A Fault-Tolerant Model for Wireless Sensor-Actor System

Keiji Ozaki†, Kenichi Watanabe†, Satoshi Itaya†, Naohiro Hayashibara†, Tomoya Enokido††,
and Makoto Takizawa†
Tokyo Denki University†, Japan
E-mail {kei, nabe, itaya, haya, taki}@takilab.k.dendai.ac.jp
Rissho University††, Japan
E-mail eno@ris.ac.jp

Abstract

In a wireless sensor and actor network (WSAN), a group of sensors and actors are geographically distributed and linked by wireless networks. Sensors gather information sensed for an event in the physical world and send them to actors. Actors perform appropriate actions on actuation devices by making a decision on receipt of sensed information from sensors. Sensors are low cost, low powered devices with limited energy, computation, and wireless communication capabilities. Sensors may not only stop by fault but also suffer from arbitrary faults. Furthermore, wireless communication is less reliable due to noise and shortage of power of sensors. Reliable real time communication among sensors, actors, and actuation devices, is required in WSAN applications. We newly propose a multi-actor/multi-sensor (MAMS) model. In addition, multiple actors may perform actions on receipt of sensed information. Multiple redundant execution of an action on each device have to be prevented and conflicting actions on each device from multiple actors have to be serialized. In this paper, we discuss how to make WSAN reliable and available and how to reliably and non-redundantly perform actions with realtime constraints.

1. Introduction

A wireless sensor and actor network (WSAN) is a collection of sensors, actors and, actuation devices linked by wireless medium to perform distributed sensoring and acting tasks [1, 2, 12]. Sensors gather information about physical world. Actors are capable of making a decision on actions and perform appropriate actions for information gathered by sensors. There are many discussions on how to reliably and efficiently broadcast messages among sensors and actors [5, 12]. WSAN is one of the most significant technologies to realize ubiquitous societies [1, 2, 17].

Sensors are low-cost, low-power devices which are equipped with limited energy, computation, and wireless communication capabilities. Sensors may stop, even malfunction [10] due to the out-of-charge and fluctuation of observed phenomena in the physical world. In addition, the wireless communication between sensors and actors is less reliable. We propose a novel *multi-actor/multi-sensor* (*MAMS*) model where each sensor sends sensed information to multiple actors and an actor receives sensed informations on a same event from multiple sensors in order to be tolerant of faults of sensors and wireless networks. Even if some sensors are faulty and messages are lost in the wireless link, each actor can receive proper sensed information from the other proper sensors. If sensors are arbitrarily faulty [10], an actor takes the majority-based decision on sensed information from multiple sensors. If multiple actors receive the same sensed information, the same action may be performed multiple times by the actors even if the action should be performed only once for the sensed information. On receipt of sensed information from sensors, an actor makes a decision on what action to be performed on devices. In distributed systems where multiple peer processes are cooperating, the processes are required to be causally delivered to the processes [16, 4, 11, 9]. We discuss how to order sensed information received from sensors and actions performed by each actor. In addition, each event in the physical world is characterized by properties like temperature and location. Different sensors may collect properties of an event different from each other. Here, we define *Quality of Event* (QoE) to be properties of event. It depends on observed QoE of events how to order events. We discuss how to order events and actions by considering QoE of events.

In section 2, we present a system model of WSAN. In section 3, we discuss how to make sensors and sensor-actor communications fault-tolerant. In section 4, we discuss how actors reliably and non-redundantly perform actions in an event area.

Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06)
1550-445X/06 $20.00 © 2006 IEEE

2. System Model

2.1. Sensors and actors

A wireless sensor and actor network (WSAN) is composed of sensors, actors, and actuation devices interconnected with wireless medium [1, 2, 12]. Let $\boldsymbol{S}$ be a set of sensors, $\boldsymbol{A}$ be a set of actors, and $\mathbf{O}$ be a set of actuation devices in WSAN.

Phenomena in the physical world is changed by events. Each event is characterized in attributes like temperature and location. Let Ω be a set of all the attributes. Let $\Omega(e)$ show a *scheme* of an event e which is a set of attributes of the event e. An event e is represented as a collection of values of attributes. Here, $e.a$ shows a value of an attribute a of an event e. If an event occurs in some location of the physical world, sensors in some distance from the location gather values of some attributes of the event. An event area is a geographical unit of WSAN. A WSAN W is partitioned into event areas $W_1, \cdots, W_m$ ($m \geq 1$). Each event area W_i is composed of sensors, actors, and actuation devices. $\mathbf{S}_i$, $\mathbf{A}_i$, and $\mathbf{O}_i$ show sets of sensors, actors, and devices in an event area W_i. Sensors in $\mathbf{S}_i$ gather physical phenomena of events occurring in an event area W_i. Then, the sensors send to sensed information to actors in $\mathbf{A}_i$. The actors perform actions on actuation devices in $\mathbf{O}_i$.

A *type* of a sensor is a subset of the attributes. Let $\Omega(s)$ be a type of a sensor s, i.e. a collection of attributes ($\Omega(s) \subseteq \Omega$). $e[s]$ shows information on an event e which a sensor s gathers. Multiple sensors sense a same event e in an event area. Then, the sensor s sends the sensed information $e[s]$ of the event e to actors in the event area. Figure 1 shows the relations of sensors, actors, and actuation device objects.

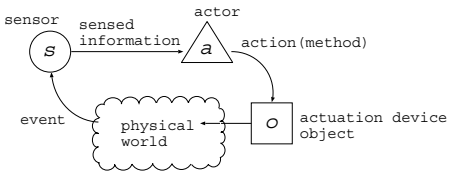

Figure 1. Sensor, actor, and device object.

2.2. Multi-actor/multi-sensor (MAMS) model

In the single-actor (SA) model [1], all the sensors in an event area send sensed information to one actor [Figure 2]. Here, the actor can be a single point of failure. On the other hand, in the multi-actor (MA) model [1], each sensor sends sensed information to one actor but some pair of sensors in an event area may send sensed information of an event to different actors. In order for each actor to make the decision, more number of sensors are required. In the SA and MA models, each sensor sends sensed information to one actor in each event area. In this paper, we propose a *multi-actor/multi-sensor* (*MAMS*) model to realize the fault-tolerant WSAN. Each sensor sends sensed information to multiple actors and each actor receives sensed information from multiple sensors in an event area [Figure 3].

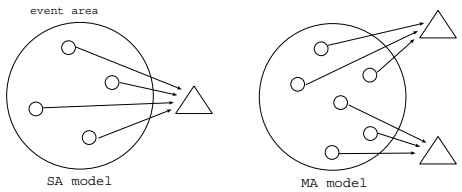

Figure 2. SA and MA models.

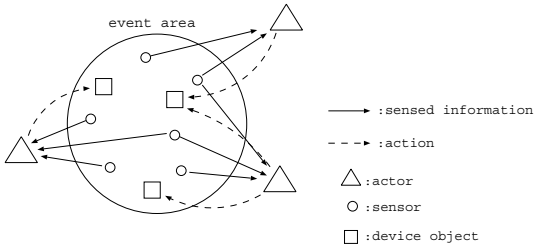

Figure 3. Multi-actor/multi-sensor (MAMS) model

Each sensor broadcasts a message of sensed information in wireless medium. An area where each sensor can deliver a message with wireless medium is referred to as *broadcast cell*. Durresi *et al.* [5] discuss how to distribute sensors in a area so that every event to occur can be sensed by some number of sensors.

2.3. Object-based actions

An actuation device is modeled to be an object in this paper. An action is modeled to be the execution of a method on an object. On receipt of a method issued by an actor, the method is performed on an actuation device object.

Let $op(s)$ show the result obtained by performing a method op on a state s of the world. Here, let op_1 and op_2 be a pair of methods of an object o. Let $op_1 \circ op_2$ show a serial execution of methods op_1 and op_2 on an object o. Let $op_1 \parallel op_2$ denote a parallel execution of methods op_1 and op_2 on an object o. A method op_1 is referred to as *equivalent* with another method op_2 ($op_1 \equiv op_2$) iff the result obtained by performing op_1 on every state is the same as op_2. ϕ shows a null method which does nothing on the object o. There are following relations among a pair of methods op_1 and op_2 of an object o:

1. op_1 and op_2 *conflict* with one another if and only if (iff) $op_1 \circ op_2(s) \neq op_2 \circ op_1(s)$ for some state s.

1550-445X/06 $20.00 © 2006 IEEE

2. op_1 and op_2 are *compatible* (or do not conflict) iff $op_1 \circ op_2(s) = op_2 \circ op_1(s)$ for every state s.
3. op_2 *absorbs* op_1 iff $op_1 \circ op_2(s) = op_2(s)$ for every state s.
4. op_1 is *compensated* by op_2 iff $op_1 \circ op_2(s) = s$ for every state s.
5. op_1 is *idempotent* if $op_1(s) = op_1 \circ op_1(s)$ for every state s.

At most one method can be performed at a time on a type of an actuation device object. This type of device is referred to as *sequential* object. On the other hand, a device object is referred to as *concurrent* object if and only if (iff) multiple methods can be in parallel performed on the object. The execution of some method is not assumed to be atomic since it take a longer time to perform the method. That is, the execution of a method may be interrupted.

3. Fault-Tolerant Sensors

3.1. Types of faults

A pair of sensors may collect different values for an event due to sensor errors. That is, $e[s_i] \neq e[s_j]$ for an event e and some pair of sensors s_i and s_j in an event area. Even if $e[s_i] \neq e[s_j]$, $e[s_i]$ and $e[s_j]$ are *equivalent* with respect to the quality of event (QoE) ($e[s_i] \equiv e[s_j]$) if QoE of sensed values $e[s_i]$ and $e[s_j]$ are considered to be the same from the application point of view. An actor a collects sensed values $e[s_i]$, $\cdots$, $e[s_l]$ from sensors s_1, $\cdots$, s_l in an event area. Here, the actor a first takes the median v of the values. If $|v$ - $e[s_i]| \leq \mu$ for some constant μ, $e[s_i]$ is considered to be proper. μ is decided for attributes in an application. Here, a pair of sensed information $e[s_i]$ and $e[s_j]$ are *equivalent* iff $e[s_i]$ and $e[s_j]$ are proper. Otherwise, $e[s_i]$ and $e[s_j]$ are *different*. If a sensor s_i shows a non-proper value $e[s_i]$, an actor a considers s_i to be faulty. Thus, sensors not only stop by fault but also show arbitrary fault.

Since actors are higher-cost devices with enough energy and computation capability. Actors are reasonably assumed to only stop by fault in this paper. That is, an operational actor is always proper. In addition, we assume actors are communicating with each other by using highly reliable communication links. We also assume actuation device objects and communication links among actors and objects to be reliable.

In summary, we make the following assumptions on faults to occur in WSAN:

1. A sensor may be arbitrarily faulty. That is, a sensor may not send a message for an event, may send a different value from other sensors to actors for some event, and may send a message even if an event does not occur in an event area.
2. An actor may only stop by fault.
3. An actuation device object is reliable.
4. Omission faults occur in communication between sensors and actors, i.e. messages may be lost.
5. Actors reliably communicate with each other. Actors also reliably communicate with device objects.

We discuss how to increase the reliability of sensors and sensor-actor communication in this paper.

3.2. Fault-tolerant sensors

We assume every pair of proper sensor s sends equivalent sensed value s for an event e to actors in an event area. We assume that at most fs_i sensors out of l_i sensors $s_{i1}, \cdots, s_{il_i}$ are faulty in an event area W_i. There are m_i (> 1) actors $a_{i1}, \cdots, a_{im_i}$ in an event area W_i.

An actor has to detect faulty sensors. Let $t_{ik}[e]$ show global time when a sensor s_{ik} senses an event e in an event area W_i. Let $r_{isk}[e]$ be global time when an actor a_{is} receives a message of an event e from a sensor s_{ik}. Here, we make the following assumptions in each event area W_i:

1. For every pair of proper sensors s_{ik} and s_{ih} in an event area, $|t_{ik}[e]$ - $t_{ih}[e]| \leq \alpha_i$ if an event e occurs in an event area W_i.
2. $|r_{isk}[e]$ - $r_{ish}[e]| \leq \beta_i$ if an actor a_{is} receives messages of sensed information of an event e from a pair of proper sensors s_{ik} and s_{ih}.

Here, α_i and β_i are constants on an event area W_i. α_i shows time difference between a pair of sensors which sensed an event. We assume at most one event occurs for every α_i time units in W_i. The constant β_i is given by the constant α_i and the maximum difference τ_i of delay time from a sensor to an action in an event area W_i. If $|r_{ish}[e] - r_{isk}[e]| \leq \beta_i$, the actor a_{is} is referred to as *simultaneously* receive sensed information $e[s_{ik}]$ and $e[s_{ih}]$ from a pair of the sensors s_{ik} and s_{ih}, respectively. Hence, β_i is minimum time deference which an actor can recognize a pair of different events. In this paper, each actor is assumed to be synchronized with a global clock. For example, a clock in each actor is synchronized with a GPS time server in NTP (network time protocol) [13].

An actor a_{it} detects a sensor s_{ik} to be faulty in an event area W_i as follows:

1. If an actor a_{is} does not receive any message from a sensor s_{ik} for β_i time units since a_{is} received the first message from some sensor, a_{is} considers s_{ik} to be faulty.
2. An actor a_{is} collects sensed values from sensors since a_{is} take proper values our of the values, from the sensors. If the proper values are majority in a set of the

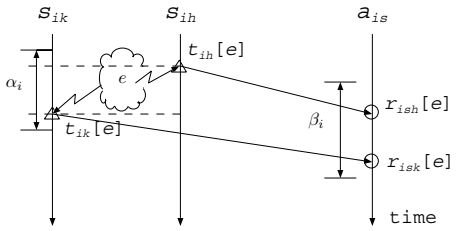

Figure 4. Event in an event area W_i.

sensed values, the actor a_{is} takes the average value v of the proper values. Here, a_{is} considers a sensor s_{ik} to be faulty if s_{ik} sends a non-proper value v'.

If each actor a_{is} receives sensed information from more than $2fs_i + 1$ sensors, at most fs_i sensors are faulty and at least $fs_i + 1$ sensors are proper. Hence, a_{is} takes the majority proper value of the sensed values sent by the sensors [Figure 5].

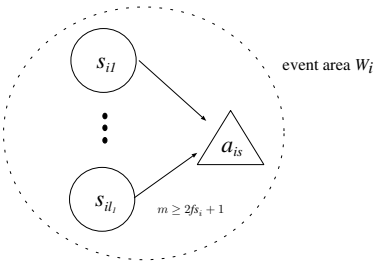

Figure 5. Communication from sensors to an actor.

4. Actors

4.1. Multiple instances of an action

In the MAMS model, each of actors $a_{i1}, \cdots, a_{im_i}$ receives sensed value of an event e from multiple sensors s_{i1}, $\cdots$, s_{il_i} in the event area W_i. Suppose that a method op is to be performed on an actuation device object o in the event area for an event e. If each actor a_{is} makes a decision on the method op and performs op, op is $m_i(\geq 1)$ times performed on the object o. Here, the state of the object o may get inconsistent. The redundant invocation of a method by multiple actors have to be resolved.

There are *actor-side* and *object-side* approaches to realizing the unique execution of a method on an actuation device object. In the actor-side approach, only one method is issued to an object from multiple actors. On receipt of a method, the method is just performed on the objects. In one way, the actors cooperate with each other to make consensus on which actor to issue a method to an object o. Then, only the selected actor issues the method while the other actors do not issue the method to the object o. It takes time to exchange messages among the actors.

In the object-side approach, an actuation device object o takes a method op only once even if each of multiple actors sends the method to the object. Each of the actors issues a method op to an object o. Each instance op_{is} of a method op issued by an actor a_{is} is identified in a pair of method type op and time t_{is} when a_{is} receives an event. Here, a pair of identifiers $\langle op, t_{is} \rangle$ and $\langle op', t_{it} \rangle$ of instances op_{is} and op_{it}, respectively, are referred to as *temporarily equivalent* ($op_{is} \cong op_{it}$) iff $op = op'$ and $|t_{is} - t_{it}| \leq \alpha_i$. An object o considers a method op_{it} to be redundant. On each device object, each method can be only once performed even if multiple actors send instances of the method to the device object.

4.2. Synchronization of multiple actors

Next, suppose a pair of events e_1 and e_2 occur in an event area W_i. An actor a_1 receives sensed information of the event e_1 and another actor a_2 receives sensed information of the event e_2. The actor a_1 issues a pair of the methods op_{11} and op_{12} to the objects o_1 and o_2, respectively, and the actor a_2 issues op_{21} and op_{22} to the objects o_1 and o_2, respectively, as shown in Figure 6. A pair of the methods op_{11} and op_{21} are performed on the object o_1 and a pair of op_{12} and op_{22} are performed on o_2. Here, suppose a pair of methods op_{11} and op_{21} conflict on the object o_1 as well as a pair of op_{12} and op_{22} conflict on o_2. Here, the serializability [3] is required.

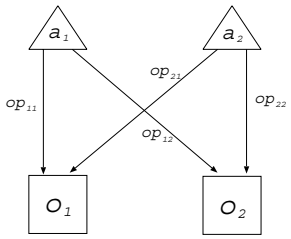

Figure 6. Serializability.

In order to realize the serializability, objects are locked before methods are performed on the object in one way [3, 6]. If an object is locked by another method conflicting with a method op, op is kept waiting until the lock is released. In another way, method requests are timestamped in each actor [3]. Here, every pair of conflicting methods are performed in the time-stamp order.

Actuation device objects are classified into a pair of types: *synchronous* and *non-synchronous* types of object. A synchronous object supports some synchronization mechanisms. One type of synchronous object is a *locking type*

which supports the strict two phase locking (2PL) protocol [6]. Here, if *lock* and *unlock* methods are supported by the object. An actor issues a lock request to an object before issuing a method. The method is performed on the object if the object is locked. Another synchronization type of an object is a *TO* (timestamp ordering) [3] object. An actor issues a method op to an object o. Here, the method op is given timestamp $ts(op)$ by the actor. Here, op is sent to the TO scheduler of the object o. In the TO scheduler, every pair of conflicting methods op_1 and op_2 are ordered in the timestamp order. That, op_1 precedes op_2 if $ts(op_1) < ts(op_2)$. Then, the TO scheduler delivers methods to the object in the timestamp ordering.

A non-synchronous object does not support any synchronization mechanism. An actuation device object normally does not support any synchronization mechanism. Hence, actors are required to issue synchronously conflicting methods with each other. Here, an actor is assumed to communicate with sensors and the other actors by using the wireless medium in an event area W_i. An actor a_{is} sends a request message of a method op with the timestamp $ts(op)$ to an object by broadcasting the message in an event area. Every actor selects one actor whose timestamp is the minimum. Then, the selected actor whose timestamp is the minimum sends the method op to the object. If actors are assumed to be arbitrarily faulty each actor takes the majority-based decision as discussed in the sensor's faults.

5. Concluding Remarks

In this paper, we discussed how to make a wireless sensor and actor network (WSAN) fault-tolerant. A WSAN is composed of event areas. Each event area is composed of sensors, actors, and actuation device objects. Each sensor communicates with multiple actors and each actor receives sensed information from multiple sensors by using the wireless communication medium. This is the multi-actor/multi-sensor (MAMS) model which we proposed to make WSAN fault-tolerant in this paper. An actor makes a decision on what method. Sensors are less reliable and may be arbitrarily faulty, due to low-energy, low cost devices. Actors are assumed to only stop by fault. We discussed fault-tolerant sensor-actor and actor-actor communication protocols. The protocols are based on the physical time. In addition, multiple actors issue method requests to an object in the MAMS model. We discussed how to realize the reliable and non-redundant execution of a method on an object.

Acknowledgment

This research is partially supported by Research Institute for Science and Technology [Q05J-04] and Academic Frontier Research and Department Center [16-J-6], Tokyo Denki University.

References

[1] I. F. Akyildiz and I. H. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks journal (Elsevier)*, 2:351–367, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks journal (Elsevier)*, 38:393–422, 2002.

[3] P. A. Bernstein, V. Hadzilacos, and N. Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.

[4] K. Birman, A. Schiper, and P. Stephenson. Lightweight causal and atomic group multicast. *ACM Transactions on Computer Systems*, 9(3):272–314, 1991.

[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)*, 1:343–348, 2005.

[6] 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):624–637, 1976.

[7] N. Hayashibara, X. Défago, R. Yared, and T. Katayama. The φ accural failure detector. *Proc. of the 23rd IEEE International Symposium on Reliable Distributed Systems (SRDS-23)*, 1:68–78, 2004.

[8] S. Kawanami, T. Nishimura, T. Enokido, and M. Takizawa. A Scalable Group Communication Protocol with Global Clock. In *Proc. of AINA-2005 International Workshop on Ubiquitous Smart Worlds (USW 2005)*, volume 2, pages 625–630, 2005.

[9] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Comm. ACM*, 21(7):558–565, 1978.

[10] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[11] F. Mattern. Virtual time and global states of distributed systems. *Parallel and Distributed Algorithms*, pages 215–226, 1989.

[12] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz. A distributed coordination framework for wireless sensor and actor networks. *Proc. of the 6th ACM international symposium on Mobile ad hoc networking and computing*, 1:99–110, 2005.

[13] D. L. Mills. Network time protocol. RFC 1350, 1992.

[14] V. Paruchuri, A. Durresi, and L. Barolli. Energy aware routing protocol for heterogeneous wireless sensor networks. *Proc. of 16th International Workshop on Database and Expert Systems Applications (DEXA2005)*, pages 133–137, 2005.

[15] A. K. Somani and N. H. Vaidya. Understanding fault tolerance and reliability. *IEEE Computer*, 30:45–50, 1997.

[16] T. Tachikawa, H. Higaki, and M. Takizawa. Group communication protocol for realtime applications. *Proc. of the 18th IEEE International Conf. on Distributed Computing Systems(ICDCS-18)*, pages 40–47, 1998.

[17] M. Weiser. Hot topics: Ubiquitous computing. *IEEE Computer*, pages 71–72, 1993.