Mobile Information Systems 4 (2008) 147–161
IOS Press

Coordination protocols for a reliable sensor, actuator, and device network (SADN)

Keiji Ozaki[a,*], Tomoya Enokido[b] and Makoto Takizawa[a]
[a]*Department of Computers and Systems Engineering, Tokyo Denki University, Saitama, Japan*
[b]*Faculty of Business Administration, Rissho University, Osaki, Japan*

Abstract. A sensor, actuator, and device network (SADN) is composed of three types of nodes, which are sensor, actuator, and actuation device nodes. Sensor nodes and actuator nodes are interconnected in wireless networks as discussed in wireless sensor and actuator networks (WSANs). Actuator nodes and device nodes are interconnected in types of networks, i.e. wireless and wired network. Sensor nodes sense an physical event and send sensed values of the event to actuator nodes. An actuator node makes a decision on proper actions on receipt of sensed values and then issue the action requests to the device nodes. A device node really acts to the physical world. For example, moves a robot arms by performing the action on receipt of the action request. Messages may be lost and nodes may be faulty. Especially, messages are lost due to noise and collision in a wireless network. We propose a fully redundant model for an SADN where each of sensor, actuator, and device functions is replicated in multiple nodes and each of sensor-actuator and actuator-device communication is realized in many-to-many type of communication protocols. Even if some number of nodes are faulty, the other nodes can perform requested tasks. Here, each sensor node sends sensed values to multiple actuator nodes and each actuator node receives sensed values from multiple sensor nodes. While multiple actuator nodes communicate with multiple replica nodes of a device. Even if messages are lost and some number of nodes are faulty, device nodes can surely receive action requests required for sensed values and the actions are performed. In this paper, we discuss a type of semi-passive coordination (SPC) protocol of multiple actuator nodes for multiple sensor nodes. We discuss a type of active coordination protocol for multiple actuator nodes and multiple actuation device nodes. We evaluate the SPC protocol for the sensor-actuator coordination in terms of the number of messages exchanged among actuators.

Keywords: Sensor networks, fault-tolerant system

1. Introduction

A wireless sensor and actuator network (WSAN) [1] is composed of sensor nodes and actuator nodes interconnected in wireless channels. Many researchers discuss how sensor nodes and actuator nodes reliably and energy-efficiently communicate with each other in a wireless network [13]. In addition to the sensor-actuator communications, device nodes have to be finally performed on the physical world. There is so far no discussion on how to reliably and efficiently perform actions on device nodes like robot arms. In this paper, we newly propose a sensor, actuator, and device network (SADN) to take into account to the behaviors of the device nodes. An SADN is a collection of three types of nodes, *sensor*, *actuator*, and *device* nodes. Sensor nodes gather values like temperature about physical world and send the sensed

*Corresponding author: Keiji Ozaki, Takizawa Laboratory, Department of Computers and Systems Engineering, Tokyo Denki University, Ishizaka, Hatoyama, Saitama 350-0394, Japan. Tel.: +49 296 2911 ext. 2544; Fax: +49 296 6185; E-mail: kei@ takilab.k.dendai.ac.jp.

1574-017X/08/$17.00 © 2008 – IOS Press and the authors. All rights reserved

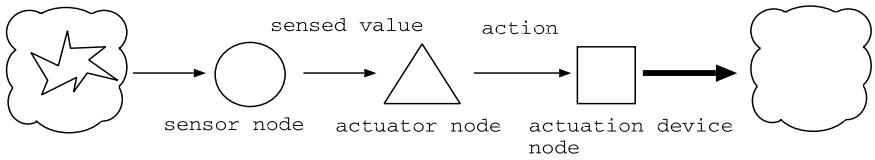


Fig. 1. Sensor, actuator, and device nodes.

values to actuator nodes. Sensor nodes are low-cost, low-power devices which are equipped with limited battery energy, computation, and wireless communication capabilities like MICA2 MOTE [17]. On the other hand, an actuator node is equipped with more powerful energy. An actuator node collects sensed values from sensors. Then, the actuator node makes a decision on actions for the sensed values and issues the actions to device nodes. On receipt of an action request from an actuator node, a device node performs the action on the physical world, e.g. moves an arm of a robot and ventilates cool air. Sensor nodes and actuator nodes are interconnected in a wireless network as discussed in the WSAN. On the other hand, actuator nodes and device nodes are interconnected in different types of networks, i.e. not only wireless networks but also wired networks. Wireless networks are less reliable, i.e. messages may be lost due to noise and collision although wired networks can be assumed to be reliable. There are many discussions [5,12] on how to reliably transmit messages among sensor nodes in the presence of message loss so as to reduce the energy consumption.

Nodes might be faulty and messages might be lost in networks. The SADN has to be reliable in the presence of faults of nodes and message loss in networks. In this paper, we propose a fault-tolerant SADN model where every type of node is replicated and nodes communicate with other nodes in a many-to-many communication. First, if an event occurs in a physical world, multiple sensors sense the event. Each of the sensor nodes sends a sensed value to multiple actuator nodes while each actuator node receives sensed values from multiple sensor nodes. Secondly, an actuator node sends an action request message to multiple device nodes while each device node receives action request messages from multiple actuator nodes. Even if some number of nodes are faulty and messages are lost, an action required for an event has to be performed on a device node. In this paper, we discuss how to realize the reliable sensor-actuator and actuator-device communications. We take a semi-passive type of coordination (SPC) protocol to realize the reliable many-to-many communication among sensor nodes, actuator nodes, and device nodes. Here, there is one primary actuator node and the others are backup actuator nodes. The primary actuator node makes a decision on a sensed value from multiple sensor nodes. Then, the primary actuator node sends the decision to multiple device nodes.

In Section 2, we present a system model of SADN. In Section 3, we discuss the Multi-actuator/Multi-sensor (MAMS) model. In Section 4, we discuss types of coordination protocols for multiple actuators. In Section 5, we evaluate the semi-passive coordination (SPC) protocol for the sensor-actuator coordination in terms of number of messages exchanged among multiple actuators.

2. A sensor, actuator, and device network (SADN) model

2.1. Nodes

A sensor, actuator, and device network (SADN) is composed of three types of nodes, *sensor* nodes, *actuator* nodes, and actuation device nodes. A sensor node gathers values about the physical world

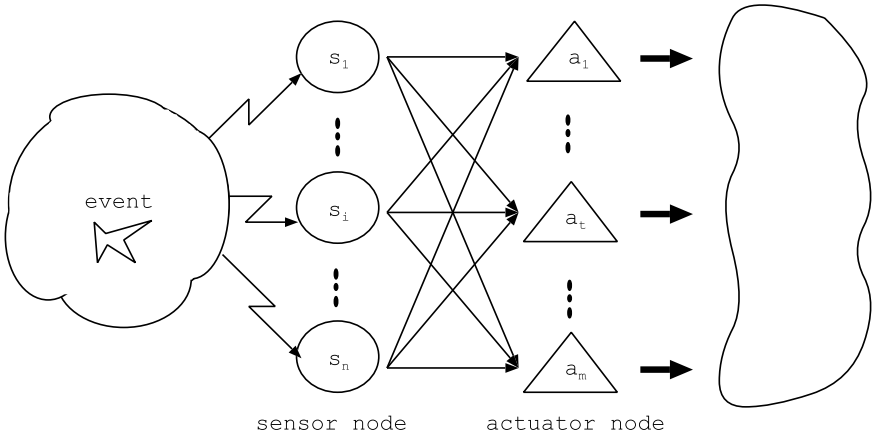


Fig. 2. Sensor-actuator communication.

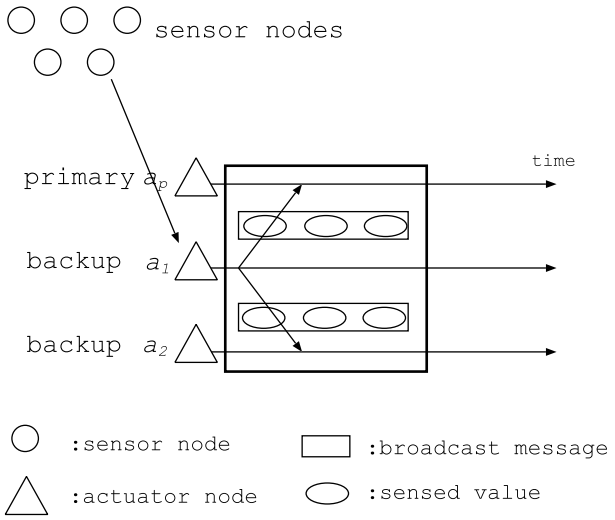


Fig. 3. Centralized forwarding phase.

like temperature and sends the sensed values to actuator nodes. A sensor node is composed of sensors, wireless communication device, processing facilities, and battery like MICA2 MOTE [17]. A sensor node is a low-cost, low-power device with limited battery energy. An actuator node decides on what actions to be performed for sensed values and then sends the action commands to the device node. An actuator node is equipped with more rich energy than a sensor node. On receipt of an action command from an actuator node, the action is performed on the device node. Nodes are interconnected in types of networks, wireless and wired networks. Sensor nodes and actuator nodes are interconnected in a wireless network. On the other hand, actuator nodes and actuation device nodes are interconnected in different types of networks, wireless and wired networks.

Phenomena in the physical world is changed on occurrence of an event. If an event occurs in some location, sensor nodes in some distance from the location gather values of some attributes of the event. There are the following types of sensors:

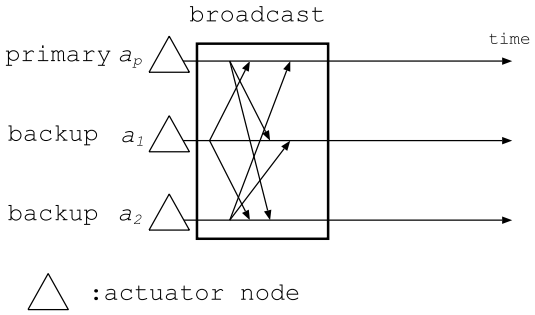


Fig. 4. Distributed forwarding phase.

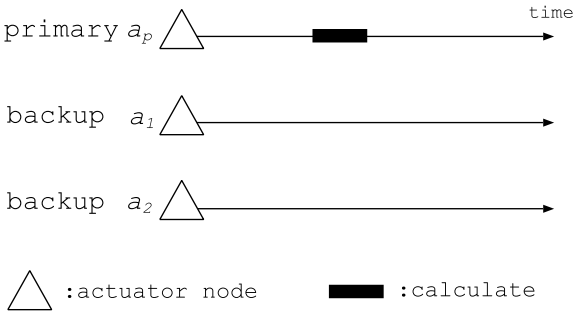


Fig. 5. Centralized decision phase.

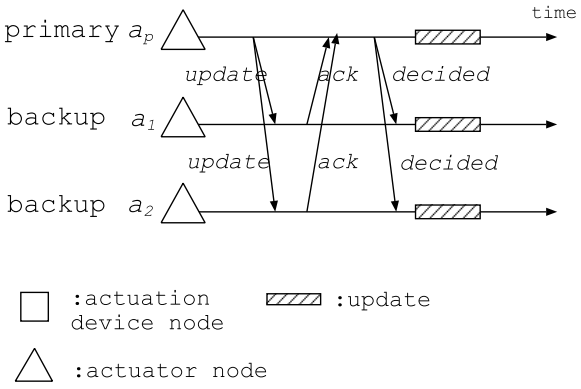


Fig. 6. Centralized update phase.

1. *Discrete* sensors:
2. *Continuous* sensors:

A *discrete* sensor takes discrete values like ON and OFF, open and close. A *continuous* sensor takes a continuous value like *temperature*. Values sensed by different sensor nodes might be different. Especially, continuous sensor nodes in nature take different values even if the difference between the values is small.

An actuator node is a process which receives sensed values from sensors in a wireless channel and

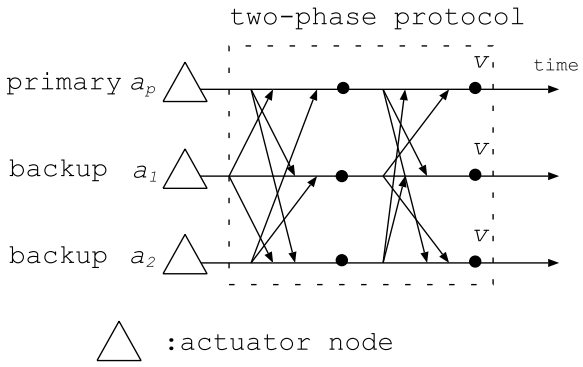


Fig. 7. Distributed update phase.

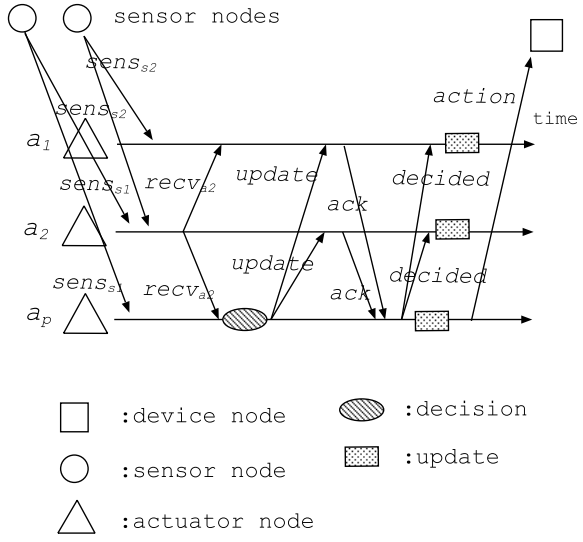


Fig. 8. Semi-passive coordination (SPC).

generates action commands to be performed on actuation device nodes from sensed values collected from sensors. By using a kind of decision function, the actuator node takes a value from the collection of the sensed values e.g. average value and median value. The actuator node generates an action command for the value. Then, the actuator node controls actuation device nodes by issuing the action command. Device nodes act to the physical world like robot arms on receipt of a action command from an actuator node. A device node is modeled to be a object [8] in this paper. An object is an encapsulation of state and methods for manipulating the state. Actuation of a device node is modeled to be execution of a method on the device object. An actuator node issues a method to a device object. The authors discuss types of actuation devices, serial and parallel ones, stateless and stateful ones [15]. In this paper, we assume every actuation device is a serial, stateless type, i.e. at most one method is performed on a device node at a time and a device node does not have memory to store methods performed and the previous state.

Multiple actuator nodes may issue methods to a device node. Even if each of the actuator nodes sends a same method, the method has to be performed only once on the device node. In addition, if each of the actuator nodes sends a different method to a device node but the method conflicts with methods issued

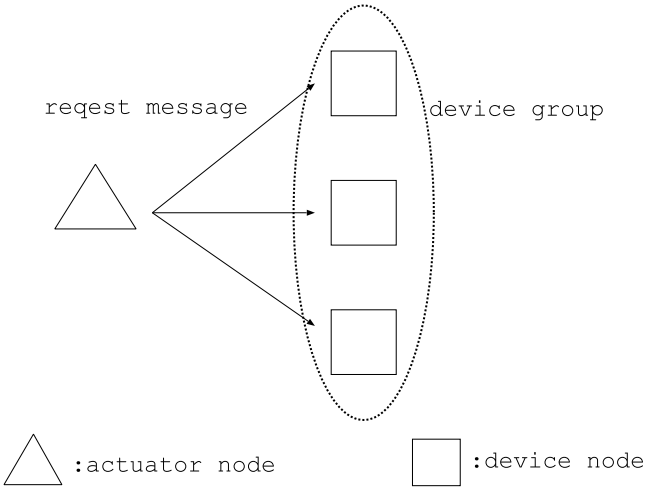


Fig. 9. Broadcast of request message to the device group.

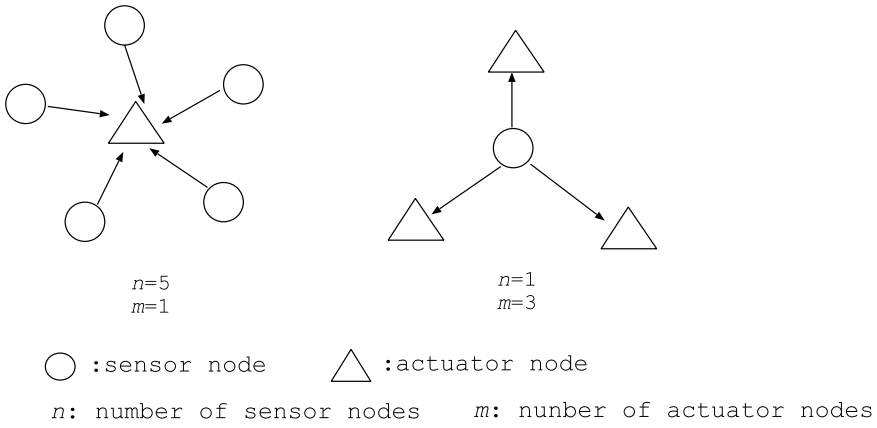


Fig. 10. Actuator and sensor nodes.

by other actuator nodes, the methods have to be serialized on the device node. The authors discuss how to resolve redundant execution of a method and serialize conflicting methods from multiple actuator nodes [15].

A sensor node sends sensed values to actuator nodes in a wireless channel. A message sent by a sensor node can be received by nodes depending on the radio field intensity. A sensor node s_k cannot deliver messages to distant nodes due to less energy supply. MICA2 MOTE sensor node can deliver message to node in 4 5m. On the other hand, an actuator node can deliver messages to distant nodes. In addition, since sensor nodes use same channel messages are lost due to collision if multiple sensor nodes send simultaneously the messages. Thus messages are lost due to noise and collision. In order to resolve the collision in the channel, some synchronization protocols [9–11] are used. For example, the CSMA synchronization protocol is used in the sensor node MICA2 with TinyOS [18] operating system because of the simplicity. In sensor networks, more messages are lost due to noise than collision [13].

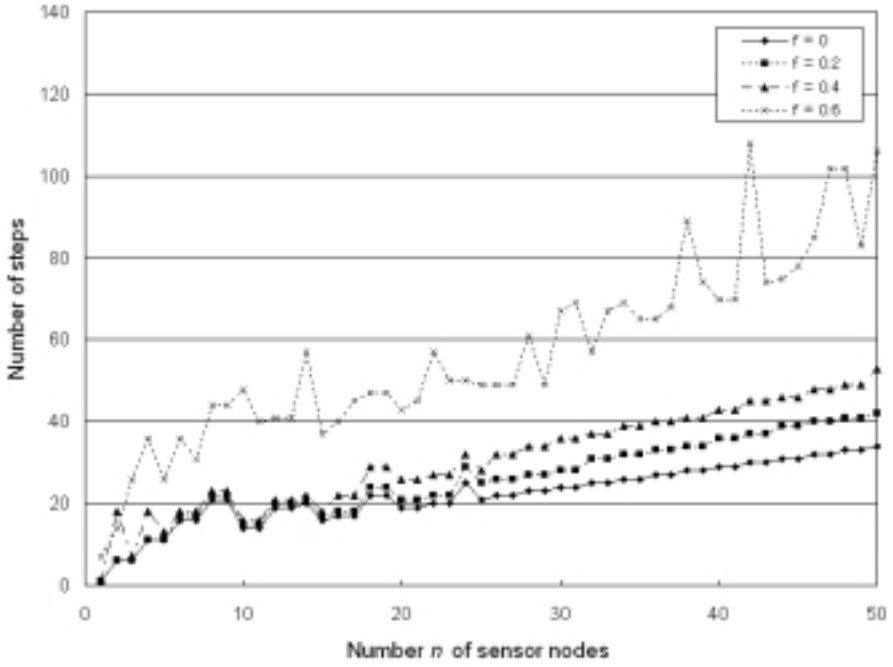


Fig. 11. Number of steps in the forwarding phase.

2.2. *Multi-sensor/multi-actuator/multi-device* (M^3) *model*

In this paper, we take the *multi-sensor/multi-actuator/multi-device* (M^3) model to realize the reliable sensor-actuator and actuator-device communication in a sensor, actuator, and device network (SADN). Here, each sensor node sends a sensed value to multiple actuator nodes and each of the actuator nodes receives sensed values from multiple sensor nodes. Each actuator node can receive a sensed value even if some sensor node is faulty and messages are lost. A sensed value can be delivered to at least one actuator node even if some number of actuator nodes are faulty. An actuator node is replicated. Each actuator node sends an action command message to multiple replica nodes of the device while each replica node receives action commands from multiple actuator nodes. Here, a device node means a replica of a device. Even if an actuation device is faulty, an action can be performed on another device node.

In sensor-actuator communications, each actuator node a_t makes the following decisions:

1. An actuator node a_t makes a decision on a value v from sensed values received from child sensor nodes.
2. An actuator node a_t makes a decision on which method to be performed on which actuation device for the sensed value v.

There are the discrete and continuous types of sensor nodes. If the sensors discrete ones, every actuator node has to make an agreement on one *discrete* value in the domain. Here, a majority value of the sensed values can be taken. On the other hand, a *continuous* attribute takes a continuous value like temperature. Even if a pair of sensor nodes s_i and s_j are proper, s_i and s_j may send different values to actuators. Here, some statistics value like average of the sensed values is taken. Let V be a set of sensed values. The function *filter* removes values which are out of two-sided $\alpha\%$ confidence interval in the set V. α is

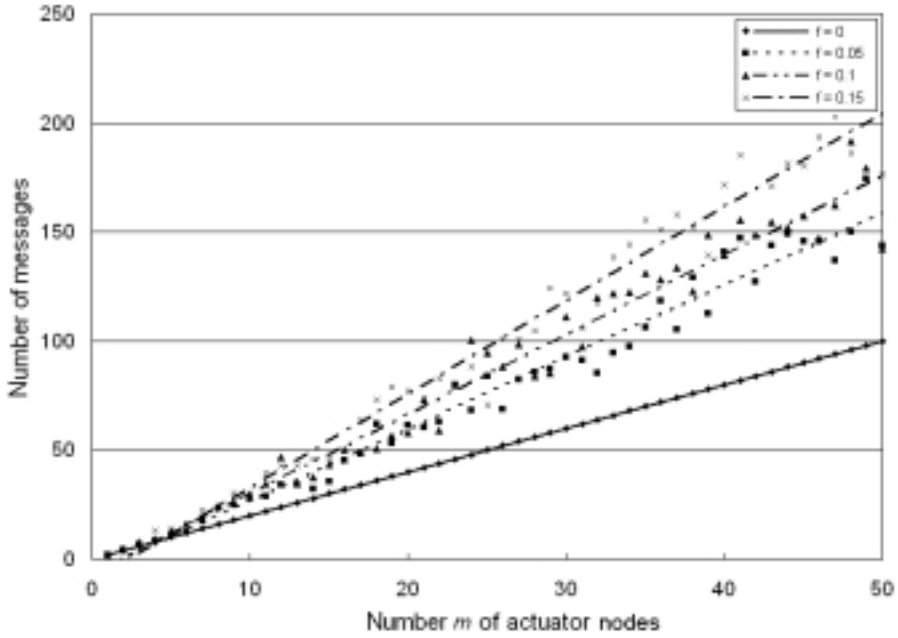


Fig. 12. Number of messages in the update phase.

assumed to be 5[%] in this paper. Then, an average value v is taken by a function *average*. That is, a value $v = average\ (filter\ (V))$ is taken.

A device node is replicated so that an action generated by an actuator node for a sensed value can be performed on one replica of the device node even if some number of the replicas are faulty. An action command message of a method *op* is sent to multiple device nodes. If the method *op* is idempotent like just reference, the method *op* can be performed on multiple device nodes. Otherwise, the method *op* is required to be performed only to one device node.

3. Coordination among sensor nodes and actuator nodes

3.1. Types of protocols

We discuss how multiple actuator nodes $a_1, \cdots, a_m$ ($m \geqslant 1$) cooperate to make a decision on a value v for a collection of sensed values sent from sensor nodes (Fig. 2). *Active* [19], *passive* [19], *semi-passive* [4], and *semi-active* [19] replications are so far discussed to coordinate multiple replicas of a process. In the active coordination protocol, every actuator node receives the same sensed values from sensor nodes and performs the same methods on device nodes in the same order. Redundant executions of a method on a device node [15] have to be resolved. Here, some synchronization mechanisms like timestamp ordering (TO) schedulers [2] and locking protocols [7] have to be supported. In addition, every actuator node is required to be deterministic.

In the passive, semi-passive, and semi-active protocols, there is one *primary* actuator node a_1 which plays a role of a coordinator of all the other *backup* actuator nodes $a_2, \cdots, a_n$. On receipt of a sensed value from a sensor node s_k, a backup actuator node a_t forwards the value to the primary actuator node a_1

since the primary actuator node a_1 may not receive the value. Then, the primary actuator node a_1 makes a decision on a value from the sensed values collected by the actuator nodes $a_2, \cdots, a_m$. The primary actuator node a_1 performs methods for the sensed value on actuation devices. There is no redundant execution of a method. If primary actuator node a_1 is faulty, one of the backup nodes, takes over the primary one.

In the passive and semi-passive coordination ways, every backup actuator node neither makes a decision on methods nor performs methods. Since every backup actuator node does not do the computation, the battery energy is not consumed in every backup actuator node even if each actuator node is equipped with the powerful battery. In the passive coordination ways, every backup actuator node does not receive the same messages as the primary actuator node. The primary actuator node takes a checkpoint where the local state is saved in a stable memory and sends the state saved at the checkpoint to every backup actuator node. If the primary actuator node a_1 gets faulty, one backup actuator node a_k restarts from the checkpoint. Messages which the primary actuator node a_1 receives after taking the checkpoint are lost. In the semi-passive coordination way, each backup actuator node receives the same messages in the same order as the primary actuator node a_1. Hence, if the primary actuator node a_1 gets faulty, a backup actuator node a_k can catch up with the state of the faulty primary actuator node a_1 by doing the computation for the messages received. On the other hand, each backup actuator node makes a decision on what methods to be performed in what order but does not perform the methods in the semi-active coordination way.

In the semi-active coordination protocol, every backup actuator node receives the same sensed values and make the same decisions on methods. However, every backup actuator node does not perform the methods while only the primary actuator node performs the methods. Here, any backup actuator node takes over the primary actuator node as soon as the primary actuator node gets faulty. Every actuator node is required to be deterministic.

3.2. Protocol modules

We discuss a semi-passive coordination (SPC) protocol for multiple actuator nodes. The SPC protocol is composed of four phases, which are forwarding, decision, update, and action phase.

3.2.1. Forwarding phase

A sensor node s_k cannot deliver messages to every actuator node as presented in the system model. An actuator node a_t may not receive a sensed value from some sensor node s_k. Some actuator node is required to receive sensed values from more than some number h of sensor nodes. For example, $h \geqslant n - f$ where f is the maximum number of faulty sensor nodes for discrete sensor nodes. Hence, on receipt of a sensed value v_k from a sensor node s_k, an actuator node a_t sends a message m with a pair $\langle s_k, v_k \rangle$ of the sensed value v_k and the identifier of the sensor node s_k to another actuator node a_u. There are two approaches to forwarding sensed values to actuator nodes, *centralized* and *distributed* ones. In the centralized approach, every actuator node which receives sensed values forwards the values to one primary actuator node, say a_p. The primary actuator node a_p receives a pair $\langle v_k, s_k \rangle$ of a sensed value v_k and a sensor node s_k from every backup actuator nodes. Here, the pair $\langle v_k, s_k \rangle$ is stored in the receipt buffer V_p. If the primary actuator node a_p receives messages from more than na ($\leqslant l$) sensor nodes, the primary actuator node a_p broadcasts the set V_p of the sensed values to all the backup actuator nodes. Here, f is a maximum number of sensor nodes which may stop by fault. Then, $na = (l - f)/m$.

In the distributed approach, an actuator node delivers sensed values to every actuator node. On receipt of a sensed value from a sensor node, an actuator node a_t broadcasts the value in the same way as the

centralized approach. Each actuator node a_t receives sensed values from every other actuator node. If an actuator node a_t receives sensed values from more than na sensor nodes, the forwarding phase terminates. Here, atomic group communication protocols [3,14] are used to broadcast sensed values to all the actuator nodes.

3.2.2. Decision phase

In the forwarding phase, some actuator node receives sensed values in the receipt buffer V_t. Then, an actuator node makes a decision on which method to be performed on which actuation device object. There are two types of decision phase, *centralized* and *distributed* one with respect to which actuator node makes a decision. In the centralized decision, only the primary actuator node a_p makes a decision on a value v from the sensed values in the buffer V_p received. If values are ones of a discrete attribute, the primary actuator node a_p takes a majority value in the value set V_p. On the other hand, the primary actuator node a_p obtains a value v = *average*(*filter*(V_p)) with the function *filter*. This is the decision phase (Fig. 5).

In the distributed approach, every actuator node makes the same decision as the primary actuator node in the centralized approach. Here, every actuator node is required to hold the same sensed values from sensor nodes.

3.2.3. Update phase

In the decision phase, only the primary actuator node a_p obtains the value v from sensed values. The primary actuator node a_p sends an *update* message with the internal state, i.e. the value v obtained from sensed values to all the backup actuator nodes $a_1, \cdots, a_m$ (Fig. 6). On receipt of the *update* message with a value v from the primary actuator node a_p, a backup actuator node a_t stores the value v in its local stable storage and sends an acknowledgment *ack* message to the primary actuator node a_p. If the primary actuator node a_p receives *ack* messages from more than half of the backup actuator nodes, the primary actuator node a_p sends a *decided* message to all the backup actuator nodes. On receipt of a *decided* message from the primary actuator node a_p, each backup actuator node a_t updates the state by using value stored in the local storage. The primary actuator node a_p takes a method *op* from the value v and performs the method *op* on the actuation device. Here, the *decided* message carries the value v as well as the *update* message. Even if a backup actuator node a_t could not receive the *update* message, the backup actuator node a_t obtains the value v from the *decided* message only if the backup actuator node a_t receives a *decided* message. On receipt of a *decided* message from the primary actuator node a_p, the backup actuator node a_t sends an *ok* message to the primary actuator node a_p.

In the distributed decision phase, every actuator node a_t makes a decision on a value v_t. Every actuator node makes an agreement on an value. For example, by using the two-phase commitment (2PC) protocol [16], every actuator node takes the same value v.

3.2.4. Action phase

An actuator node performs a method *op* on an actuation device. There are two approaches, *centralized* and *distributed* one. In the *centralized* action phase, only one primary actuator node performs the method *op* on actuation devices and the other backup actuator nodes do not perform the method *op*.

On the other hand, multiple actuator nodes perform the method *op* in the *distributed* approach. An actuation device receives a request of method *op* from multiple actuation nodes. Here, we have to resolve the redundant executions of the method [15].

Table 1
Table of coordination phases

	Forwarding	Decision	Update	Action
Active (AC)	D	D	D	D
Semi-active (SA)	D	D	C	C
Semi-passive (SP)	D	C	C	C
Passive (PS)	C	C	C	C
D: distributed		C: centralized		

3.3. Coordination protocols

Table 1 summarizes the coordination protocols, active (AC), semi-active (SA), passive (PS), and semi-passive (SP) coordinations, which are composed of types of phases. Each phase is centralized (C) or distributed (D). For example, the semi-passive (SP) coordination is composed of the distributed forwarding phase, centralized decision phase, and centralized update phase.

Figure 8 shows an example of the semi-passive coordination (SPC) among a primary actuator node a_p and a pair of backup actuator nodes a_1 and a_2. Here, no actuator node is faulty. Sensor nodes send sensed values to actuator nodes. Each backup actuator node a_t broadcasts a set V_t of sensed values which the actuator node a_t has received from sensor nodes ($t = 1, 2$). The primary actuator node a_p receives the value sets V_1 and V_2 from the backup actuator nodes a_1 and a_2, respectively. Then, the primary actuator node a_p calculates a value v from the sets of values V_1, V_2 and V_p. The primary actuator node a_p broadcasts the value v to every backup actuator node a_t and the state of the backup actuator node a_t is updated. The primary actuator node a_p performs a method op obtained for the value v on the actuation device.

4. Coordination among actuator nodes and device nodes

Next, we discuss how multiple actuator nodes cooperate with multiple replica nodes of a device node. Here, an action op is decided to be performed on a device node as discussed in the preceding section. Suppose there are multiple actuator nodes $a_1, \cdots, a_m$ ($m \geqslant 1$) and multiple device nodes $d_1, \cdots, d_l$ ($l \geqslant 1$). In this paper, we assume one actuator node a_t is primary and the other actuator nodes are backup ones as discussed in the preceding section. The primary actuator node a_t issues an action command to device nodes.

A device node performs an action command of a method op which is carried in the request message from a primary actuator node a_t. The device nodes return the acknowledgment message to the primary actuator node a_t on completion of the method op. We assume each device node is a replica of an actuation device and each device node is a sequential, stateless device. That is, at most one action method can be performed on each device node at a time. Each action issued by an actuator node should be performed on at least one device node even if some number of device nodes are faulty. For example, there are multiple air-conditioners in a room. An actuator node a_t issues an action command *cool* to lower the temperature in the room. Here, the actuator node a_t can issue the *cool* command to every device node. As long as at least one air-conditioner device node is operational, the room can be make cooler even if it takes a longer time to lower the temperature. In another example, a robot is equipped with a device for launching a missile. An actuator node a_t makes a decision on launching a missile to shoot an intruding aircraft. Then, the actuator node a_t issues a *launch* command to every missile device node. Here, one missile should be launched. However, more than one missile are launched if multiple device nodes receive the launch

command. The action method should be performed only to one device node. Thus, there is the maximum number M_{op} ($\geqslant 1$) of device nodes on which an action method op can be performed. An action method op should be performed at least one device node and at most M_{op} device nodes. If $M_{op} = 1$, a method op is required to be performed by only one device node i.e. no redundant execution of the method op is allowed. In the missile device nodes, M_{op} should be one. In the air-conditioner devices, M_{op} is the total number l of the device nodes.

Let D be a set of device nodes $d_1, \cdots, d_l$ ($l \geqslant 1$). A primary actuator node a_t in the SPC protocol communicates with the device nodes in the set D as follows:

1. The primary actuator node a_t selects a subset $D_{op} = \{d_{t1}, \cdots, d_{tM}\}$ ($\subseteq D$) of device nodes where $M = M_p$ if $|D| \geqslant M_p$, else $M = |D|$. The primary actuator node a_t sends a request message of the action method op to every device node in the subset D_{op}.
2. If each device node d_x receives the request message of a method op from the primary actuator node a_t, the device node d_x performs the method op. On completion of the method op, the device node d_x sends an acknowledgment message to the primary actuator node a_t.
3. If the primary actuator node a_t does not receives an acknowledgment message from any actuation device node, $D := D - D_{op}$ and go to step 1.
4. If the primary actuator node a_t receives an acknowledgment message from some device node in D_{op}, the primary actuator node a_t informs the other actuator nodes of it. Then, the primary actuator node a_t terminates performing the method op.

A set of device nodes $D_{op} = \{d_{t1}, \cdots, d_{tM}\}$ supporting a same method op is a *device group* for the method op (Fig. 9). A request message *req* sent by an actuator node a_t includes the following information:

op = action method.
dest = destination device group D_{op} to deliver this request message *req*.
M_{op} = maximum number of device nodes where the action op can be performed.

An actuator node a_t supports the following functions:

$dest := select(D, M)$: a set *dest* including the number M of actuator nodes are selected in the set D of actuation device nodes, where $|dest| = M_p$ if $|D| \geqslant M$, otherwise $dest = D$.
broadcast(*request_message*, *dest*): a request message *request_message* is broadcast to all of the device nodes in a set *dest* of device nodes.
inform(m): a message m is sent to every actuator node. On receipt of message m, each actuator node records it in the log.

A primary actuator node a_t communicates with the device nodes to perform a method op on device nodes in a set D by the following function:

Actuate(D, op)
  dest := *select* (D, M_{op});
  broadcast (*op*, *dest*);
  if a positive acknowledgment Ack is received from at least one device node in D,
  then
    inform(*success*);
    return(*success*);
  else /* no acknowledgment */

$D := D - dest$;
if $D \neq \phi$,
then /* send *op* to device nodes */
return(*Actuate*(D, *op*));
else
inform(*failed*);
return(*failed*);

A primary actuator node a_t performs the function *Actuate*(D_{op}, *op*) to perform a method *op* on device nodes in D_{op}.

A primary actuator node a_t might be faulty. A coordination protocol for detecting and taking over the faulty primary actuator node is required to be supported. It depends on the maximum number M_{op} how the method *op* to be performed by a backup actuator node. We discuss these issues in another paper.

5. Evaluation

We evaluate the semi-passive coordination protocol (SPC) protocol for sensor-actuator coordination discussed in this paper. Let n be a number of sensor nodes and m be a number of actuator nodes (Fig. 10). We assume that no failure occurs in any node but messages are lost in a network.

Every node is identified by a unique identifier. We assume an actuator node with the smallest identifier is taken as primary node. First, we evaluate the forwarding phase. Sensor nodes sense events in the physical world and send sensed values to actuator nodes. If each actuator node receives a sensed value from a sensor node, the actuator node broadcasts the value to all the actuators nodes. This is one step. If the primary actuator node a_1 receives values sent by more than half ($> m/2$) of the sensor nodes, the forwarding phase ends. Even if the primary actuator node a_1 does not directly receive a value from a sensor node, the primary actuator a_1 node can receive the value forwarded by a backup actuator node. Every sensor node sends a value to actuator nodes using a logical sensor-actuator channel. In the channel, every actuator node loses a value with probability f. Every actuator node has a first-in first-out (FIFO) buffer to store sensed values from sensor nodes and forwarded by other actuator nodes. If an actuator node receives more than number n/m of sensed values, the actuator node forwards the values to other actuator nodes using a logical actuator-actuator channel. In this channel, no message is lost. Each time a sensor node or actuator node sends a message, the number of steps is incremented by one. We measure the number of steps in the presence of message fault with probability f as shown in Fig. 11. We assume there are five actuator nodes, $a_1, \cdots, a_5$ in the evaluation. Here, a_1 is the primary actuator node. The horizontal axis shows the number n of sensor nodes. The vertical axis indicates the number of steps. The more is the number n of sensor nodes and the larger is the fault probability f, the longer time it takes in the forwarding phase.

Next, we evaluate the update phase of the SPC protocol. The primary actuator node a_1 broadcasts an *update* message with a value v to every backup actuator node. On receipt of the *update* message from the primary actuator node a_1, each backup actuator node a_t sends an ack message to the primary actuator node a_1. The primary actuator node a_1 broadcasts a *decided* message to every backup actuator node if the primary node a_1 receives *ack* messages from more than half of the backup actuator nodes. Then, the primary actuator node a_1 waits for an *ok* message from every backup actuator node. If the primary actuator node a_1 does not receive an ok message from some backup actuator node, the primary actuator node a_1 broadcasts a *decided* message again. Until the primary actuator node a_1 receives an *ok* message

from every backup actuator node, the primary actuator node a_1 broadcasts a *decided* message. If the primary actuator node a_1 receives *ok* messages from every backup actuator node, the update phase of the SPC protocol finishes. Figure 12 shows the number of messages exchanged among actuator nodes for the number m of actuator nodes $a_1, \cdots, a_m$. Here, if an actuator node broadcasts a message to $(m-1)$ actuator nodes, the number of messages transmitted is counted to be one since the message is sent in a broadcast channel. "$f = 0$" shows that no message loss occurs in every actuator node. The number of messages transmitted is shown for the smaller lost probability $f = 0.05, 0.1$, and 0.15 in Fig. 12. For f = 0.05, 0.1, and 0.15, the number of messages transmitted is increased about 1.57, 1.76, and 2.05 times larger than the number of messages for $f = 0$, respectively.

6. Concluding remarks

In this paper, we discuss the reliable model of the sensor, actuator, and device network (SADN) which is composed of three types of nodes, sensor, actuator, and device nodes in the presence of node and network faults. We proposed the multi-sensor/multi-actuator/multi-device (M^3) model for making the SADN more reliable where every type of node, i.e. sensor, actuator, and device node is replicated in multiple nodes. Here, a sensor node sends a sensed value to multiple actuator nodes and each actuator node receives sensed values from multiple sensor nodes. An actuator node issues an action method command to multiple replica nodes of an actuation device and each device node receives action method commands from multiple actuator nodes. We discussed protocols for sensor-actuator coordination and actuator-device coordination. In the sensor-actuator coordination, we discuss the semi-passive coordination (SPC) protocol where there is one primary actuator node and the others are backup one. Every backup actuator node receives sensed values from sensor nodes. In the actuator-device coordination, we also discussed the semi-active coordination (SPC) protocol. We evaluated the forwarding and update phase of the SPC protocol in the presence of faulty backup actuator nodes.

References

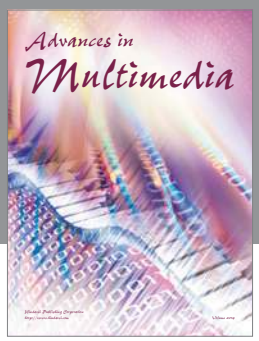
[1] I.F. Akyildiz and I.H. Kasimoglu, Wireless sensor and actor networks: research challenges, *Ad Hoc Networks journal* (*Elsevier*) **2** (2004), 351–367.
[2] P.A. Bernstein, V. Hadzilacos and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley, 1987.
[3] K. Birman, A. schiper and P. Stephenson, Lightweight causal and atomic group multicast, *ACM Transactions on Computer Systems* (*TOCS*) **9**(3) (1991), 272–314.
[4] X. Defago, A. Schiper and N. Sergent, Semi-passive replication, in: *Proceedings of the 17th IEEE Symposium on Reliable Distributed Systems* (*SRDS-17*), 1998, pp. 43–50.
[5] A. Durresi and V. Paruchuri, Geometric broadcast protocol for heterogeneous sensor networks, *Proc. of 19th IEEE International Conf. on Advanced Information Networking and Applications* (*AINA2005*) **1** (2005), 343–348.
[6] A. Durresi, V. K. Paruchuri, S. S. Iyengar and R. Kannan, Optimized broadcast protocol for sensor networks, *IEEE Transactions on Computers* **54**(8) (2005), 1013–1024.
[7] K.P. Eswaren, J.N. Gray, R. Lode and I.L. Traiger, The Notion of Consistency and Predicate Locks in Database Systems, *Communications of the ACM* **19**(11) (1976), 624–637.
[8] A. Goldberg and D. Robson, *Smalltalk-80: The interactive programming environment*, Addison-Wesley, 1983.
[9] IEEE, *IEEE Std 802.11b – Wireless LAN Medium Access Control* (*MAC*) *and Physical layer* (*PHY*) *Specifications: High Speed Physical Layer* (*PHY*) *in the 2.4 GHz Band*. IEEE, 1999.
[10] IEEE, *Carrier Sense Multiple Access with Collision Detection* (*CSMA/CD*) *Access Method and Physical Layer Specifications*, IEEE, 2000.
[11] S.S. Lam, A carrier sense multiple access protocol for local networks. Technical report, Austin, TX, USA, 1979.

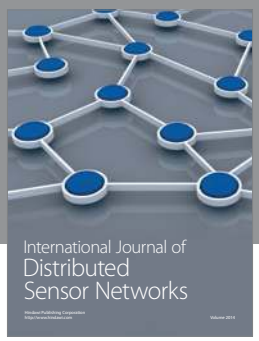
[12] T. Melodia, D. Pompili, V.C. Gungor and I.F. Akyildiz, A distributed coordination framework for wireless sensor and actor networks, *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing* **1** (2005), 99–110.
[13] K. Morita, K. Watanabe, N. Hayashibara, T. Enokido and M. Takizawa, Evaluation of reliable data transmission protocol in wireless sensor-actuator network, in: *Proceedings of the 3rd IEEE International Workshop on Heterogeneous Wireless Sensor Networks* (*HWISE-2007*), 2007, pp. 713–718.
[14] A. Nakamura and M. Takizawa, Causally ordering broadcast protocol, in: *Proceedings of the 14th IEEE International Conference on Distributed Computing Systems* (*ICDCS-14*), 1994, pp. 48–55.
[15] K. Ozaki, K. Watanabe, S. Itaya, N. Hayashibara, T. Enokido and M. Takizawa, A fault-tolerant model of wireless sensor-actor network, In *Proceedings of the 9th International Symposium on Object and Component-Oriented Read-Time Distributed Computing* (*ISORC 2006*), 2006, pp. 186–193.
[16] D. Skeen, Nonblocking Commitment Protocols, in: *Proc. of ACM SIGMOD*, 1981, pp. 133–147.
[17] C. Technology, http://www.xbow.com.
[18] TinyOS, http://www.tinyos.net.
[19] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme and G. Alonso, Understanding replication in databases and distributed systems, in: *Proceedings of 20th International Conference on Distributed Computing Systems* (*ICDCS'2000*), Apr. 2000, pp. 264–274.

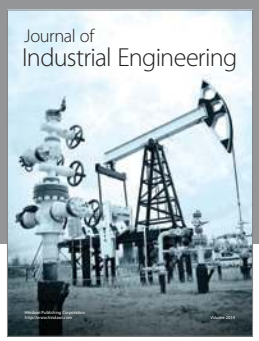
Keiji Ozaki was born in Kagawa, Japan, 1984. He received B.E. degrees in Computers and Systems Engineering from Tokyo Denki University, Japan in 2006. He is now working towards his M.E. degree with Tokyo Denki University. His research interests include wireless sensor networks, and fault-tolerant systems. He is a student member of IEEE.

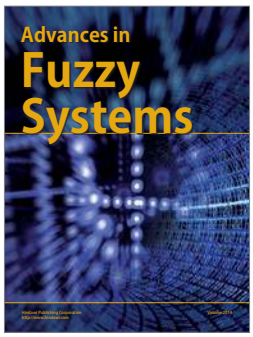
Tomoya Enokido was born in Japan, 1974. He received his B.E., M.E. and D.E. degrees in Computers and Systems Engineering from Tokyo Denki University in 1997, 1999, and 2003, respectively. After working for NTT Data Corporation and Tokyo Denki University, he is now a lecturer in Faculty of Business Administration, Rissho University since 2005. He won the excellent paper awards of IEEE AINA 2004 and 2005. His research interests include group communication, distributed objects, fault-tolerant systems, and distributed transaction management. He is a member of IEEE and IPSJ.

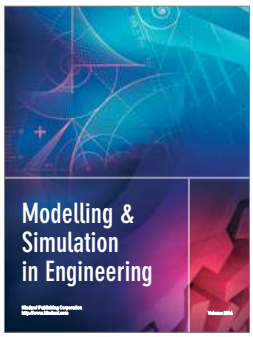
Makoto Takizawa was born in Tokyo, Japan, 1950. He is now a professor in the Department of Computers and Systems Engineering, Tokyo Denki University. He is a member of Board of Governors (2003-) and a Golden Core member of IEEE Computer Society since 2004. He is a fellow of Information Processing Society of Japan (IPSJ). He received his B.E. and M.E. in Applied Physics and D.E. in Computer Science from Tohoku University, Japan. He was a General Co-Chair (2002) and a Program Co-Chair (1998) of IEEE ICDCS. He is the founder of IEEE AINA. His research interests include group communications, fault-tolerant systems, and information security.

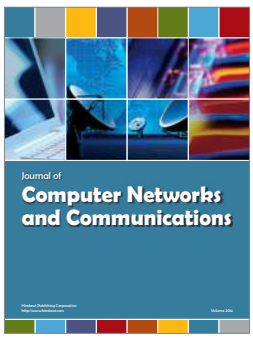












Hindawi
Submit your manuscripts at
http://www.hindawi.com