keywords: Wireless sensor network · Modeling and simulation · Discrete event system specification · Minimum-hop-count routing

Wireless sensor network (WSN) is a wireless network system composed of a number of spatial distributed sensor nodes with wireless transmission capabilities through self-organization. The fundamental purpose of WSN establishing is to comprehensively perceive and monitor target area, and to obtain spatial and temporal distributed data in target area for monitoring and analyzing the environment of target area. WSN is widely used in several domains e.g. military, industry and environment monitoring; especially it can execute data acquisition and monitoring by entering the hard to-reach areas. WSN is also the concept foundation and the underlying network of Internet of Things [1, 2].

For optimizing the design of sensor nodes and network and achieving stable and robust WSN deployment for complex environment, it is necessary to perform simulation test for performance parameters and deployment environment of a WSN. Modeling and simulation is an essential method to observe WSN behaviors under certain conditions which can analyze topology and communication protocol or forecast network performance. There are several simulators capable of WSN simulation e.g. NS2, OPNET, OMNeT++, SensorSim, TOSSIM, etc. but these simulators could partly satisfy simulation demands for WSNs and it is difficult to find a universal, customizable and modular simulator applied in simulation of behaviors and performance of a WSN and in modeling of environmental scenario a WSN exposed in [3]. In the field of WSN simulation, we need a universal simulation method which can describe internal structure of sensor nodes completely, be module-reusable and customize the external environment of a WSN. Discrete event system specification (DEVS) is able to model system

Q. Zu et al. (Eds.): HCC 2020, LNCS 12634, pp. 19–30, 2021.
https://doi.org/10.1007/978-3-030-70626-5_3

in multi-level, multi-granularity and provide simulators to simulate models. Therefore, this specification is employed in this paper to model modular sensor nodes, WSNs and external environments of WSNs and to perform simulation experiments for verifying the feasibility of WSN simulation using DEVS.

1 Discrete Event System Specification

Discrete event system specification is a modular, hierarchical and formal modeling and simulation mechanism based on general system theory which is founded by Bernard Zeigler [6–8]. The specification is able to characterize systems in which finite variations arise in finite intervals of time, e.g. discrete event systems, discrete time systems and continuous time systems [5]. Modeling and simulation stages are clearly divided in DEVS and a simulation experimental framework is provided in which models can be executed. In terms of modeling, DEVS divides models into two levels: atomic models (AM) and coupling models (CM); atomic model defines internal transitions under time domain variations and external input conditions as well as input and output behaviors; coupling model defines hierarchical structures between atomic models or between atomic models and lower level coupling models. General atomic model is defined as a 7-tuple:

$$AM = <X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta> \quad (1)$$

Where X is an input event set which includes input event values and input ports; Y is an output event set which includes output event values and output ports; S is a system state set; δ_{int} is an internal transition function in which a system state would transit from s to $\delta_{int}(s)$ when there are no external input events for a certain time; δ_{ext} is an external transition function in which a system has remained in a state for time interval e when external input event $x \in X$ arrives, then the system state would transit to $\delta_{ext}(s, e, x)$ at once and e would recover to 0; λ is an output function which outputs event $\lambda(s)$ before state transition; ta is a time advance function.

General coupling model could be defined as

$$CM = <X, Y, D, \{M_d | d \in D\}, EIC, EOC, IC, select> \quad (2)$$

Where D is a coupling member name set; $\{M_d\}$ is a member model set; EIC represents external input coupling relations which connect input ports of a coupling model to input ports of internal member models; EOC represents an external output coupling relations which connect output ports of internal member models to output ports of a coupling model; IC is an internal coupling relation set which connect input ports of some internal members to output ports of other members; $select$ is a selection function which is employed to select a state transition as the one of a coupling model from the concurrent state transitions of internal members.

2 DEVS-Based WSN Modeling

Sensor nodes are capable of computing, communicating and perceiving [9]. The architecture of typical sensor nodes is shown as Fig. 1. A typical sensor node is generally composed of a sensor component, a processor component, a wireless communication component and a power component; some sensor nodes also comprise positioning components and mobile components. The sensor component is responsible for sampling target data, determining whether sampled value exceeds a threshold and transferring sampled data to processor component; the processor component is responsible for managing activities of entire node such as the processing for sampled data as well as the processing and the routing for messages to be forwarded; the wireless communication component is responsible for wireless channel monitoring, messages receiving and forwarding to a channel.

Distributed WSN nodes communicate with each other via multi-hop and thus constitute a WSN system. Each node transfers sampled data to a sink node directionally. The sink node, which has greater processing capability and more adequate power, is a gateway node connecting the WSN to Internet or a satellite communication network. Subsections below present the establishment of the models of component-layer, node-layer and network-layer based on DEVS in detail.

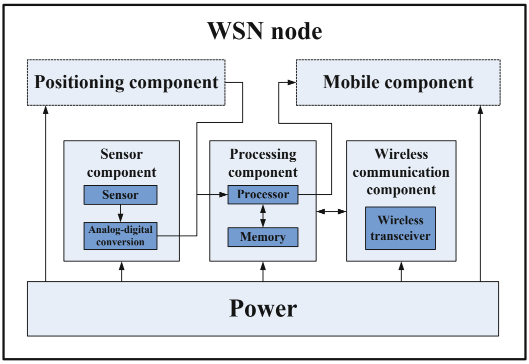

Fig. 1. Architecture of a wireless sensor node

2.1 Sensor Component Modeling

The sensor component atomic model describes perceiving and interaction behaviors of a sensor node with external environment. The structure of sensor component atomic model AM_{Sensor} is shown as Fig. 2. As the unique component interacting with external environment in a sensor node, the sensor component model reserves the port *Env_info* for external environment data input, which could connect to external environment model through an external input coupling relation and interact with external environment or not according to the simulation scenario. For establishing an integrated simulation scenario, diversified external environment models could be established with freedom to fulfill diversified simulation demands. An external temperature model $AM_{Temperature}$ is defined in this paper to provide external input for the WSN. Input port *Power_info* and output port *Power_cons* are utilized to interact with the power component: the power component transfers current state of itself to the sensor component through *Power_info*; the sensor component transfers its working condition to the power component through *Power_cons* for calculating the power consumption. Output port *Sample_info* is utilized to transfer the sampled data to the processor component.

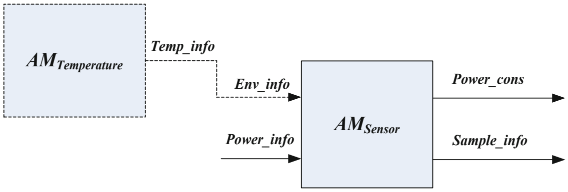

Fig. 2. Architecture of AM_{Sensor}

The formal description of atomic model AM_{Sensor} is as follows

$$
\begin{aligned}
&AM_{Sensor} = <X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta> \\
&X = \{\text{“}Env\_info\text{"}, \text{“}Power\_info\text{"}\} \\
&Y = \{\text{“}Sample\_info\text{"}, \text{“}Power\_cons\text{"}\} \\
&S = \{RUN, DEAD\} \\
&\delta_{ext} : RUN \times \text{“}Env\_info\text{"} \rightarrow RUN \\
&\quad RUN \times \text{“}Energy\_info\text{"} \rightarrow DEAD \\
&\delta_{int} : RUN \rightarrow RUN \\
&\lambda : RUN \rightarrow \text{“}Sample\_info\text{"} \\
&\quad RUN \rightarrow \text{“}Power\_cons\text{"} \\
&ta : RUN \rightarrow t_{Sample} \\
&\quad DEAD \rightarrow \infty
\end{aligned}
$$

2.2 Processor Component Modeling

The processor component is responsible for managing each of the components in a sensor node, processing information received by the wireless communication component or sampled by the sensor component and routing for messages to be forwarded. Processor component atomic model $AM_{Processor}$ is shown as Fig. 3. Input port *Mess_info* and output port *Proc_info* are responsible for interacting with wireless communication component: the wireless communication component delivers received messages to the processor component through *Mess_info* to process them, and if continue forwarding is needed, the processor component would calculate the route; the processor component delivers messages to be forwarded to the wireless communication component through *Proc_info*. Input port *Power_info* and output port *Power_info* are utilized to transfer current state of power component and calculate power consumption as well. Input port *Sample_info* is responsible for receiving sampled value from the sensor component. Calculation and storage functions are simulated in atomic model $AM_{Processor}$.

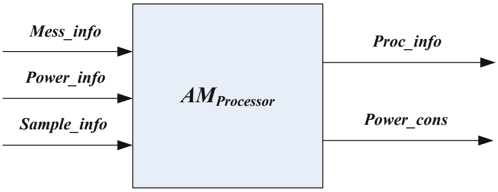

Fig. 3. Architecture of $AM_{Processor}$

The formal description of atomic model $AM_{Processor}$ is as follows

$$
\begin{aligned}
& AM_{Processor} = <X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta> \\
& X = \{\text{“}Mess\_info\text{"}, \text{“}Sample\_info\text{"}, \text{“}Power\_info\text{"}\} \\
& Y = \{\text{“}Proc\_info\text{"}, \text{“}Power\_cons\text{"}\} \\
& S = \{RUN, DEAD\} \\
& \delta_{ext} : RUN \times \text{“}Mess\_info\text{"} \rightarrow RUN \\
& \quad RUN \times \text{“}Sample\_info\text{"} \rightarrow RUN \\
& \quad RUN \times \text{“}Power\_info\text{"} \rightarrow DEAD \\
& \delta_{int} : RUN \rightarrow RUN \\
& \lambda : RUN \rightarrow \text{“}Proc\_info\text{"} \\
& \quad RUN \rightarrow \text{“}Power\_cons\text{"} \\
& ta : RUN \rightarrow t_{Proc} \\
& \quad DEAD \rightarrow \infty
\end{aligned}
$$

2.3 Wireless Communication Component Modeling

The wireless communication component model represents communication behaviors of a sensor node, namely the message interactions with nodes within communication radius of current node. Wireless communication component atomic model $AM_{Transceiver}$ is shown as Fig. 4. Input port *Mess_rec* and output port *Mess_forw* communicate with other nodes through receiving and forwarding messages; input port *Proc_info* is utilized to acquire data processed from the processor component; meanwhile, output port *Mess_proc* is utilized to transfer messages to be processed to the processor component.

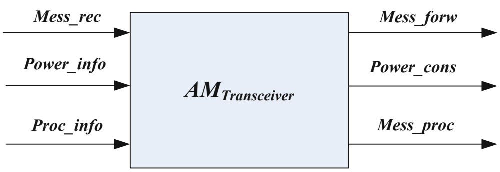

Fig. 4. Architecture of $AM_{Transceiver}$

The formal description of atomic model $AM_{Transceiver}$ is as follows

$$AM_{Transceiver} = <X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta>$$
$$X = \{\text{“}Mess\_rec\text{"}, \text{“}Proc\_info\text{"}, \text{“}Power\_info\text{"}\}$$
$$Y = \{\text{“}Mess\_forw\text{"}, \text{“}Mess\_proc\text{"}, \text{“}Power\_cons\text{"}\}$$
$$S = \{RECEIVE, IDLE, FORWARD, DEAD\}$$
$$\delta_{ext} : RECEIVE \times \text{“}Mess\_rec\text{"} \to RECEIVE$$
$$FORWARD \times \text{“}Proc\_info\text{"} \to FORWARD$$
$$RECEIVE \times \text{“}Power\_info\text{"} \to DEAD$$
$$IDLE \times \text{“}Power\_info\text{"} \to DEAD$$
$$FORWARD \times \text{“}Power\_info\text{"} \to DEAD$$
$$\delta_{int} : RECEIVE \to IDLE$$
$$IDLE \to FORWARD$$
$$FORWARD \to RECEIVE$$
$$\lambda : RECEIVE \to \text{“}Mess\_proc\text{"}$$
$$RECEIVE \to \text{“}Power\_cons\text{"}$$
$$FORWARD \to \text{“}Mess\_forw\text{"}$$
$$FORWARD \to \text{“}Power\_cons\text{"}$$
$$ta : RECEIVE \to t_{Rec}$$
$$IDLE \to t_{Idle}$$
$$FORWARD \to t_{Forw}$$
$$DEAD \to \infty$$

2.4 Power Component Modeling

The deployment environment of a WSN is usually very harsh where power supply for each sensor node could not be continuous, and therefore power equipment with fixed power capacity is a feasible solution. Accordingly, power consumption is a critical factor about whether a WSN could work continuously and effectively, and design and working condition of each component in sensor nodes need elaborative consideration about power consumption [10]. The power model is required in WSN modeling and simulation to simulate and analyze power consumptions of sensor nodes. Figure 5 shows the structure of the power component atomic model, where the wireless communication, processor and sensor components transfer their working conditions to the power component through input ports *Trans_cons*, *Proc_cons* and *Sensor_cons*, and the power component calculates power consumption according to these working conditions; meanwhile, the power component transfers its current state to other components through output port *Power_info*.

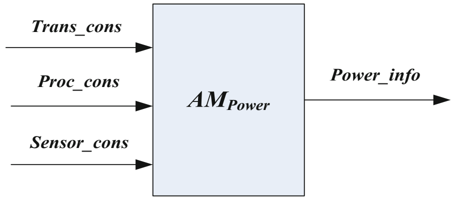

Fig. 5. Architecture of AM_{Power}

The formal description of atomic model AM_{Power} is as follows

$$
\begin{aligned}
&AM_{Power} = <X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta> \\
&X = \{\text{“}Trans\_cons\text{”}, \text{“}Proc\_cons\text{”}, \text{“}Sens\_cons\text{”}\} \\
&Y = \{\text{“}Power\_info\text{”}\} \\
&S = \{RUN, DEAD\} \\
&\delta_{ext} : RUN \times \text{“}Trans\_cons\text{”} \to RUN \\
&\quad RUN \times \text{“}Proc\_cons\text{”} \to RUN \\
&\quad RUN \times \text{“}Sens\_cons\text{”} \to RUN \\
&\delta_{int} : RUN \xrightarrow{if\ capacity > 0} RUN \\
&\quad RUN \xrightarrow{if\ capacity > 0} DEAD \\
&\lambda : RUN \xrightarrow{if\ capacity > 0} \text{“}Power\_info\text{”} \\
&ta : RUN \to t_{Run} \\
&\quad DEAD \to \infty
\end{aligned}
$$

2.5 WSN Node Modeling

One of essential advantages of DEVS is that it could achieve modular and hierarchical model design employing the I/O ports and coupling relations [6]. The component atomic models mentioned above could be coupled into sensor node model CM_{Node} using internal and external coupling relations, which is shown in Fig. 6.

The formal description of coupling model CM_{Node} is as follows

$$
\begin{aligned}
&CM_{Node} = <X, Y, D, \{M_d\}, EIC, EOC, IC, select> \\
&X = \{\text{“}Mess\_rec\text{”}, \text{“}Env\_info\text{”}\} \\
&Y = \{\text{“}Mess\_forw\text{”}\} \\
&\{M_d\} = \{Sensor, Processor, Transceiver, Power\} \\
&EIC = \{(CM_{Node}.Mess\_rec, Transceiver.Mess\_rec), \\
&\quad (CM_{Node}.Env\_info, Sensor.Env\_info)\}
\end{aligned}
$$

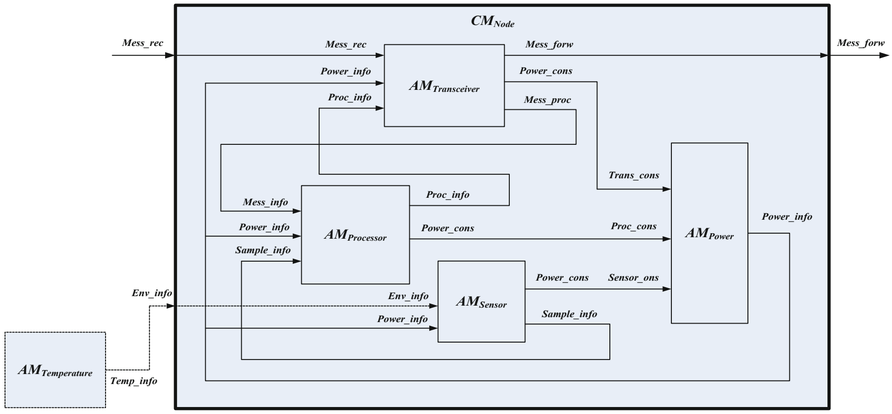

Fig. 6. Architecture of CM_{Node}

$$EOC = \{(Transceiver.Mess\_forw, CM_{Node}.Mess\_forw)\}$$
$$IC = \{(Transceiver.Power\_cons, Power.Trans\_cons)\},$$
$$(Transceiver.Mess\_proc, Processor.Mess\_info),$$
$$(Processor.Proc\_info, Transceiver.Proc\_info),$$
$$(Processor.Power\_cons, Power.Proc\_cons),$$
$$(Sensor.Sample\_info, Processor.Sample\_info),$$
$$(Sensor.Power\_cons, Power.Sens\_cons),$$
$$(Power.Power\_info, Transceiver.Power\_info),$$
$$(Power.Power\_info, Processor.Power\_info),$$
$$(Power.Power\_info, Sensor.Power\_info)\}$$
$$select(\{Sensor, Processor, Transceiver, Power\}) = Power$$

Coupling model CM_{Node} could communicates with other nodes in network topology only through ports *Mess_rec* and *Mess_forw* without extending ports as required while constructing network topology. That ensures a node coupling model is reusable in constructing WSN simulation scenarios.

2.6 WSN Modeling

A WSN can be established by setting coupling relations between WSN nodes according to network topology, where modeling of homogeneous nodes manifests the modular and reusable advantages of DEVS. The entire WSN model will be established based on the topology described in the simulation experiment in the next section.

3 Simulation Experiment and Analysis

3.1 Routing Protocol

Minimum Hop Count (MHC) routing protocol [11] is chosen as the routing protocol implementation in WSN node models in this paper. This protocol maintains a minimum hop count from current node to a sink node within each node, and nodes perform message flooding according to MHC, which achieves directional data flow to the sink node.

3.2 Simulator Introduction and Simulation Analysis

Simulation packet CD++ [12] is employed to execute WSN modeling and simulation based on DEVS in this paper. CD++ is composed of a series of models with hierarchical structure, and each model associates with a simulation entity. Atomic models are established with C++ language, and coupling models are established with the built-in formal definition language which describes the underlying member models and the coupling relations between these models.

The network topology of the simulation experiment is shown as Fig. 7. The topology of a WSN can be represented via describing coupling relations between nodes employing the formal definition language built in CD++. For establishing the integrated simulation scenario, an external temperature model $AM_{Temperature}$ is designed to provide external input for the WSN. The simulation scenario is set as single source and single sink, where node 8 is chosen as the source node to perceive temperature. The sensor component of node 8 samples the environment temperature, and when the temperature reaches the minimum measurement limit, the sensor component will transmit the real-time temperature to the processor component; when the temperature exceeds the maximum measurement limit, the sensor component will transmit the ultralimit signal to the processor component.

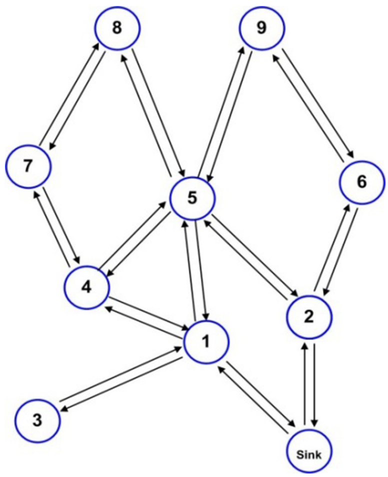

Fig. 7. Topology of wireless sensor network

Configurations for other relevant parameters in the simulation are as follows: the power capacity of each sensor node is set as 120000 units; the sensor component consumes 5 units per sampling cycle; the processor consumes 8 units per processing operation; the wireless communication component consumes 12 units per message forwarding. The total delay of message forwarding and processing is 100 ms.

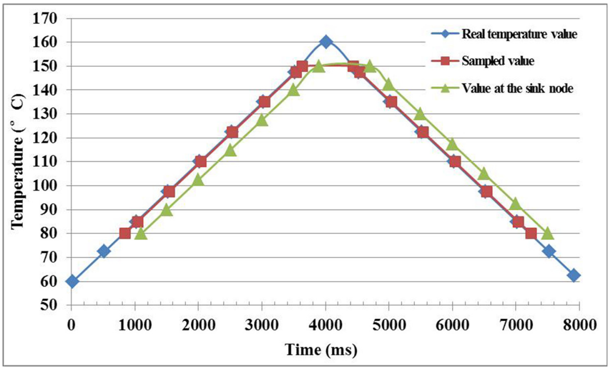

Fig. 8. Temperature variation curves during simulation

The experimental results are shown as Fig. 8, Fig. 9 and Fig. 10. Figure 8 manifests the temperature variation of the external temperature model, the sampling and processing of node 8 and the temperature data reception of the sink node. We can draw from Fig. 8 that the sink node started to receive temperature data 200 ms after node 8 forwarded data for the first time.

Figure 9 manifests the power consumption of each node. From 0 ms to about 500ms, the entire WSN was in the network establishment stage of MHC protocol. In this stage, query messages that serve for the establishment of the minimum hop gradient field were flooding, and all the nodes had obvious power consumption. From 500 ms to about 900 ms, the network establishment stage had finished, but node 8 did not perceive that the temperature reached the minimum measurement limit, and there were no messages flooding in the WSN, so powers of all nodes merely maintained basic running of sensor components and channel listening. After 900 ms, the entire WSN was in message forwarding stage: node 8 needed to process sampled value from sensor component continuously and to flood data messages; node 3, 6, 7, 9 could only receive flooding messages from a single node and no longer forwarded them, thus these nodes had the minimum power consumption; node 1 and 2 only received messages forwarded from node 5, and continued to forward them; node 4 was the only node which received messages forwarded from 2 nodes (i.e. node 1 and 5); as the unique node through which messages flooded from nodes with hop 3 to nodes with hop 1, node 5 had the maximum power consumption. Figure 10 manifests the amounts of received and forwarded messages of nodes in the WSN.

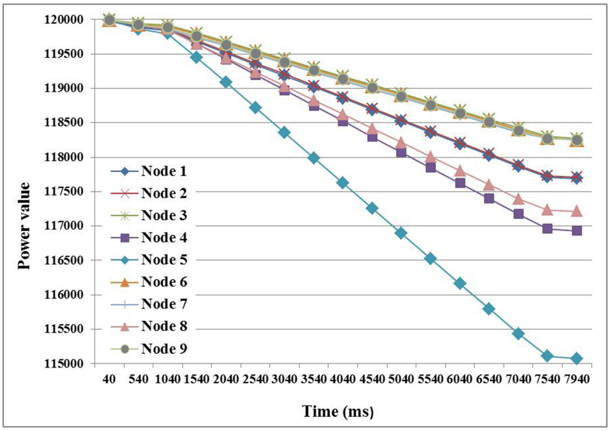

Fig. 9. Power consumption curves of each node

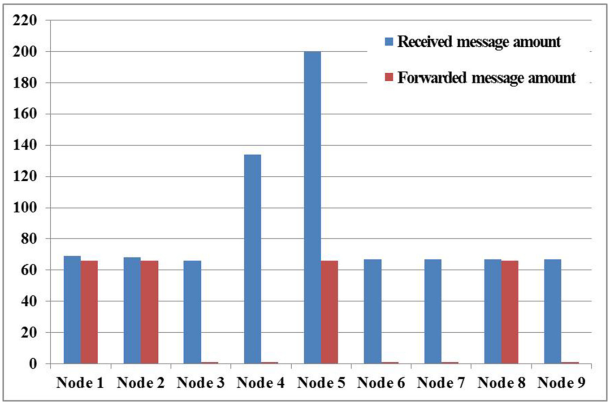

Fig. 10. Histogram of amount of received and forwarded messages of each node

4 Conclusion

A modeling and simulation method for WSN based on DEVS was proposed in this paper. The method was utilized to model sensor node components, sensor nodes, WSN and external environment, and ultimately a simulation model which can completely describe internal structure of sensor nodes, WSN topology and customized external environment was acquired. MHC was chosen as the WSN routing protocol in the simulation experiment to analyze the WSN performance, and the experiment results verified the modeling and simulation feasibility of the proposed DEVS- based method. A few nodes and a simple temperature variation were set as the simulation scenario in our first experiment; more elaborate models will be established to fulfill simulation with larger scale and more complicated scenario.

References

1. Xu, Y., Wang, X.F., He, Q.Y.: Internet of things based information support system for multi-agent decision. Ruan Jian Xue Bao **25**(10), 2325–2345 (2014)
2. Li, D.R., Yao, Y., Shao, Z.F.: Big data in smart city. Geomat. Inform. Sci. Wuhan Univ. **39**(6), 631–640 (2014)
3. Antoine-Santoni, T., Santucci, J.F., De Gentili, E., et al.: Discrete event modeling and simulation of wireless sensor network performance. Simulation **84**(2–3), 103–121 (2008)
4. Gabriel, A.: Wainer Discrete-Event Modeling and Simulation: A Practitioner's Approach. Taylor and Francis Press, UK (2009)
5. Bergero, F., Kofman, E.: A vectorial DEVS extension for large scale system modeling and parallel simulation. Simulation **90**(5), 522–546 (2014)
6. Seo, K.M., Choi, C., Kim, T.G., et al.: DEVS-based combat modeling for engagement-level simulation[J]. Simulation **90**(7), 759–781 (2014)
7. Bogado, V., Gonnet, S., Leone, H.: Modeling and simulation of software architecture in discrete event system specification for quality evaluation. Simul. Trans. Soc. Model. Simul. Int. **90**(3), 290–319 (2014)
8. Kapos, G.D., Dalakas, V., Nikolaidou, M., et al.: An integrated framework for automated simulation of SysML models using DEVS. Simul. Trans. Soc. Model. Simul. Int. **90**(717), 717–744 (2014)
9. Wang, R.Z., Shi, T.X., Jiao, W.P.: Collaborative sensing mechanism for intelligent sensors based on tuple space. J. Softw. **26**(4), 790–801 (2015)
10. Yang, C., Li, Q., Liu, J.: Dualsink-based continuous disaster tracking and early warning in power grid by wireless sensor networks. Geomat. Inf. Sci. Wuhan Univ. **38**(3), 303–306 (2013)
11. Duan, W., Qi, J., Zhao, Y., et al.: Research on minimum hop count routing protocol in wireless sensor network. Comput. Eng. Appl. **46**(22), 88–90 (2010)
12. Wainer, G.A., Tavanpour, M., Broutin, E.: Application of the DEVS and Cell-DEVS formalisms for modeling networking applications. In: Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World, pp. 2923–2934. IEEE Press (2013)