International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.9 (2014), pp.335-350
http://dx.doi.org/10.14257/ijmue.2014.9.9.34

New Causally Ordered Delivery Protocol Using Information of Immediate Predecessor Messages between Brokers and Subscribers

Chayoung Kim and Jinho Ahn[1]

Dept. of Comp. Scie., Kyonggi Univ., Iuidong, Yeongtong, Suwon 443-760 Gyeonggi, Republic of Korea
{kimcha0, jhahn}@kgu.ac.kr

Abstract

In most large-scale distributed applications, publish/subscribe (P/S) systems allow a decoupling between senders and receivers to interact with publishers and subscribers. Recently, in these areas, there has been increasing emphasis in managing end-to-end message delivery performance and message order-based consistency, which have been addressed in distributed collaborative applications. Especially, a causally ordered delivery consistency is more useful for these applications in which a large number of processes request collaboratively and interactively. In P/S systems of wireless sensor networks (WSNs), data fusion, the process of correlating individual sensor readings into high-level sensing results, depends on a causally ordered delivery. And gossip protocols based on P/S systems are becoming one of the promising solutions for addressing P/S scalability problems and very useful for the applications with a mixture of diverse message order consistencies.

In this paper, we present two versions of causally ordered delivery protocols based on P/S systems using gossip protocols. The one is that only the predecessors immediately before the multicast message are disseminated from brokers to subscribers. And the other, specifically for P/S systems of WSNs, is that the timestamps that represent the gossip round in which the immediate predecessors are generated are disseminated from brokers to subscribers. The features of these two versions might be highly scalable and suitable for the area of the applications requiring only the minimum causal information with flexible consistency.

Keywords: *Publish/Subscribe, Wireless Sensor Network, Group Communication, scalability, Causal message ordering*

1. Introduction

In most large-scale distributed applications, such as social web platforms, publish/subscribe (P/S) systems are suitable for communication between software components that are deployed over a large number of sites. P/S systems follow a many-to-many communication pattern, allowing a decoupling between senders and receivers to interact with publishers and subscribers [10]. Recently, social web platforms, such as Facebook and Twitter, have become real-time social communication ones, focusing on the dissemination, processing and caching of fresh data. So, it is important and reasonable that end users expect on-the-fly data, which is immediately available to all, interested other end users [5]. And, there has been increasing emphasis in managing end-to-end message delivery performance and message order-based consistency, which have been addressed in distributed collaborative applications. Especially, for on-the-fly

[1] Corresponding author: Tel.:+82-31-249-9674; Fax:+82-31-249-9673.

data processing, some distributed applications and products have offered message order consistency guarantees, such as Isis2 system [3]. Isis2 supports virtually synchronous process groups and considers full-fledged atomic message ordering, but not causal message ordering. A causal order protocol ensures that if two messages are causally related and have the same destination, they are delivered to the application in their sending order [1]. A causal order is more useful than a strong atomic order for large-scale distributed applications in which a large number of processes request collaboratively and interactively in real-time social web platforms based on P/S systems [15]. And gossip protocols based on P/S systems are becoming one of the promising solutions for addressing P/S scalability problems and very useful for the applications with a mixture of diverse message order consistencies [3]. In this paper, we present two versions of causal order protocols using gossip protocols.

One of the proposed protocols is appropriate for broker-based P/S systems, which is adequate communication infrastructures to alleviate hot spots and to elastically scale in and out for better exploiting network and computational resources provisioned from the brokers, where the network connections are much more reliable and the node memberships are much more stable. In these systems, end users subscribe to their chosen brokers. In the protocol, P/S systems use dedicated and interconnected brokers to process events based on gossip-style disseminations, i.e., these systems are in the same manner as Patrick *et al.* [4] shows. The other of the proposed protocols is for wireless sensor networks (WSNs) consisting of large numbers of cooperating small-scale nodes capable of wireless communication and sensing [12]. The protocol is based on gossip protocols of sensor broker-based networks like as [19] and guarantees causal message ordering, in a manner, which is somewhat similar to temporal ordering, achieved through data fusion depending on the time of occurrence of sensor readings [17]. But, the temporal order is based on the physical time synchronization, but the causal ordering is not.

In broker-based P/S systems of our two proposed protocols, brokers might aggregate the information of the results based on the subscribers' interests, while guaranteeing causal message ordering. And the subscribers receive the results in the way of gossip-style disseminations by their chosen brokers. P/S systems are based on gossip protocols [4], which are seemed more appealing in these systems because they are more scalable and easy to deploy than traditional reliable group communications [1]. And if gossip protocols based on P/S systems could deal with end-to-end message delay and ordering consistencies, such as causal ordering, distributed collaborative applications running on the real-time social web platforms might focus on synchronizing their end users' needs, such as video playback on multiple devices with different engines and network bandwidth [5, 10].

In large-scale distributed applications of P/S systems, if causal ordering protocol is performed by the all brokers on global membership views, it is likely to be high overloaded on every member and not scalable. In order to address this problem, promising gossip protocols should have all the required features by achieving a high degree of reliability and strong message delivery ordering guarantees, even if every broker has a local membership view [4]. So, we present two versions of causal order protocol, the one is based on a local view, which is for larger-scale distributed P/S systems and the other is based on a global view, which is for pre-planned wireless sensor networks (WSNs). One of the proposed protocol based on local views guarantees the causally ordered delivery by using the context graph [16], which manages the causal order information based on the semantics of sent and received messages. But, the other based on the global views uses the whole set of vectors [1], used for traditional reliable group communications [1]. In some WSNs, such as pre-planned and time-lines ones [18, 19], it is not considerable to manage the global views. In the proposed protocol based on local views, all ancestors before the multicast message are sent and received between the brokers. But, from brokers to subscribers, only the immediate predecessors are disseminated instead of all ancestors. Its features might result in its very low communication overhead between brokers and subscribers because the immediate predecessors are in the structure of one-dimensional vector. But, all ancestors are in the structure of two-dimensional context graph [16].

In the global views of the whole set of vectors, every broker can manage a vector per group that represents its knowledge for the number of multicast messages generated by other members, as same as each member in the protocol of Birman *et al.* [1]. In some WSNs, which can manage global view without high loads, every broker can aggregate, send and receive the whole set of group vectors for causal ordering by gossip protocols and fire synchronization [18]. In the proposed protocol based on global views, between the sensor brokers, the whole set of vectors, which represents the knowledge for the number of multicast messages are sent and received. But, from brokers to subscribers, only the timestamp that represents the gossip round in which the immediate predecessor are generated are disseminated. Especially, in the protocol, the timestamp is represented in the way of colors, which stands for the gossip round. And broker A and B can generate a message per gossip round. That means that the proposed protocol needs one-dimensional vector, whose size is the number of brokers because of one color per sensor broker. The protocol is appropriate for sensor networks in a pre-planned manner time-lines ones [18] because it is not a high burden to manage global membership views. Therefore, these two versions of causally ordered delivery protocols are highly scalable and suitable for the area of the applications requiring only the minimum causal information with flexible consistency.

2. The Proposed Protocol

2-1. Basic Idea

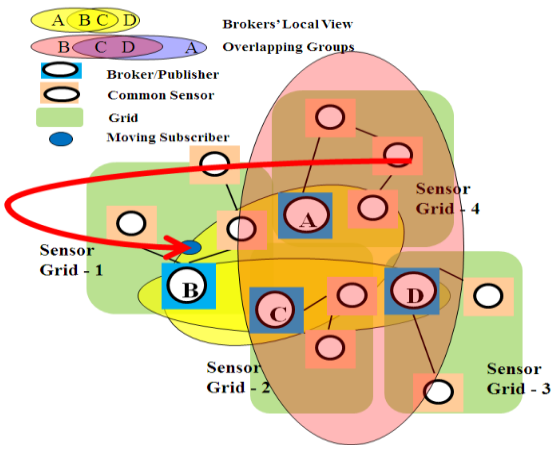

Figure 1. A Wireless Sensor Network

In P/S systems of our two proposed protocols, brokers are publishers to send topics to their subscribers and subscribers receive messages matching their interests through their chosen brokers by gossip-style disseminations. Recently, much research has been devoted to designing broker selection methods that best suits application needs [7, 17, 20]. The brokers might gossip about the information based on the subscribers' needs, while guaranteeing the causally ordered delivery [7, 17]. Researches on P/S systems in WSNs [20] have mainly focused on mobile subscribers relying on broker-based infrastructure. In figure 1, we can see that each sensor broker manages a sensor grid and a moving subscriber can migrate to another sensor grid.

In the protocol based on the context graph [16], although each broker is based on its local view of gossiping, if a message includes the context graph, as a consequence, it is possible for the other brokers to know what they should have received. In the protocol, in the first step of sending a message, when every broker generates a multicast message, it puts its ID, the sequence number and the group lists of all groups that it participates on the message. In the second step, the broker attaches the message to all leaf nodes in its context graph. Then, the multicast message becomes the leaf and the parent messages of it become the immediate predecessors. In the last step, the broker sends the message including all ancestors of it to other brokers like as the protocol of [16]. On the other hand, when the broker disseminates the multicast message to subscribers, it includes only the parent messages, that are immediate predecessors, instead of all ancestors in the context graph. The size of the immediate predecessors is only one-dimensional vector because every broker can generate a message per gossip round. But, the size of the context graph is two-dimensional vector. Therefore, from brokers to subscribers, the size of the information for causal ordering could be reduced. When a broker receives the multicast message, if the broker has received all ancestors, then the broker delivers it. And when a subscriber receives it, if the subscriber has received the immediate predecessors, then the subscriber delivers it.

In the protocol for the pre-planned WSNs [19], every broker manages one vector per group, whose element has the number of multicast messages sent to other brokers. In these WSNs, every broker can gossip about the multicast messages based on global views because it knows about every other broker. And it is not considerable for every broker to manage global views like in an environment of [18]. In the existing protocol [1], every member should include the whole set of group vectors without distinguishing brokers (publishers) and subscribers. But, our proposed protocol is based on P/S systems for pre-planned WSNs. In the protocol, in the first step of sending a message, when every broker generates a multicast message, it posts the current gossip round timestamp on it. In the second step, the broker updates every element of the whole set of group vectors. Then the broker updates all elements of a one-dimensional vector, whose elements represent the gossip round timestamp of all immediate predecessor messages of the newly generated message. So, in the protocol, the one-dimensional vector is called as the immediate predecessor vector. In the last step, the broker sends the message including the whole set of group vectors to other brokers, like as the protocol of [1]. On the other hand, when the broker disseminates the multicast message to subscribers, it includes only the immediate predecessor vector of length N, where N is the number of all brokers, instead of the whole set of group vectors, which is two-dimensional vector. When a broker receives the multicast message, if the broker verifies the whole set of group vectors, then the broker delivers it and when a subscriber receives the message, if the subscriber verifies the immediate predecessor vector, then the subscriber delivers it.

Therefore, two versions of our proposed protocols result in its very low cost communication overhead in between brokers and subscribers because there is different causal order information between communication groups, from brokers to brokers and from brokers to subscribers.

2.2. The Protocol in Pre-planned Wireless Sensor Networks

2.2.1. Algorithm Description

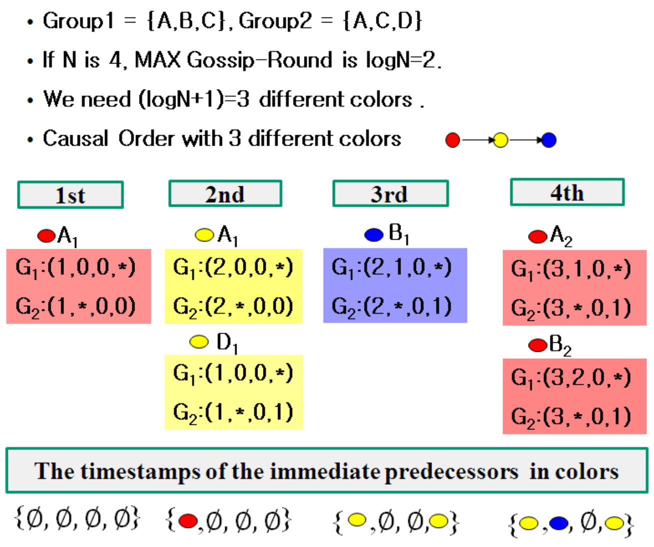

Figure 2. Each Gossip Round Represented in Each Color

In this section, we describe our proposed protocol for pre-planned WSNs through examples of figure 2 and 3, which show how in detail each broker generates a multicast message and aggregates causally ordered delivery information. As same as the protocol of Birman *et al.* [1], in our proposed protocol, a vector timestamp (VT) per group, piggybacked on a multicast message counts the number of messages that causally precedes it. So, the notation g_α : VT[i] counts multicast messages sent in the group g_α by a process i. In general, gossip protocols take O (logN) gossip rounds to reach all nodes [4], where N is the number of nodes. In each gossip round, every process has initiated a gossip message exactly once and gossips about it to $f \geq 1$ other processes, called as fan-out (f) or gossip targets, at random. So, each gossip round can be characterized as a unique notation represented using a color. The proposed protocol needs logN + α colors because the maximum number of gossip rounds in which all processes receive all messages eventually is logN and α may be application specific for buffering. As shown in figure 2, if two messages m(A_1) and m'(D_1) have been sent at the same gossip round, then they are independent of each other and represented in the same color, that is yellow.

This example of figure 2 shows how in detail each broker participating in G_1 = {A, B, C} and G_2 = {A, C, D} aggregates the information of causally ordered delivery and sends it to subscribers. The example of figure 2 sets α to 1. So, it needs 3 colors because logN is 2 and α is 1. The stale messages might be removed periodically to respect the maximum number of gossip rounds as same as pbcast [4]. Also, our proposed protocol uses the epoch, which is incremented by 1 whenever all colors (in Figure 2, red->yellow->blue) have been completed exactly once, distinguishing new message from previous message sent by the same process in the same color. As shown in figure 2, A_1 and A_2 in red can be distinguished by the epoch 1 and 2. In Figure 2, in the first round, broker A generates the message and makes it with ID "A", the epoch "1" and the current gossip round color "red", as "red$_{A1}$". At the beginning, the vector of the immediate predecessors is all 0, that is {0, 0, 0, 0}. In the second round, broker A and D generate each message, and make it with ID "A" and "D", the epoch "1" and the current gossip round color "yellow", as "yellow$_{A1}$" and "yellow$_{D1}$", respectively. On receiving

"red_{A1}", the broker A and D update the vector of the immediate predecessors, as {red, 0, 0, 0}. On being the fourth round, the epoch is incremented by 1 because red->yellow->blue have been completed exactly once. So, in the fourth round, broker A and B generate each message, and make it with ID "A" and "B", the epoch "2" and the current gossip round color "red", as "red_{A2}" and "red_{B2}", respectively. On receiving "$yellow_{B1}$", all brokers update the vector of the immediate predecessors, as {yellow, blue, 0, yellow}.

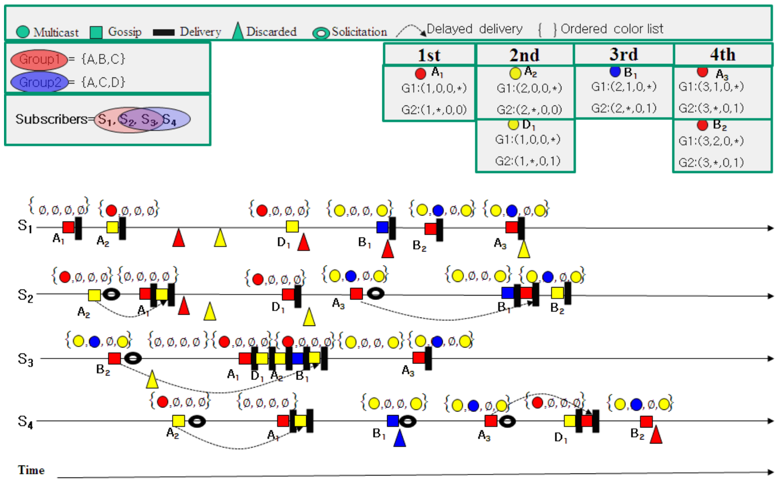

Figure 3. An Example of the Vector of the Latest Gossip Rounds from Sensor Brokers to Subscribers

Figure 3 shows how in detail broker groups G_1={A,B,C} and G_2={A,C,D} gossip about the multicast messages including one-dimensional vector to subscribers={S_1,S_2,S_3,S_4}, whose element is the gossip round timestamps of the immediate predecessors, represented in colors(as shown in Figure 2, red->yellow->blue). In between brokers, the two-dimensional the whole set of vectors (a vector per group) are sent and received, like as the previous protocol [1]. But from brokers to subscribers, the vector of the group round timestamps of the immediate predecessors is disseminated, instead of the whole set of all group vectors. In the example of Figure 3, subscriber S1 subscribes to vector C in G_1, S_2 and S_3 subscribe to A or B in both G_1 and G_2, and S_4 subscribe to D in G_2. The current color in the vector is all 0 at the beginning. Every broker makes the causal ordering in what order is "red_{A1}" -> "$yellow_{A1}$" = "$yellow_{D1}$" -> "$blue_{B1}$" -> "red_{A2}" = "red_{B2}". S1 receives all messages from Broker B. It discards "$yellow_{D1}$", because it does not belong to G_2. On receiving "$yellow_{A1}$", S_2 requests "red_{A1}" to A because it knows that it did not receive "red_{A1}" based on the piggybacked vector of the group round timestamps of the immediate predecessors, {red,0,0,0}. On receiving "red_{B2}", S_3 requests all messages it has not received to B. On receiving "$yellow_{A1}$", S_4 requests the "red_{A1}" to D. And S_4 discards all messages from broker B because it does not belong to G_1. As figure 3 shows, the size of the vector is the number of brokers because one color represents only one sensor broker and every broker gossips about the message per one gossip round. It is surely the main cause of highly scalable, guaranteeing flexible consistency.

2.2.2. Proof of the Protocol

Rule 1:

For Gossip round Timestamps (GT), $\forall \in k\{1, 2, \ldots, n\}-i$, $GT_S[k] >= GT_m[k]$ and $GT_S[i] = GT_m[i]-1$.

1. Causality is never violated (Safety).

Proof:

Consider the actions of a process pj that receive two messages m_1 and m_2 such that $m_1 \rightarrow m_2$.

Case 1:

m_1 and m_2 are both transmitted by the same process p_i. This case is trivial. Under rules, in the protocol, m_2 can only be delivered after m_1 has been delivered.

Case 2:

m_1 and m_2 are transmitted by two distinct processes p_i and $p_{i'}$. By induction on the messages received by process p_j, we will prove that m_2 cannot be delivered before m_1. Assume that m_1 has not been delivered and that p_j has received k messages. Since $m_1 \rightarrow m_2$, we have $GT_{m1}[i] <= GT_{m2}[i]$. ------ Relation (1)

Base step:

The first message delivered by S cannot be m_2. Recall that if no messages have been delivered to S, $GT_S[i]=0$. However, $GT_{m1}[i]>0$ (because m_1 is sent by p_i), hence $GT_{m2}[i]>0$. By the rules, m_2 cannot be delivered by S.

Inductive step:

Suppose S has received k messages, none of which is a message m such that $m_1 \rightarrow m$. If m_1 has not yet been delivered, then $GT_S[i]< GT_{m1}[i]$ -------------------- Relation (2)

The only way to assign a value to $GT_S[i]$ greater than $GT_{m1}[i]$ is to deliver a message from S that was sent subsequent to m1 and such a message would be causally dependent on m_1. From Relation (1) and Relation (2) it follows that $GT_S[i]< GT_{m2}[i]$. By application of the rules of our protocol, the k+1st message delivered by S cannot be m_2.

2. In the absence of failures, every message in overlapping groups is indeed delivered (Liveness).

Proof:

Suppose there is a multicast message m sent by process p_i in overlapping groups that can never be delivered to process p_j. The rule of the protocol implies that for some i, $GT_{pj}[i]< GT_m[i]$. The number of messages that must be delivered to p_j before m is finite. Communication links are fair-loss, but correct processes can construct reliable communication links on top of fair-loss links by periodically retransmitting messages. So, in the absence of failures and after some finite time, all these messages will have arrived at p_j. If every such message had been delivered, then we would have that $GT_{pj}[i]> GT_m[i]$; contradiction.

So, there exists at least another message m' which will not be delivered to pi and should be before m. If m' is in a waiting buffer, then $GT_{m'}[i] < GT_m[i]$ for some i. We can thus apply the same reasoning to m' as to m, which completes the proof by finite decreasing induction.

2.3. The Protocol in Pre-planned Wireless Sensor Networks

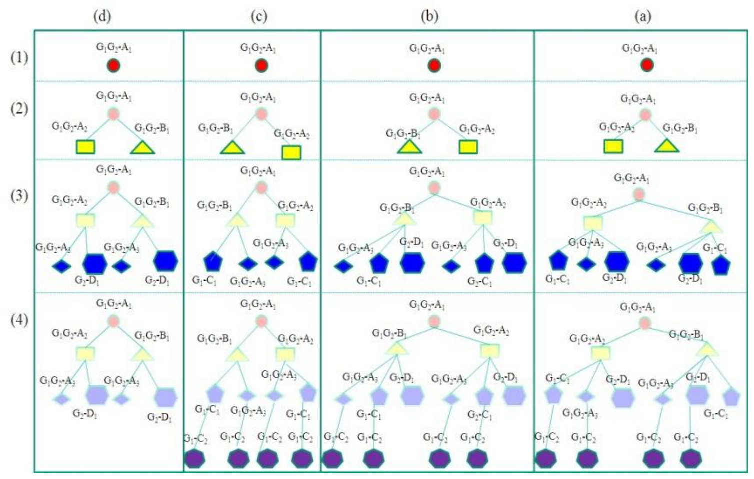

Figure 4. Each Context Graph Sent and Received between Brokers in Overlapping Groups

2.3.1. Algorithm Description

In this section, we describe our proposed protocol based on each local view of gossiping by using a context graph [16] through examples of figure 4, 5 and 6. If each broker is based on its local view and a message includes the context graph, it is available for the other brokers to receive because the context graph is defined in terms of the semantics of send and receive messages as shown in figure 4. These examples from row (1), column (d) to row (4), column (a) of figure 4 show how in detail each broker = {A, B, C, D} participating in G1 = {A, B, C} and G2 = {A, B, D} generates a multicast message, puts ID, the sequence number and the group lists on it, and attaches it to the leaf nodes of the context graph. Column (D) is the context graph of broker D, column (C) is for C, column (B) is for B and column (A) is for A. In figure 4, "G_1" is the name of group G_1 and "A_1" is the message generated by A at first. When A receives a multicast message, A verifies that all ancestor messages preceding it have been already received, comparing its context graph with the included all ancestors. In row (1), column (A), A generates the first multicast message and makes it as "G_1G_2-A_1", with ID "A", the group lists “G_1G_2” and sequence number “1”. Then, A makes its context graph with G_1G_2-A_1 as the root and sends the message G_1G_2-A_1 including the context graph to other brokers. In this case, because there is no preceding message, there is no the included context graph in G_1G_2-A_1. In row (1), column (B), row (1), column (C) and row (1), column (D), B, C, and D receive G1G2-A1 and make their context graph with G_1G_2-A_1 as the root, respectively. In row (2), column (A), A generates the second message and makes it as "G_1G_2-A_2". Then, A attaches "G_1G_2-A_2" to G_1G_2-A_1 in its context graph as the leaf. So, G_1G_2-A_2 becomes the leaf and G_1G_2-A_1 becomes the immediate predecessor. At the same time, in row (2), column (B), B generates the first message and makes it as "G_1G_2-B_1". Then, B attaches "G_1G_2-B_1" to G_1G_2-A_1 in its context graph as the leaf. In row (2), column (C), C receives G_1G_2-A_2 and G_1G_2-B_1 and compares its context graphs with the included all ancestors, G_1G_2-A_1. If C has already received all ancestors, it attaches G_1G_2-A_2 and G_1G_2-B_1 to G_1G_2-A_1 in its context graph as the leaf. In row (2), column (D), D does what C does. In row (3), column (C), C generates the first message and makes it as "G_1-C_1". Then, C attaches "G_1-C_1" to G_1G_2-A_2 and G_1G_2-B_1 in its context graph as the leaf. In row (3), column (D), D generates the first message and makes it as "G_2-D_1". Then, D attaches "G_2-D_1" to G_1G_2-A_2 and G_1G_2-B_1 in its context graph as the leaf. In row (3), column (A), A generates the third message and makes it as "G_1G_2-A_3". Then, A attaches "G_1G_2-A_3" to G_1G_2-A_2 and G_1G_2-B_1 in its context graph as the leaf. In row (3), column (B), B receives each message G_1-C_1, G_2-D_1 and G_1G_2-A_3 and compares its context graph with the included all ancestors, G_1G_2-A_2, G_1G_2-B_1 and G_1G_2-A_1. If B has already all ancestors, it attaches G_1G_2-A_3, G_1-C_1 and G_2-D_1 to G_1G_2-A_2 and G_1G_2-B_1 in its context graph as the leaves. As this example of figure 4 shows, (G_1G_2-A_1, G_1G_2-A_2, G_1G_2-B_1, G_1-C_1, G_2-D_1, G_1G_2-A_3), (G_1G_2-A_1, G_1G_2-B_1, G_1G_2-A_2, G_1G_2-A_3, G_1-C_1, G_2-D_1), and (G_1G_2-A_1, G_1G_2-A_2, G_1G_2-B_1, G_2-D_1, G_1G_2-A_3, G_1-C_1) are all valid causal orderings for each broker, where different participants might see a different ordering.

The summary of immediate predecessor messages at each step in Figure 4.

Each step (each row in Figure 4)	The immediate predecessor messages	Generated messages
(1)	Nothing	G_1G_2-A_1(The Root)
(2)	[G_1G_2-A_1]	G_1G_2-A_2 and G_1G_2-B_1
(3)	[G_1G_2-A_2 , G_1G_2-B_1]	G_1G_2-A_3 , G_1-C_1 and G_1-D_1
(4)	[G_1G_2-A_3 , G_1-C_1]	G_1-C_2

Figure 5. The Information of Predecessors Immediately before the Corresponding Message

Figure 5 shows the summary of the information of the immediate predecessors of the corresponding messages at each gossip round as shown in figure 4. In between brokers, the context graph is sent and received, like as the previous protocol [16]. But from brokers to subscribers, only the immediate predecessor messages of each message are disseminated, instead of all ancestors of the context graph. So, the size of the causal information can be reduced. Row (2) of figure 5 shows the immediate predecessors [G_1G_2-A_1], included with G_1G_2-A_2 and G_1G_2-B_1. Figure 5(3) shows the immediate predecessors [G_1G_2-A_2, G_1G_2-B_1], included with G_1G_2-A_3, G_1-C_1 and G_1-D_1. Figure 5(4) shows the immediate predecessors [G_1G_2-A_3, G_1-C_1], included with G_1-C_2. The least information, that is the immediate predecessors of the corresponding multicast messages, is transmitted from brokers to subscribers for very low cost communication overhead.

Figure 6 shows how in detail broker groups G1={A,B,C} and G2={A,B,D} gossip about the multicast messages including the immediate predecessors of it to subscribers={S1,S2,S3,S4}. In the example of figure 6, there are the context graph of D in G2, the one of C in G1 and the one of B and the one of A in G1 and G2, respectively. Subscriber S1 subscribes to C, S2 and S3 subscribe to A or B, and S4 subscribe to D. In figure 6, let m < m' denote the process that sent or received m' had either sent m or already received m and m and m' are represented in different colors. And let m = m' denote two messages m and m' that are independent of each other have been sent at the same logical time and represented in the same color. Each broker makes causal ordering in what order is G1G2-A1 (red, circle) -> G1G2-A2 (yellow, rectangle) = G1G2-B1 (yellow, diamond) -> G1-C1 (blue, pentagon) = G1-D1 (blue, hexagon). In the case of S1, the causal ordering (G1G2-A1< G1G2-A2 = G1G2-B1 < G1-C1 = G1-D1) is validated. On receiving G1-C1, S1 requests G1G2-B1 to its broker because it knows that G1G2-A2 has already received and G1G2-B1 is not received by verifying the included immediate predecessors [G1G2-A2= G1G2-B1]. In the case of S2 and S3, the causal ordering (G1G2-A1 < G1G2-A2 = G1G2-B1 < G1-C1 = G1-D1) is validated. On receiving G1G2-A2, S2 requests G1G2-A1 to its broker by verifying [G1G2-A1]. On receiving G1-C1, S3 requests G1G2-A2 and G1G2-B1 to its broker. In the case of S4, the causal ordering (G1G2-A1 < G1G2-A2 = G1G2-B1) is validated. On receiving G1G2-B1, S4 requests G1G2-A1 to its broker. As figure 6 shows, the least information of causally ordered delivery is transmitted from brokers to subscribers. It is surely the main cause of highly scalable.

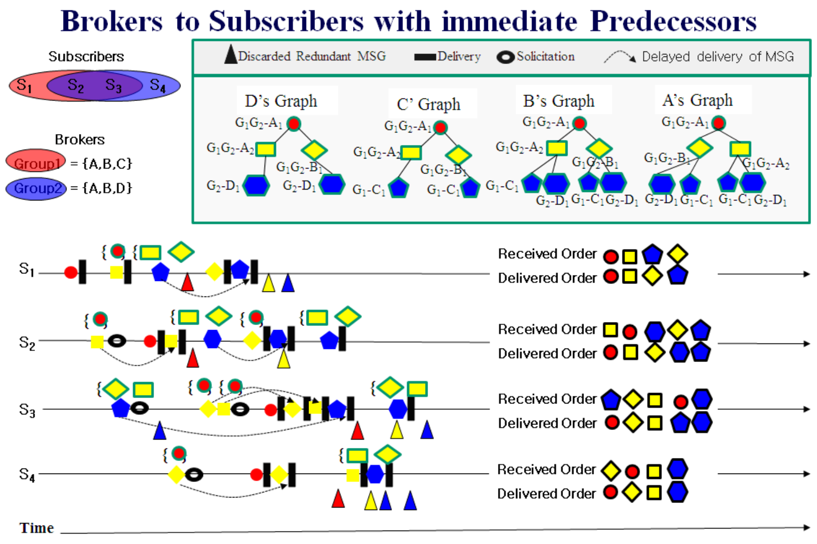

Figure 6. An Example of Immediate Predecessors from Brokers to Subscribers

2.3.2. Proof of the Protocol

Define:

Let p_i denote a participant process, M denote the set of messages processes exchange with each other, < (read "precedes") be a transitive relation on M and m_j be a member of M. Let E be a set of edges whose element is (m_j, m_k) and $G_a = (M_a, E_a)$ denote the directed acyclic graph representation of < in every process of a group G.

Rule 1:

If G_1G_2-m' < G_1G_2-m, attach an every E (G_1G_2-m' , G_1G_2-m) in G_a.

1. Causality is never violated (Safety).

Proof:

Consider the actions of a process p_j, that is involved in both overlapping groups G_1 and G_2 and receive two messages G_1G_2-m_1 and G_1G_2-m_2 such that G_1G_2- m_1 < G_1G_2-m_2.

Case 1:

G_1G_2-m_1 and G_1G_2-m_2 are both transmitted by the same process p_j. This case is trivial. Under rules, in the protocol, G_1G_2-m_2 can only be delivered after G_1G_2-m_1 has been delivered.

Case 2:

G_1G_2-m_1 and G_1G_2-m_2 are transmitted by two distinct processes p_i and p_i'. By induction on the messages received by p_j, we will prove that G_1G_2-m_2 cannot be delivered before G_1G_2-m_1. Assume that G_1G_2-m_1 has not been delivered and p_j has received *k* messages. Since G_1G_2-m_1< G_1G_2-m_2, we have an E (G_1G_2-m_1, G_1G_2-m_2) ----------------- Relation (1)

Base step:

The first message delivered by p_j cannot be G_1G_2-m_2. Recall that if no messages have been delivered to p_j, there are no edges and predecessors in G_{pj}. There is no immediate predecessor of G_1G_2-m_2 in G_{pj}. By the rules, G_1G_2-m_2 cannot be delivered by p_j.

Inductive step:

Suppose p_j has received *k* messages, none of which is a message G_1G_2-m such that G_1G_2-m_1< G_1G_2-m. If G_1G_2-m_1 has not yet been delivered, then there is no edge, E (G_1G_2-m_1's predecessor, G_1G_2-m_1) and there is no node, an immediate predecessor, G_1G_2-m_1 in G_{pj}. ------------------- Relation (2)

If a new node, G_1G_2-m_1 is a leaf, there can be an edge from G_1G_2-m_1's predecessor to G_1G_2-m_1 comprising of one-length, hence, E (G_1G_2-m_1's predecessor, G_1G_2-m_1). Therefore, if we can see the case that the existence of a path comprising of longer than one-length edge from G_1G_2-m_1's predecessor to G_1G_2-m_2 implies, then we cannot see the case that there exist any path comprising of one-length edge from G_1G_2-m_1's predecessor to G_1G_2-m_2 in G_{pj}. From Relations (1) and (2), it follows that there is no immediate predecessor of G_1G_2-m_2 in G_{pj}. By application of the rules of the protocol, the *k+1*st message delivered by p_j cannot be G_1G_2-m_2.

2. Every message in overlapping groups is indeed delivered (Liveness).

Proof:

Suppose there is a multicast message G_1G_2-m sent by process p_i in overlapping groups G_1 and G_2 that can never be delivered to process p_j. The rules of the protocol imply that the number of immediate predecessors happened before G_1G_2-m in G_{pj} is smaller than the number of immediate predecessors piggybacked on G_1G_2-m. The number of messages that must be delivered to p_j before G_1G_2-m is finite. Communication links are fair-loss, but correct processes can construct reliable communication links on top of fair-loss links by periodically retransmitting messages. So, in the absence of failures and after some finite time, all these messages will have arrived at p_j. If every such message had been delivered, then we would have known that the number of immediate predecessors happened before G_1G_2-m in G_{pj} is bigger than the number of immediate predecessors piggybacked on G_1G_2-m and G_1G_2-m could be delivered; contradiction.

So, there exists at least another message G_1G_2-m' which will not be delivered to p_i and should be before G_1G_2-m. If G_1G_2-m' is in a waiting buffer, then the number of immediate predecessors piggybacked on G_1G_2-m' is smaller than that of immediate predecessors piggybacked on G_1G_2-m. We can thus apply the same reasoning to G_1G_2-m' as to G_1G_2-m, which completes the proof by finite decreasing induction.

3. Performance Evaluation

We consider a system composed of a finite set of brokers (publishers) that communicate by message passing. In the system, there is also a set of subscribers. For simplicity, we assume that there is not more than one broker per node of the network. Brokers and subscribers can be implemented in the context of topic-based P/S like as [4] and joining/leaving can be viewed as subscribing/unsubscribing from the topic. The system is asynchronous. Brokers can only fail by crashing (i.e., we do not consider Byzantine failures). A broker that never fails is correct. For simplicity, we do not include process recovery in the model. We assume further that failures are independent. The probability of a message loss does not exceed a $\varepsilon > 0$, $\varepsilon=0.05$. The number of broker crashes does not exceed $f < n$. The probability of a broker crash during a run is bounded by $\tau = f / n$, $\tau=0.01$. At every round, each broker has an independent uniformly distributed random view of size l of subscribers like as [4], [6]. These views called as uniform views are not constant, but continue evolving. The subset of subscribers in the uniform view to which a broker gossips a message, are chosen randomly according to a uniform distribution.

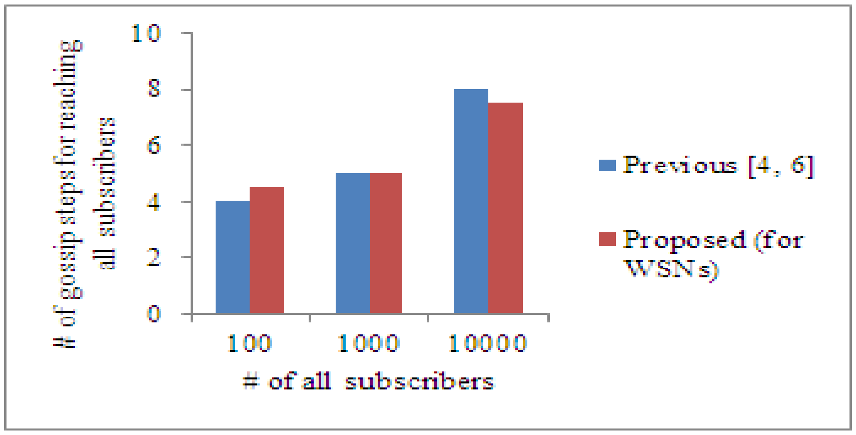

Fig. 7. The Number of Gossip Steps with Increasing the Number of Subscribers

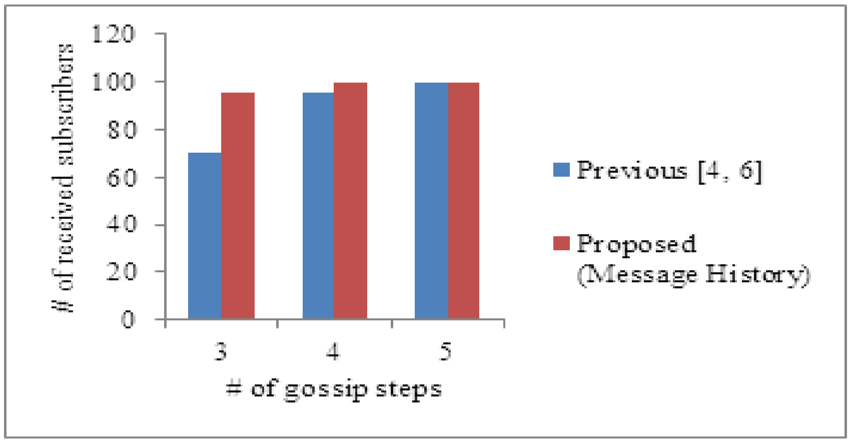

Figure 8. The Number of Gossip Steps after the Message has been sent

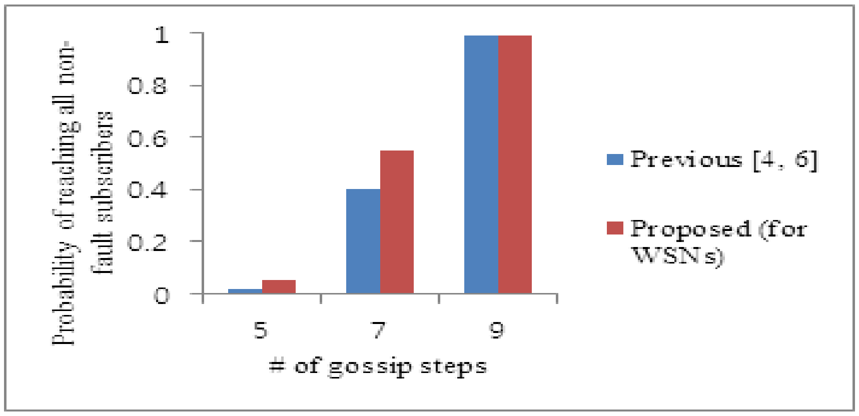

Fig. 9. Probability that All Non-fault Subscribers Received a Message after the Number of Gossip Steps

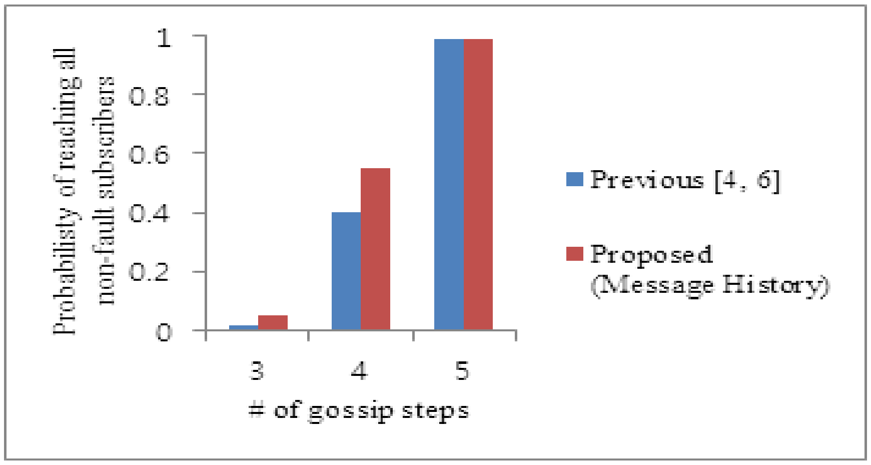

Figure 10. Probability that All Non-fault Subscribers Received a Message after the Number of Gossip Steps

We are interest in how our proposed protocols, based on P/S systems with difference causal order information between communication groups, from brokers and to brokers and from brokers and subscribers, scale better than other probabilistic protocols without distinguishing P/S ones like as [4, 6]. We try to attempt to capture the degree of scalability achieved with our proposed protocols and confirm the results obtained from our simulations in terms of the analysis presented in [4, 6]. Figure 7 shows the expected number of gossip steps necessary to reach all subscribers as a function of the number of subscribers. The number of steps increases with the logarithm of the number of subscribers, which is also demonstrated in [6]. But we can see our proposed protocol especially for WSNs needs smaller steps than the previous one [4, 6]. Also, figure 8 shows the expected number of subscribers reached by a message *m* after a finite gossip steps grows. We can see that our proposed protocol using minimal history information converges faster than the previous one [4, 6]. As expected like in [6], the eventual convergences is captured with probability ≒1. We can see that our proposed protocols are faster than the previous ones [4, 6]. And also we are interested in that as a function of the number of gossip steps after the message has been sent, the probability of receiving all non-fault subscribers converges to 1 at different information of causal ordering. According to the analysis presented in [6], the probability that all subscribers have received a message, converges to 1 as the number of gossip steps grows, we try to capture how our proposed protocols converge faster than other protocols [4, 6] with reasonable reliability. In figure 9 and figure 10, we can see that in our two proposed protocols, one is with the minimal history information and the other is used for WSNs, all non-fault subscribers can receive messages and terminate the gossip round in which the message has been processed faster than those in other protocols [4, 6]. Therefore, our proposed protocols are very scalable with reasonable reliability than the previous ones [4, 6].

4. Related Works

There have been a large number of academic researches on P/S systems, classified into topic-based, attribute-based, and content-based depending on the matching model. In many P/S systems designed for enterprise environments, subscribers establish affinity with brokers and connect to their chosen brokers. A subscriber sends subscriptions and receives the messages matching their interests published at its chosen broker. Content-based P/S networks scale to large numbers of publishers and subscribers by having brokers summarize subscriptions from subscribers and downstream brokers based on coverage relationships between subscriptions. In some P/S systems like completely decentralized P2P without dedicated brokers, structured overlay techniques like DHT (CAN [11], PASTRY [13], and CHORD [14]) for distributing subscriptions and messages in dynamics such as node churns and unreliable links are usually used. PASTRY [13] uses routing based on address prefixes built over distributed index trees, CHORD [14] forwards messages based on numerical differences with their destinations and CAN [11] routes messages in a d-dimensional space. In comparison, there is much less dynamics in a cloud-based P/S.

And there are researches based on the P (publish)/S (subscribe) paradigm in the area of sensor network communications to approach the problem of querying sensors from mobile nodes [7, 17]. Directed Diffusion [7] can be seen as publish-subscribe mechanism, which is implemented using the tree-based architecture rooted at the publisher. SENSTRACT [17] is mapping from queries to topics and the corresponding underlying sensor network structure. SENSTRACT [17] is a tree-based P/S system structured by service providers as roots, representing one of the data-centric routing protocols for data dissemination of sensor networks. Cross Reality is about connecting "location-specific 3D animated constructs" in virtual worlds to in-building sensors [8]. The global behavior of the WSN constructed with limited functionality of sensors is achieved, in part, through data fusion, which often depends on the time of occurrence of fused sensor readings. So, recently, protocols for physical time synchronization in sensor networks have been published [18]. One of our proposed protocol is based on gossip protocols for completely decentralized distributed applications, such as an environment of [19] and guarantees causal message ordering, which is somewhat similar to temporal message ordering [12], based on logical time not physical time. [19] is based on gossip protocols and firefly synchronization [18], for the management policy distribution and synchronization over a number of nodes in an application level.

5. Conclusions

In this paper, we present two versions of broker-based causal order multicast protocols in P/S systems. In the protocol based on local views of gossiping, each broker sends and receives the multicast message including all ancestors of the context graph. But, from brokers to subscribers, each broker disseminates the multicast message including only the immediate predecessors instead of all ancestors. The immediate predecessors are in the structure of the one-dimensional vector, while all ancestors are the two-dimensional of the context graph. In the first step of sending a message, when every broker generates a multicast message, it puts its ID, the sequence number and the group lists of all groups that it participates on the message. In the second step, the broker attaches the message to all leaf nodes in its context graph. Then, the multicast message becomes the leaf and the parent messages of it become the immediate predecessors. In the last step, the broker sends the message including all ancestors to other brokers, but it including only the immediate predecessors, instead of all ancestors of the context graph to subscribers. In the protocol based on global views of gossiping, every broker sends and receives the multicast message including the whole set of group vectors. But, from brokers to subscribers, each broker disseminates the multicast message including only the immediate predecessors instead of the whole set of group vectors. The immediate predecessors are in the structure of the one-dimensional vector, while the whole set of group vectors are two-dimensional structure. In the first step of sending a message, when every broker generates a multicast message, it posts the current gossip round timestamp on it. In the second step, the broker updates every element of the whole set of group vectors. Then the broker updates all elements of the immediate predecessor vector. In the last step, the broker sends the message including the whole set of group vectors to other brokers, but it including only the immediate predecessor vector of length N, where N is the number of all brokers, instead of the whole set of group vectors. These features might result in its very low cost communication overhead between brokers and subscribers because there is difference causal order information between communication groups, from brokers and to brokers and from brokers and subscribers. Therefore, these two versions of the proposed protocol might be significantly scalable in P/S applications requiring only the minimum causal information of message delivery with flexible consistency.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (grant number 2012R1A1A2044660).

References

[1] K. Birman, A. Schiper and P. Stephenson, "Lightweight Causal and Atomic Group Multicast", ACM Transactions on Computer Systems, vol. 9, no. 3, (**1991**), pp. 272-314.

[2] K. Birman, M. Hayden, O. Ozkasap. Z. Xiao, M. Budiu and Y. Minsky, "Bimodal Multicast", ACM Transactions on Computer Systems, vol. 17, no. 2, pp. 41-88, (**1999**).

[3] K. Birman, Q. Huang and D. Freedman, "Overcoming CAP with Consistent Soft-State Replication", IEEE Internet Computing, vol. 12, (**2012**) February, pp. 50-58.

[4] P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov and A.-M. Kermarrec, "Lightweight probabilistic broadcast", ACM Transactions on Computer Systems, vol. 21, no. 4, (**2003**), pp. 341-374.

[5] D. Eyers, T. Freudenreich, A. Margara, S. Frischbier, P. Pietzuch and P. Eugster, " Living in the present: on-the-fly information processing in scalable web architectures", In CloudCP, (**2012**).

[6] P. Felber and F. Pedone, "Probabilistic Atomic Broadcast", in Proceedings of 21st IEEE Symposium on Reliable Distributed Systems (SRDS'02), Osaka, Japan, (**2002**) October, pp.170-179.

[7] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", in Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00), August (**2000**), pp. 56-67, Boston, MA.

[8] J. Lifton, M. Laibowitz, D. Harry, N.-W. Gong, M. Mittal, J. and A. Paradiso, "Metaphor and Manifestation—Cross-Reality with Ubiquitous Sensor/Actuator Networks", IEEE Pervasive Computing, vol. 8, no. 3, (**2009**), pp. 24-33.

[9] A. Mostefaoui and M. Raynal, "Causal Multicasts in Overlapping Groups: Towards a Low Cost Approach", In Proceedings of the 4th IEEE International Conference on Future Trends in Distributed Computing Systems, (**1993**), pp. 136-142.
[10] A. Margara, S. Frischbier, T. Freudenreich, P. Eugster, D. Eyers and P. Pietzuch, "ASIA: Application-specific integrated aggregation for publish/subscribe systems", Technical report, (**2012**).
[11] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, "A scalable content addressable network", In SIGCOMM, (**2001**).
[12] K. Römer, "Temporal Message Ordering in Wireless Sensor Networks", IFIP MedHocNet, Mahdia, Tunisia, (**2003**) June, pp. 131-142.
[13] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", In Middleware "'01.
[14] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", In SIGCOMM "'01.
[15] K. Ostrowski, K. Birman, and D. Dolev, "QuickSilver Scalable Multicast", 7th IEEE International Symposium on Network Computing and Applications (IEEE NCA 2008), (**2008**) July, pp. 9-18, Cambridge.
[16] L. Peterson, N. Buchholzand and R. Schlichting, "Preserving and using context information interprocess communication", ACM Transaction Computer Systems, vol. 7, no. 3, (**1989**), pp. 217-246.
[17] S. Pleisch and K. Birman, "SENSTRAC: Scalable Querying of SENSor Networks from Mobile Platforms Using TRACking-Style Queries", International Journal of Sensor Networks, vol. 3, no. 4, (**2008**), pp. 266-280.
[18] A. Tyrrell, G. Auer and C. Bettstetter, "Fireflies as Role Models for Synchronization in Ad Hoc Networks", In Proc. Intern. Conf. on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS), Article No. 4, Cavalese, Italy, (**2006**) December.
[19] I. Wokoma I. Liabotis O. Prnjat L. Sacks and I. Marshall, "A Weakly Coupled Adaptive Gossip Protocol for Application Level Active Networks", In proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02), pp. 244 - 247, Monterey, California, (**2002**) January.
[20] J. Yick, B. Mukherjee and D. Ghosal, "Wireless sensor network survey", Computer Networks, vol. 52, no. 22, (**2008**) August, pp. 2292-2330.

Authors

Chayoung Kim, she received B.S. and M.S. degrees from the Sookmyung Women's University, Seoul, Korea, in 1996 and 1998, respectively and Ph.D. degree from the Korea University in 2006. From 2005 to 2008, she was a senior researcher in Korea Institute of Science and Technology Information, Korea, where she has been engaged in National e-Science of Supercomputing Center. From 2009 to 2012, she was a researcher at Contents Convergence Software Research Center in Kyonggi University, Korea. Since 2012, she has been an adjunct professor in Department of Computer Science, Kyonggi University, Korea. Her research interests include distributed computing, group communications and peer-to-peer computing.

Jinho Ahn, she received his B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Korea University, Korea, in 1997, 1999 and 2003, respectively. He has been a full professor in Department of Computer Science, Kyonggi University. He has published more than 70 papers in refereed journals and conference proceedings and served as program or organizing committee member or session chair in several domestic/international conferences and editor-in-chief of journal of Korean Institute of Information

Technology and editorial board member of journal of Korean Society for Internet Information. His research interests include distributed computing, fault-tolerance, sensor networks and mobile agent systems