Systematic Approach for Scheduling of Tasks and Messages under Noise Environment

Hyoung Yuk KIM, Hye Min SHIN, and Hong Seong PARK

Dept. of Electrical and Computer Eng., Kangwon National University
192-1 Hyoja 2 Dong, Chuncheon, 200-701, Korea

Abstract

High degree of EMI (Electro-Magnetic Interference) or noise is generated in the plant environment, where control systems consist of smart sensors, smart actuators, and controllers connected via a fieldbus. The noise generated by some devices such as high-power motors may cause communication errors and delay the successful transmission of data. Therefore, the noise condition is one of the important parameters considered in the design of a reliable network-based control system. This paper presents the scheduling method of task and message to guarantee the given end-to-end constraints under noise environments. The presented scheduling method is applied to an example of a control system that uses CAN (Controller Area Network), considering two kinds of noise models. The comparison results for each noise condition shows the importance of considering the noise condition in system design. System designers are able to design the control system, guaranteeing its requirements under a noise environment by using the proposed scheduling method.

Keywords: Noise, Real-Time, Scheduling, Fieldbus.

1. INTRODUCTION

Nowadays, control systems generally consist of smart sensors and smart actuators, which are connected with controllers via fieldbuses such as Profibus, FIP, CAN, and LonWorks. Because sensors, actuators, and controllers may be distributed, data transmission from sensor to controller and from controller to actuator may be randomly delayed due to network characteristics. Also, the task execution time at a node depends on the node's characteristics such as the operating system, the number of its tasks, and so on. Therefore, the network-induced random delay and the variable task execution time should be considered in system design [1-3]. For example, if task A for transmitting a message to another node has lower priority than task B for diagnostics at the same node but starts simultaneously, the former is completed after the latter is finished. This way, the worst-case response time is varied according to the task's priority. In addition, message transmission can be delayed if its priority is lower than the priorities of other messages that are competing to use the available media. In other words, the worst-case response times depend on the priorities of tasks and messages.

There are several researches on the system scheduling method for solving problems mentioned above [4-8]. A heuristic scheduling method is proposed in [4], which assumes that any one task cannot receive more than one message. A scheduling method using several pruning algorithms is proposed in [6], which extends a task-based scheduling method [5] for distributed control systems. In [7], a method for scheduling distributed systems consisting of CSMA/CA and TDMA networks is proposed. This method uses a genetic algorithm and a clustering algorithm. In [8], a period assignment method for tasks and a heuristic assignment rule for message's priority are proposed, and the scheduling method, which uses the previous two methods, for a practical distributed control system consisting of multiple control loops is proposed. The above research assumed error-free or noise-free networks. In other words, they cannot be applied to plants under noise environments.

Most distributed control systems are installed and operated in the plant with generally many high-power motors. Therefore, high degrees of Electro Magnetic Interferences (EMIs) from these motors strongly affect the control systems [9,10], and these interferences may cause the transmission errors of data. The transmission errors may lead to performance degradation or serious faults in the control systems because they make the successful transmission of data delayed. For example, if the end-to-end response time of a control loop increases due to the transmission errors, the control system might miss the control loop's deadline. Therefore, when a distributed control system operated under noise environment is designed, the transmission error due to noise should be considered to satisfy given real-time constraints. Therefore, a new method for designing distributed control systems under noise environment is required.

There is some research on message transmission delay caused by errors. An analysis method on the effect of noise is presented in [11]. It uses an error model based on the bit error rate of simple sporadic noise and presents the analysis method of a message's response time transmitted via CAN under this noise model. Also, a method that can consider multiple noise sources simultaneously is presented in [9]. But, these research however have analyzed only the message transmission delay affected by the transmission errors and have not considered the task execution delay caused by precedence relations between tasks and messages and the end-to-end delay from sensors to actuators caused by transmission errors.

This paper proposes the scheduling method of task and message for control systems operating under noise environment and verifies the proposed scheduling method by applying it to an example of a distributed control system connected via CAN. The proposed method schedules the periods and the priorities of

tasks and messages to satisfy the given end-to-end constraints under noise environment and extends the scheduling method proposed in [8] to consider the communication errors caused by noise. We used two kinds of noise models adopting multi-sources for the noise. The scheduling results, namely the system's end-to-end response times and the messages' response times, are derived for each noise model.

In Section 2, a noise model adopting multi-noise sources and an analysis method for the message response times are described. The proposed scheduling method is applied to the example system, and the analysis of the scheduling results is presented in Section 3. Finally, conclusions are given in Section 4.

2. NOISE MODEL AND DELAY TIME

2.1 Problems caused by noise

Most control systems operate in plants using high-power motors. In this environment, high degrees of EMI may generate communication errors. These errors delay the message transmission between nodes and may cause critical faults in the control systems. This problem is illustrated in Fig.1. In this figure, there are two control loops: The 1st control loop consists of SensorNode1, SensorNode2, Controller1, and ActuatorNode1; the 2nd control loop consists of SensorNode1, SensorNode3, Controller2, and ActuatorNode2. Each node has only one task. All tasks are invoked simultaneously, and control tasks and actuator tasks wait until the input messages are received. The messages are prioritized as follows: SensorTask1, followed by SensorTask2, SensorTask3, ControlTask1, and ControlTask2. It is assumed that the deadline of each control loop is equal to each control task's period.

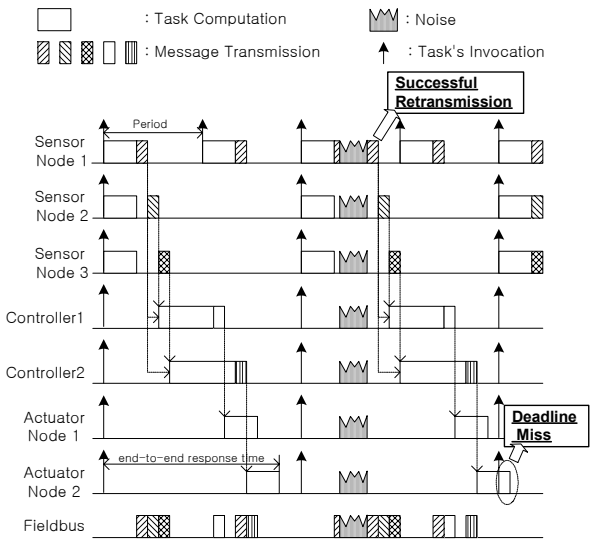

Fig.1 End-to-end response time delayed by noise

As shown in Fig.1, the end-to-end constraints from sensor sampling to actuator outputting are satisfied within the deadline of each control loop when there is no noise. If noise causes errors at that time the SensorTask1's message is sent, an error recovery mechanism such as retransmission method as shown in Fig.1 can correct the message transmission. Because of this communication delay, ControlTask1 and ControlTask2, which wait for a message from SensorTask1, are delayed. Actuator Task1 and ActuatorTask2 are also delayed. Finally, the end-to-end response time of the 2nd control loop becomes longer than its deadline. This increase in response time indicates the occurrence of a critical fault in the control system. Therefore, communication errors caused by noise should be considered in design of distributed control system operating under noise environment to guarantee the real-time requirements.

2.2 Noise model and delay time [9]

To consider noise in system design, a proper noise model, which can analyze the effects of noise on the system, should be considered. We used a noise model that had multiple noise sources [9], as shown in Fig.2.

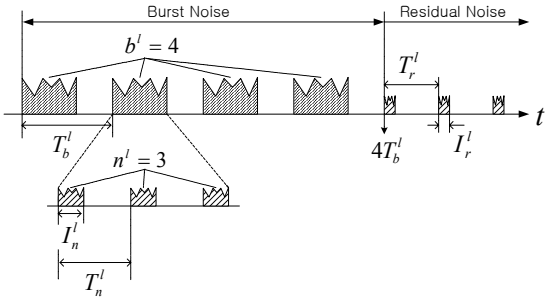

Fig.2 Noise model

Fig.2 represents one noise source l, which consists of a burst noise group and a residual noise group. The burst noise group is composed of b^l subgroups being generated with the minimum period T_b^l, and each subgroup consists of n^l noises occurring with the minimum period T_n^l during I_n^l. The residual noises occur with the period T_r^l during I_r^l after the burst noise. Using this noise model, communication delay time can be derived from error detection and recovery mechanism of the corresponding network.

If a noise which can be represented by this model is generated, we can get the number of noises occurring for the period t by calculating the number of burst noises $Bu^l(t)$ and that of residual noises $Re^l(t)$ separately. If the period t includes the part of the residual noise, $Bu^l(t)$ is equal to $n^l * b^l$. Otherwise, $Bu^l(t)$ is equal to the sum of the number of noises of the subgroups wholly included in the period $t \left\lfloor t/T_b^l \right\rfloor \times n^l$ and that of noises of the subgroups partly included in the period $t \left\lfloor t \bmod T_b^l / T_n^l \right\rfloor$. Therefore, $Bu^l(t)$ is equal to (1).

$$Bu^l(t) = \min\left(n^l \times b^l, \left\lfloor \frac{t}{T_b^l} \right\rfloor \times n^l + \min\left(n^l, \left\lceil \frac{t \bmod T_b^l}{T_n^l} \right\rceil \right) \right) \qquad (1)$$

If the period t includes only the part of burst noises, the number of residual noises $Re(t)$ is 0. If the period t includes the part of residual noises, we can get $Re(t)$ as follows:

$$\mathrm{Re}^{l}(t) = \max\left(0, \left\lceil \frac{t - T_b^l \times b^l}{T_r^l} \right\rceil\right) \quad (2)$$

Let $E_i(t)$ be the transmission delay time of message i caused by total number of noises during period t and $E_i^l(t)$ be that of message i caused by noise source l during the period t. If there are k noise sources, $E_i(t)$ is equal to (3).

$$E_i(t) = E_i^1(t) + E_i^2(t) + \ldots + E_i^k(t) \quad (3)$$

If communication errors occur in the CAN, the worst-case response time of message can be calculated by the presented noise model and (1) ~ (3). The overhead of message i O_i caused by one error bit in the CAN consists of 31 error recovery bits and the message retransmission [6,7] and is represented as (4).

$$O_i = 31 \times \tau_{bit} + \max_{k \in hp(i) \cup \{i\}} (C_k) \quad (4)$$

τ_{bit} is the time for transmitting one bit, and $hp(i)$ is a set consisting of messages with higher priority than message i. C_k is the real transmission time of message i. In order to evaluate in the worst-case, the maximum transmission time between corresponding messages is used. Using (4), $E_i^l(t)$ is the product of the overhead and the number of error bits by burst noises and residual noises. If noise duration of burst noise part I_b^l and that of residual one I_r^l are longer than one bit time, the differences should be compensated. Therefore, $E_i^l(t)$ is equal to (5).

$$\begin{aligned} E_i^l(t) = {} & Bu^l(t) \times (O_i + \max(0, I_b^l - \tau_{bit})) + \\ & \mathrm{Re}^l(t) \times (O_i + \max(0, I_r^l - \tau_{bit})) \end{aligned} \quad (5)$$

The transmission delay of message i is influenced by all noises occurring from the time that the message is inserted into the transmission queue of CAN to the time that the message is transmitted completely. Therefore, the transmission delay by noise is included in the queuing delay of message i q_i^E as shown in (6).

$$q_i^E = B_i + \sum_{j \in hp(i)} \left(\left\lceil \frac{q_i^E + \tau_{bit} + J_i}{T_j} \right\rceil \times C_j \right) + E_i(q_i^E + C_i) \quad (6)$$

B_i is the maximum blocking time for which the message i can be delayed by the messages with lower priority than its priority. J_i is a jitter that occurs when message i is inserted into the transmission queue of CAN, and T_j is the period of message i. In the CAN, the worst-case response time of message i R_i^E - from the time that the message is inserted into the transmission queue of CAN to the time that the message is transmitted completely – consists of jitter for inserting into the queue J_i, the queuing delay q_i^E, and the message transmission time C_i and is shown in (7).

$$R_i^E = J_i + q_i^E + C_i \quad (7)$$

The worst-case response time of message i R_i^E under noise condition should be equal or less than the message's deadline Ed_i. Therefore, the relation $R_i^E \le Ed_i$ is established, and R_i^E can be used as the predictable deadline of message i.

3. SCHEDULING OF TASKS AND MESSAGES UNDER NOISE ENVIROMENT

3.1 Task and message scheduling

Because the priority and the period of message and task have a very important effect on the real-time characteristics of distributed control systems, system scheduling must assign the priority and the period guaranteeing the system's real-time requirements in the system design. The scheduling process proposed in [8] consists of four steps and is shown in Fig.3. In step 1, a task graph is designed according to a system model and its given requirements, and then each task is allocated to a physical node. In step 2, the equations or the inequalities of constraints on tasks and messages are derived from the task graph. In step 3, a priority and a period are allocated to each task and message by the priority and period assignment algorithm. In step 4, the derived equations or inequalities are solved with the worst-case response times of messages and tasks. If the solved results do not meet the end-to-end constraints, this method goes back to step 3 and changes some of the parameters of tasks and messages, such as period or priority. This process is repeated until the calculated results satisfy the end-to-end constraints. If the period and the priority that meet the constraints cannot be found, the scheduling process returns to step 1 and changes the system's requirements or redesigns the task graph.

The scheduling method presented above is extended to consider the noise condition in the system scheduling, and the extended method is shown in Fig.3. The additional processes are the noise modeling of an actual spot at the time of system requirement derivation and the derivation of delay time equations due to noise at the time of constraint derivation. Finally, the derived delay time equations are used to solve derived constraints after the periods and the priorities are assigned. With this method, one can schedule the system, considering noise condition of the spot.

Because nodes or stations can be distributed within the distributed control systems, noise characteristics vary according to where these are installed. For example, if nodes or communication media are installed near high power motors, these nodes or media may be influenced by high power noise. But, if they are installed at a place without a noise source, their error rate may be very low or zero. Therefore, if noise conditions can be applied differently depending on the spots, systems can be designed more suitably. But this paper leaves this consideration to future works and assumes noise conditions are same within the systems.

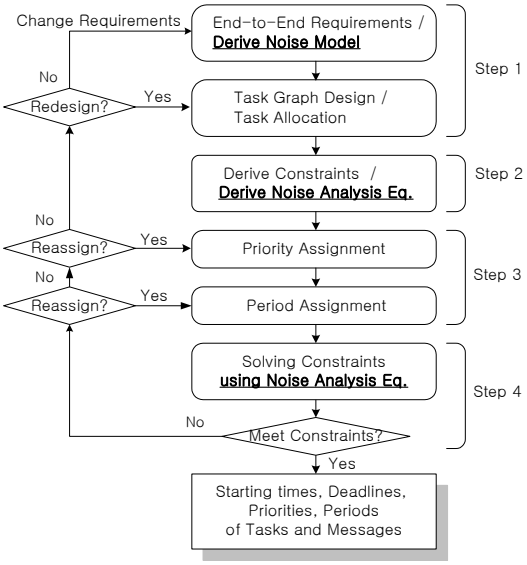

Fig.3 Task and message scheduling under noise environment

3.2 Notations

The following notations will be used throughout this paper.

τ_α^i the task α included in the *i*-th control loop (*i*=1~3) and α ∈{*Sj*, *C*, *Ak*, *ml, Om* } where *Sj*, *C*, *Ak*, *ml*, and *Om* denote the *j*-th sensor task (*j*=1~6), a control task, the *k*-th actuator task (*k*=1~5), the *l*-th message task (*l*=1~11), and the *m*-other task exiting in a control node but not being related to control loop (*m*=1~3) respectively.

$\tau_\alpha^{i,j}$ the task α that is included in the *i*-th control loop and the *j*-th control loop at the same time and α is equal to the definition of τ_α^i

e_α^i the execution time of τ_α^i.

T_α^i the period of τ_α^i.

d_α^i the deadline of τ_α^i.

p_α^i the priority of τ_α^i.

ϕ_α^i the initial phase time or starting time of τ_α^i.

π_β^i the port for the message β included in the *i*-th control loop and $\beta \in$ {*Sj*, *Ck*, *ml* } where *Sj*, *Ck*, and *ml* denote the message of the *j*-th sensor task (*j*=1~6), the *k*-th message of a control task (*k*=1,2), and the message of the *l*-th message task (*l*=1~11) respectively.

$\pi_\beta^{i,j}$ the port for the message β that is included in the *i*-th control loop and the *j*-th control loop at the same time and β is equal to the definition of π_β^i.

$MADT^i$ the end-to-end maximum allowable delay time of the *i*-th control loop

3.3 System model and task graph

An example system to which the proposed scheduling method is applied and its task graph are shown in Fig.4 and Fig.5, respectively. The system consists of 6 sensor nodes, 3 controllers, and 5 actuator nodes that form three control loops. The 1st control loop is composed of SensorNode1, 2, 3, 4, Controller1, and ActuatorNode1, 2; the 2nd control loop is SensorNode3, 4, Controller2, and ActuatorNode3; and the last control loop consists of SensorNode4, 5, 6, Controller3, and ActuatorNode4, 5. All nodes are connected via CAN.

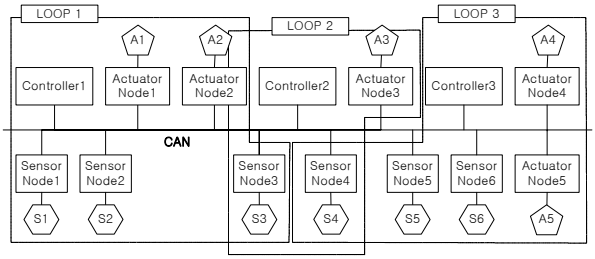

Fig.4 Target System

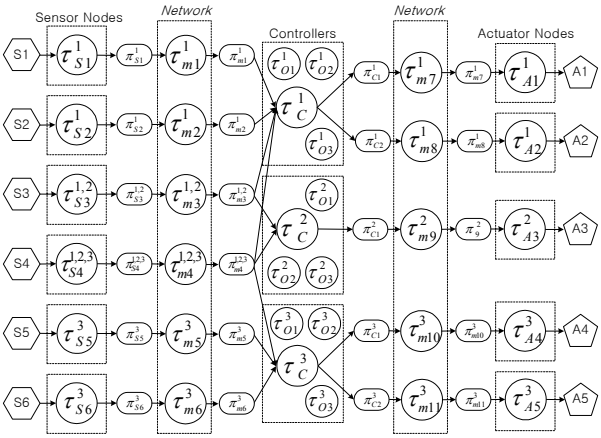

Fig.5 Task graph of target system

The parameters of system requirements previously given before scheduling the system are summarized in Table 1. There are the end-to-end deadlines of three loops, the execution time of each task, the size of each message, and the information of the tasks that exist in the controllers but are not included in the control loops.

Table 1 System parameters

MADT, [ms]	$MADT^1 = 60,\ MADT^2 = 80,\ MADT^3 = 100$
Execution Time of Tasks, [ms]	$e_{S1}^1=1,\ e_{S2}^1=1,\ e_{S3}^{1,2}=1,\ e_{S4}^{1,2,3}=1,\ e_{S5}^3=1,\ e_{S6}^3=1,$ $e_C^1=7,\ e_{O1}^1=4,\ e_{O2}^1=6,\ e_{O3}^1=8$ $e_C^2=9,\ e_{O1}^2=4,\ e_{O2}^2=6,\ e_{O3}^2=8,$ $e_C^3=11,\ e_{O1}^3=4,\ e_{O2}^3=6,\ e_{O3}^3=8,$ $e_{A1}^1=1,\ e_{A2}^1=1,\ e_{A3}^2=1,\ e_{A4}^3=1,\ e_{A5}^3=1$
Message Length, [Byte]	$\tau_{m1}^1=2,\ \tau_{m2}^1=2,\ \tau_{m3}^{1,2}=8,\ \tau_{m4}^{1,2,3}=8,\ \tau_{m5}^3=2,\ \tau_{m6}^3=2,$ $\tau_{m7}^1=2,\ \tau_{m8}^1=2,\ \tau_{m9}^2=2,\ \tau_{m10}^3=2,\ \tau_{m11}^3=2$
Period of Tasks, [ms]	$T_{O1}^1=20,\ T_{O2}^1=50,\ T_{O3}^1=60$ $T_{O1}^2=20,\ T_{O2}^2=50,\ T_{O3}^2=60,$ $T_{O1}^3=20,\ T_{O2}^3=50,\ T_{O3}^3=60$

In Table 2, two kinds of noise models-CASE1 and CASE2-to be applied to the target system are represented, using the method presented in Section 2.2. In case of CASE2, two noise sources exist simultaneously. The operational

characteristics of the target system under two kinds of noise conditions were analyzed and compared. The results showed that the proposed scheduling method could schedule the system under these noise conditions. Also, the compared scheduling results showed that the effect of noise was important to control systems.

Table 2 Error models

	Noise Source	b^l	n^l	I^l	I_r^l	T_f^l	t_f^l	r_f^l
CASE1	1	1	4	0.5	0.5	4	0.01	10
CASE2	1	4	2	0.1	0.1	1	0.001	10
	2	2	2	0.1	0.1	1	0.001	20

3.4 Deriving Constraints

After deriving the system requirements and the noise model, the system's task graph is drawn, and each task is assigned to each node. And then, equations or inequalities representing the system requirements and the relation of task and message are derived from the task graph. The derived equations or inequalities are used as constraints at the system scheduling. Constraints consist of one for the end-to-end response time of control loops, one for the period relations between tasks and messages, and one for the precedence relations between tasks and messages.

The constraints for the 1st control loop are as follows: The end-to-end time constraint is

$$\max(\phi_{A1}^1 + d_{A1}^1, \phi_{A2}^1 + d_{A2}^1) \le MADT^1 .$$

The period constraints of tasks and messages are

$$T_{S1}^1 = T_{m1}^1,\ T_{S2}^1 = T_{m2}^1,\ T_{S3}^{1,2} = T_{m3}^{1,2},\ T_{S4}^{1,2,3} = T_{m4}^{1,2,3},$$
$$T_{m1}^1 \mid T_C^1,\ T_{m2}^1 \mid T_C^1,\ T_{m3}^{1,2} \mid T_C^1,\ T_{m4}^{1,2,3} \mid T_C^1,\ T_C^1 = T_{m7}^1 = T_{A1}^1,$$
$$T_C^1 = T_{m8}^1 = T_{A2}^1 .$$

The precedence constraints of tasks and messages are

$$\phi_{S1}^1 + d_{S1}^1 \le \phi_{m1}^1,\ \phi_{S2}^1 + d_{S2}^1 \le \phi_{m2}^1,\ \phi_{S3}^{1,2} + d_{S3}^{1,2} \le \phi_{m3}^{1,2},$$
$$\phi_{S4}^{1,2,3} + d_{S4}^{1,2,3} \le \phi_{m4}^{1,2,3},$$
$$\max(\phi_{m1}^1 + Ed_{m1}^1, \phi_{m2}^1 + Ed_{m2}^1, \phi_{m3}^{1,2} + Ed_{m3}^{1,2}) \le \phi_C^1,$$
$$\phi_{C1}^1 + d_{C1}^1 \le \phi_{m7}^1,\ \phi_{C1}^1 + d_{C1}^1 \le \phi_{m8}^1,\ \phi_{m7}^1 + Ed_{m7}^1 \le \phi_{A1}^1,$$
$$\phi_{m8}^1 + Ed_{m8}^1 \le \phi_{A2}^1 .$$

The constraints about the 2nd control loop are as follows: The end-to-end time constraint is

$$\phi_{A3}^2 + d_{A3}^2 \le MADT^2 .$$

The period constraints of tasks and messages are

$$T_{S3}^{1,2} = T_{m3}^{1,2},\ T_{S4}^{1,2,3} = T_{m4}^{1,2,3},\ T_{m3}^{1,2} \mid T_C^2,\ T_{m4}^{1,2,3} \mid T_C^2,\ T_C^2 = T_{m9}^2 = T_{A3}^2 .$$

The precedence constraints of tasks and messages are

$$\phi_{S3}^{1,2} + d_{S3}^{1,2} \le \phi_{m3}^{1,2},\ \phi_{S4}^{1,2,3} + d_{S4}^{1,2,3} \le \phi_{m4}^{1,2,3},$$
$$\max(\phi_{m3}^{1,2} + Ed_{m3}^{1,2}, \phi_{m4}^{1,2,3} + Ed_{m4}^{1,2,3}) \le \phi_C^2,\ \phi_{C2}^2 + d_{C2}^2 \le \phi_{m9}^2,$$
$$\phi_{m9}^2 + Ed_{m8}^2 \le \phi_{A3}^2 .$$

The constraints for the 3rd control loop as follows: The end-to-end time constraint is

$$\max(\phi_{A4}^3 + d_{A4}^3, \phi_{A5}^3 + d_{A5}^3) \le MADT^3 .$$

The period constraints of tasks and messages are

$$T_{S4}^{1,2,3} = T_{m4}^{1,2,3},\ T_{S5}^3 = T_{m5}^3,\ T_{S6}^3 = T_{m6}^3,\ T_{m4}^{1,2,3} \mid T_C^3,$$
$$T_{m5}^3 \mid T_C^3,\ T_{m6}^3 \mid T_C^3,\ T_C^3 = T_{m10}^3 = T_{A4}^3,\ T_C^3 = T_{m11}^3 = T_{A5}^3 .$$

The precedence constraints of tasks and messages are

$$\phi_{S4}^{1,2,3} + d_{S4}^{1,2,3} \le \phi_{m4}^{1,2,3},\ \phi_{S5}^3 + d_{S5}^3 \le \phi_{m5}^3,\ \phi_{S6}^3 + d_{S6}^3 \le \phi_{m6}^3,$$
$$\max(\phi_{m4}^{1,2,3} + Ed_{m4}^{1,2,3}, \phi_{m5}^3 + Ed_{m5}^3, \phi_{m6}^3 + Ed_{m6}^3) \le \phi_C^3,$$
$$\phi_{C3}^3 + d_{C3}^3 \le \phi_{m10}^3,\ \phi_{C3}^3 + d_{C3}^3 \le \phi_{m11}^3,\ \phi_{m10}^3 + Ed_{m10}^3 \le \phi_{A4}^3,$$
$$\phi_{m11}^3 + Ed_{m11}^3 \le \phi_{A5}^3 .$$

3.5 Scheduling results

To schedule the system by using the constraints derived in subsection 3.4, the priority and the period of task and message was assigned by using the assignment algorithms proposed in [8] to assign these. After assigning, the worst-case response times of tasks and messages were calculated by the analysis method described in subsection 2.2. The calculated response time of task and message were substituted for the deadlines of the derived constraints, and then the derived constraints were solved. If the results met the inequalities of end-to-end time constraints, the parameters of task and message such as priority, period, and initial phase time were saved as schedulable parameters.

The scheduling results are illustrated from Fig.6 to Fig.10. They represent the periods and the end-to-end worst-case response times of control loops in the cases of two kinds of noise models and no noise condition. The X-axis represents the number of the iterative calculations of the period assignment algorithm in [8], and the Y-axis represents the end-to-end response time and period of control loops. The period assignment algorithm tries to find out the smallest period of the control loop that also sustains the system to meet its real-time requirements. That is, the period of control loop has to be longer than the end-to-end worst-case response time. This algorithm repeats a searching algorithm based on a bisection algorithm until the smallest one is founded.

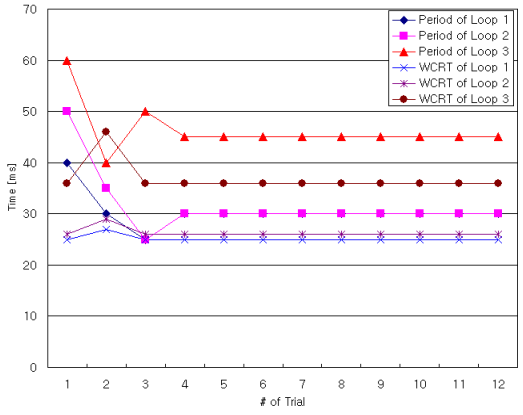

Fig.6 CAN: 100kbps under no noise condition

In Fig.6, the assigned period and the end-to-end worst-case response time of each control for no noise source and 100kbps CAN are represented. Fig.6 shows that the periods of the control loops are set to 30ms, 30ms, and 45ms, and the end-to-end response times are 25ms, 26ms, and 36ms. These scheduling results mean that the system can be scheduled by setting the periods of the control loops to 30ms, 30ms, and 45ms under these conditions.

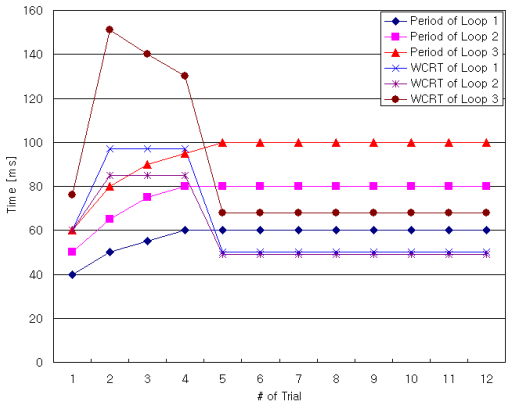

Fig.7 CAN: 100kbps under CASE1 noise condition

In Fig.7, the assigned period and the end-t-end worst-case response time of each control are represented when CASE1 noise condition of Table 2 was applied to the system, and the speed of CAN was 100kbps. The periods of control loops were 60ms, 80ms, and 100ms, and the end-to-end response times were 50ms, 50ms, and 70ms. Therefore, the system can be scheduled under these conditions.

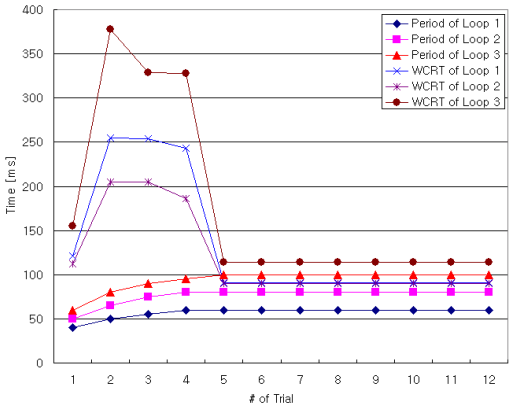

Fig.8 CAN: 100kbps under CASE2 noise condition

Fig.8 shows the scheduling results obtained when CASE2 noise condition of Table 2 having two noise sources was applied to the system, and the speed of CAN was 100kbps. Because of high degrees of noise, the transmission delay of each message increased, and this delay increased the end-to-end response time of the control loop.

Fig.6, Fig.7, and Fig.8 show that we can find a feasible set of periods for the no noise condition and the CASE1 noise condition but cannot find one for the CASE2 noise condition. In order to schedule the system for the CASE2 noise condition, we should change some parameters such as the network speed. For example, let us change the network speed to 500 kbps. When 500kbps CAN is used, the scheduling results for CASE1 and CASE2 noise conditions are shown in Fig.9 and Fig.10.

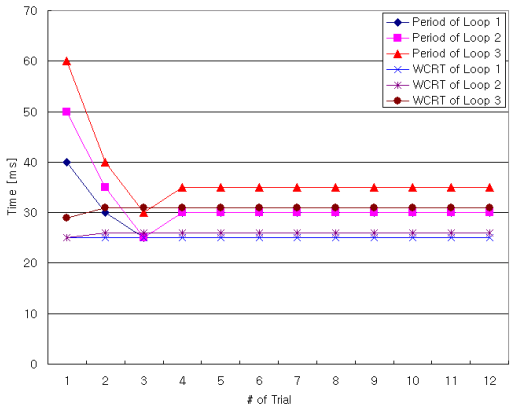

Fig. 9 CAN: 500kbps under CASE1 noise condition

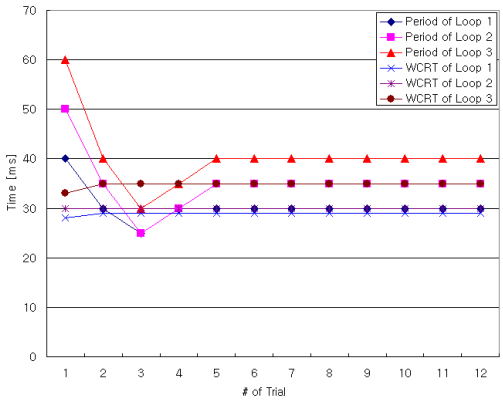

Fig.10 CAN: 500kbps under CASE2 noise condition

Fig.9 and Fig.10 show that the system can be scheduled for CASE2 noise condition as well as CASE1. Despite the CASE2 noise condition, the system can be scheduled if the periods are set to 30ms, 35ms and 40ms and the response times to 29ms, 30ms and 35ms. From these results, we know that the system requirements must be changed to guarantee the performance according to noise condition. Using the proposed method, we can evaluate and design the system to meet its requirements under noise conditions before run-time.

The worst-case response times of all messages are depicted according to the network speed and the noise models in Fig.11. At 100kbps, the response times of the messages varied markedly according to the three noise conditions. This variation means that it takes much time to recover the errors because 100kbps is a relatively low speed. At 500kbps, the response times varied less than that at 100kbps. Also, the response times

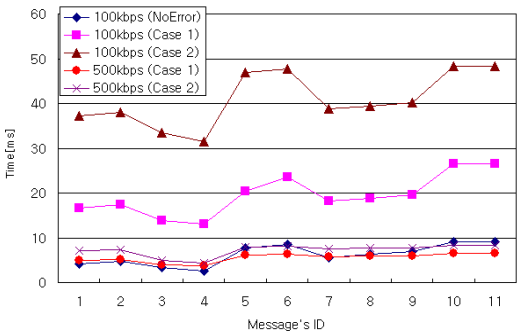

Fig.11 Worst-case response times of all messages

of 500kbs under CASE2 noise condition is similar to those of 100kbs under no noise condition. Even though the delay caused by error may be relatively small to the response time of each message, the transmission errors have a heavy effect on the system because the end-to-end response time of a control loop is the sum of all response times of the messages. A higher transmission speed is one good solution to solving the scheduling problem caused by the transmission errors.

When CAN network speed is 500kbps and noise condition is that of CASE2, the detail scheduling results are shown in Table 3. The parameters of message and task such as the priority, the period, the initial phase time, and the deadline were calculated using the proposed method. By setting the static parameters of the system to these, the system can guarantee its performance, under the noise environment at run-time.

Table 3 CAN: 500kbps under CASE2 condition

Loop 1 (MADT: 60)								Loop 2 (MADT: 80)								Loop 3 (MADT: 100)							
Task	Priority		e	T	R	d	ϕ	Task	Priority		e	T	R	d	ϕ	Task	Priority		e	T	R	d	ϕ
	T	M							T	M							T	M					
τ^1_{S1}	1		1	30	1	1	0	$\tau^{1,2}_{S3}$	1		1	5	1	1	0	$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0
τ^1_{m1}		3	0.1	30	7.2	8	1	$\tau^{1,2}_{m3}$		2	0.3	5	4.8	5	1	$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1
τ^1_{S2}	1		1	30	1	1	0	$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0	τ^3_{S5}	1		1	40	1	1	0
τ^1_{m2}		4	0.1	30	7.3	8	1	$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1	τ^3_{m5}		8	0.1	40	7.9	8	1
$\tau^{1,2}_{S3}$	1		1	5	1	1	0	τ^2_C	2		9	35	15	15	6	τ^3_{S6}	1		1	40	1	1	0
$\tau^{1,2}_{m3}$		2	0.3	5	4.8	5	1	τ^2_{O1}	1		4	20	4	4	0	τ^3_{m6}		9	0.1	40	8.1	9	1
$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0	τ^2_{O2}	3		6	50	19	19	0	τ^3_C	2		11	40	15	15	10
$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1	τ^2_{O3}	4		8	60	31	31	0	τ^3_{O1}	1		4	20	4	4	0
τ^1_C	2		7	30	11	11	9	τ^2_{m9}		7	0.1	35	7.8	8	21	τ^3_{O2}	3		6	50	25	25	0
τ^1_{O1}	1		4	20	4	4	0	τ^2_{A3}	1		1	35	1	1	29	τ^3_{O3}	4		8	60	33	33	0
τ^1_{O2}	3		6	50	17	17	0									τ^3_{m10}		10	0.1	40	8.2	9	25
τ^1_{O3}	4		8	60	29	29	0									τ^3_{A4}	1		1	40	1	1	34
τ^1_{m7}		5	0.1	30	7.5	8	20									τ^3_{m11}		11	0.1	40	8.2	9	25
τ^1_{A1}	1		1	30	1	1	28									τ^3_{A5}	1		1	40	1	1	34
τ^1_{m8}		6	0.1	30	7.6	8	20																
τ^1_{A2}	1		1	30	1	1	28																

4. CONCLUSIONS

Most distributed control systems operate in the plant generally having many devices such as high power motor, which generate noise. These motors generate high degrees of Electro Magnetic Interferences (EMIs). These interferences have much effect on the control system and its network, and cause transmission errors, which lead to performance degradation such as longer transmission delay or serious faults in the control systems. Therefore, the transmission error due to noise should be considered when trying to satisfy the given real-time constraints in the system design.

This paper proposed the scheduling method of both task and message for control systems under noise environment, and the proposed method was verified through an example. Two kinds of noise models were applied to CAN, and the end-to-end response time of control loop and the response time of message were analyzed. The results showed that the transmission delay caused by the transmission error is critical to system performance, and that the system requirements must be changed depending on the noise conditions. Using the proposed scheduling method, system designers can design a system guaranteeing its performance while the system operates under noise environment.

Future works should focus on the derivation method for the noise model of an active spot and the application method for the noise conditions according to the place within a system.

REFERENCES

[1] A. Burns, **Preemptive priority based scheduling: An appropriate engineering approach in Principles of Real-Time Systems**, Prentice Hall, 1994.

[2] J. Xu and D. Parnas, "Scheduling processes with release times, deadlines, precedence and exclusion relations," **IEEE Tr. on Software Engineering**, pp. 360-369, March, 1990.

[3] J. Y.-T Leung and J. Whitehead, "On Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks," **Performance Evaluation**, 2(4), pp. 237-250, December, 1982.

[4] K. Tindell, "Holistic Schedulability Analysis for Distributed Hard Real-time Systems," **Technical Report, YCS-197, Dept. of Computer Science Univ. of York**, NOV., 1994.

[5] R. Gerber and S.S. Hong, "Guaranteeing Real- Time Requirements with Resource-Based Calibration of Periodic Processes," **IEEE Tr. on Software Engineering**, 21(7), July, 1995.

[6] J.W. Park, Y.S. Kim, S.S. Hong, M. Saksena, S.H. Noh and W.H. Kwon, "Network conscious of Distributed Real-Time Systems," **Journal of System Architecture**, pp. 131-156, 1998.

[7] S. Faucou, A.-M. Deplanche and J.-P. Beauvais, "Heuristic Techniques for Allocating and Scheduling Communicating," **WFCS-2000**, pp. 257-265, 2000.

[8] H.Y. Kim, C.M. Lee and H.S. Park, "Optimal Period and Priority Assignment Using Task & Message-Based Scheduling in Distributed Control Systems," **Journal of Control, Automation and Systems Engineering**, Vol.8, No.6, JUN., 2002.

[9] Sasikumar Punnekkat, Hans Hansson, and Christer Norston, "Response Time Analysis under Errors for CAN," **Proc. of the 6th Real-Time Technology and**

Applications Symposium, pp.258-265, MAY, 2000

[10] Liang Cai and Wei Zhang, "EMI in hydropower plant and EMC design for its computer monitoring and control system," **the 3rd International Symposium on Electromagnetic Compatibility**, pp.378 -381, 2002

[11] K. Tindell and A. Burns, "Guaranteed Message Latencies for Distributed Safety Critical Hard Real-Time Networks," **YCS 229, Dept. of Computer Science, Univ. of York**, June, 1994.

[12] CAN in Automation (CiA), **CAN Specification 2.0 Part A and Part B**. http://www.can-cia.de

[13] Intel, **82527 Serial Communications Controller Architectural Overview**, Jan., 1996.

[14] K. Tindell, H. Hansson, and A. Wellings, "Analyzing Real-time Communications: Controller Area Network," **IEEE Real-time Systems Symposium**, 1994.