Journal of Korea Multimedia Society Vol. 20, No. 8, August 2017(pp. 1357-1368)
https://doi.org/10.9717/kmms.2017.20.8.1357

상태천이모델 기반의 센서 노드 설계

신동현†, 김창화††

Sensor Node Design based on State Transition Model

DongHyun Shin†, Changhwa Kim††

ABSTRACT

Sensor networks are used in various fields such as marine, defense, and smart home etc. Among the components of the sensor network, the sensor node collects sensor data, as one of the representative sensor network roles, and the sensor node makes a greate influence on the overall performance of the sensor network. Therefore, how to design the sensor node is an important issue in the sensor network field. However, the research on the sensor network architecture suitable for the sensor network installation environment has been made more important than the research on how to configure the sensor node. In this paper, we propose to identify elements to be considered for designing a sensor node that makes a large influence on the performance of the sensor network, and to easily implement the sensor node through the state transition model based on these elements.

Key words: Sensor Node Construction, Sensor Node Transition Model, Sensor Network

1. 서 론

센서네트워크는 해양, 국방, 스마트 홈, 스마트 팩토리 등의 다양한 분야에서 센서 데이터를 수집하고 응용하는데 있어 유용하게 활용되고 있다[1,2,3,4]. 특히 최근에 4차 산업혁명이 이슈로 떠오르면서 클라우드 컴퓨팅, 빅데이터 컴퓨팅, IoT (Internet of Things)가 함께 이슈가 되고 있는데, 센서네트워크는 IoT의 기반이 되는 대표적인 기술이라 할 수 있다. 이러한 센서네트워크는 크게 센서, 센서 노드, 게이트웨이 등으로 구성되는데 센서네트워크의 역할 중 하나인 센서 데이터를 수집하는데 있어 직접적으로 관여하는 요소는 센서 노드이다.

센서 노드의 주된 역할은 주위 환경의 데이터를 수집하여 처리 및 상위 노드로 전달하며, 센서 데이터의 수집 및 상위 노드로 전달을 위한 빈번한 통신 등으로 인해 센서네트워크의 에너지 효율, 통신 오버헤드 등의 성능에 막대한 영향을 미친다. 특히 한정된 자원을 갖는 센서네트워크는 에너지 효율이 중요하기 때문에 센서네트워크의 성능에 큰 영향을 미치는 센서 노드를 사용 환경에 적합하도록 구성하여 에너지 효율을 높여야 한다.

하지만 센서 노드는 센서네트워크 전체 성능에 상당한 영향을 미침에도 센서 노드를 어떻게 설계할 것인가에 대한 연구보다 단순히 해양, 산악, 국방 등의 인간이 쉽게 접근할 수 없는 환경에서의 센서 데이터를 수집하는데 적합한 센서네트워크의 아키텍처는 무엇인가에 대한 연구가 주를 이루었다[3,4,5,6,

※ Corresponding Author : Changhwa Kim, Address: (26403) 150, Namwon-ro, Heungop-myeon, Wonju-si, Gangwon-do, Korea, TEL : +82-33-760-8663, FAX : +82-33-760-8718, E-mail : kch@gwnu.ac.kr
Receipt date : Jul. 17, 2017, Approval date : Jul. 28, 2017
† Department of Computer Science & Engineering, Gangneung-Wonju National University
(E-mail : dhshin@cs.gwnu.ac.kr)
†† Department of Computer Science & Engineering, Gangneung-Wonju National University
※ This research was a part of the project titled "Development of the wide-band underwater mobile communication systems" funded by the Ministry of Oceans and Fisheries, Korea.
※ This study was also partly supported by Gangneung-Wonju National University.

7,8,9]. 최근에는 상태천이모델 기반으로 게이트웨이에서의 통신 오류 처리에 대한 연구[10]도 진행되었지만 게이트웨이는 특징과 역할이 센서 노드와 다르기 때문에 센서 노드 설계에 대한 내용을 직접 비교하기 어려울 뿐만 아니라 해당 논문에서는 센서 노드 설계에 대한 내용은 다루고 있지 않다.

따라서 본 논문에서는 센서 노드를 구성하는데 있어 센서네트워크 성능에 영향을 미칠 수 있는 대표적인 요소를 식별하고, 이 요소들을 기반으로 센서 노드 설계를 위한 방법을 제안한다. 센서 노드 설계를 위하여 상태천이모델을 이용하며, 상태천이모델을 이용하여 센서 노드를 설계하는 방법을 제안하는 이유는 다음과 같은 장점을 가지기 때문이다[10].

1. 센서 노드의 모델링을 수월하게 할 수 있다.
2. 상태천이모델 자체로 프로그램 설계가 되어 이벤트 중심의 코드로 쉽게 변환 가능하다.
3. 모델별 처리 방식, 차이점 등을 비교하기 쉽다.
4. 프로그램의 유지보수가 쉽다.

본 논문은 2절에서 센서 노드의 아키텍처 분류, 3절에서 센서 노드 설계을 위한 상태천이모델, 4절에서 성능평가, 5절에서 결론으로 마무리 짓는다.

2. 센서 노드 아키텍처의 분류

센서 노드를 어떻게 설계하는가에 따라 센서네트워크 전체 성능에 많은 영향을 미칠 수 있다. 따라서 이러한 요소들을 고려하여 센서 노드를 설계해야하며, 본 절에서는 센서네트워크 성능에 영향을 미치는 요소 중 대표 요소 5가지를 식별하여 이 요소들을 고려한 센서 노드 모델 설계 방법을 제시한다. 또한, 대표적인 3가지 센서 노드의 아키텍처를 함께 제시한다.

2.1 센서 노드 설계를 위한 요소 식별

센서 노드의 성능에 영향을 미치는 요소는 Table 1과 같이 크게 센서 데이터 전송방법, 통신 방법, 센서 데이터 통합 방법, 캘리브레이션, 센서 데이터 수집 신호, 총 5가지로 식별할 수 있다.

각 요소에 대한 자세한 설명은 다음과 같이 정리할 수 있다[1].

• Sensor Data Transmission Method : 센서에서 수집된 데이터를 센서 노드에서 상위 노드로 데이터 전송 시 상위 노드의 요청에 따라 전송하는 방법과 주기적으로 전송하는 방법으로 나눌 수 있다. 요청 방식의 경우 필요할 때에만 데이터를 요청하기 때문에 에너지가 더 적게 소모되지만 기능을 자주 사용할 경우 에너지 소모가 증가할 수 있다. 반대로 주기적인 방식의 경우 센싱을 위해 요청할 필요 없이 주기적으로 데이터 수집이 가능하지만 데이터가 필요하지 않은 구간에도 계속하여 센싱하기 때문에 에너지 소모가 증가할 수 있다.

• Communication Method : 센서 노드와 상위 노드간 통신 시 센서 노드의 설치 환경에 따라 무선, 유선, 유・무선 통신 중 적합한 통신을 사용할 수 있다. 유선 통신의 경우 무선 통신에 비해 통신 품질이 좋지만 설치비용이 상대적으로 더 높은 단점이 있고, 무선 통신의 경우 유선 통신에 비해 품질이 떨어지지만 설치비용이 유선 통신에 비해 상대적으로 낮으며 설치 환경에 큰 제약이 없다. 따라서 설치 환경에 적합하도록 통신 방법을 선택할 수 있으며, 유・무선 통신을 적절하게 혼합하여 사용 가능하다.

• Sensor Data Aggregation Method : 센서로부터 수집된 데이터를 센서 노드가 상위 노드로 전송할 때, 개별 센서 값 혹은 개별 센서 값을 통합하여 전송할 수 있다. 센서 데이터를 개별 센서 값으로 보내는 경우 실시간성이 보장될 수 있지만 메시지 생성이

Table 1. Elements for Sensor Node Design

Elements	Categories
Sensor Data Transmission Method	Request / Period
Communication Method	Wire / Wireless / Wire・Wireless
Sensor Data Aggregation Method	One / Complex
Calibration	Use / Unused
Sensor Data Collection Signal	Analog / Digital

비효율적이며 통합하여 보낼 경우 실시간성은 감소하지만 메시지 생성이 효율적이고 데이터 송·수신으로 인한 에너지 소모가 감소한다.

• Calibration : 센서 노드는 센서로부터 센서 신호를 수신하게 되는데 실제 센서 데이터 값을 알기

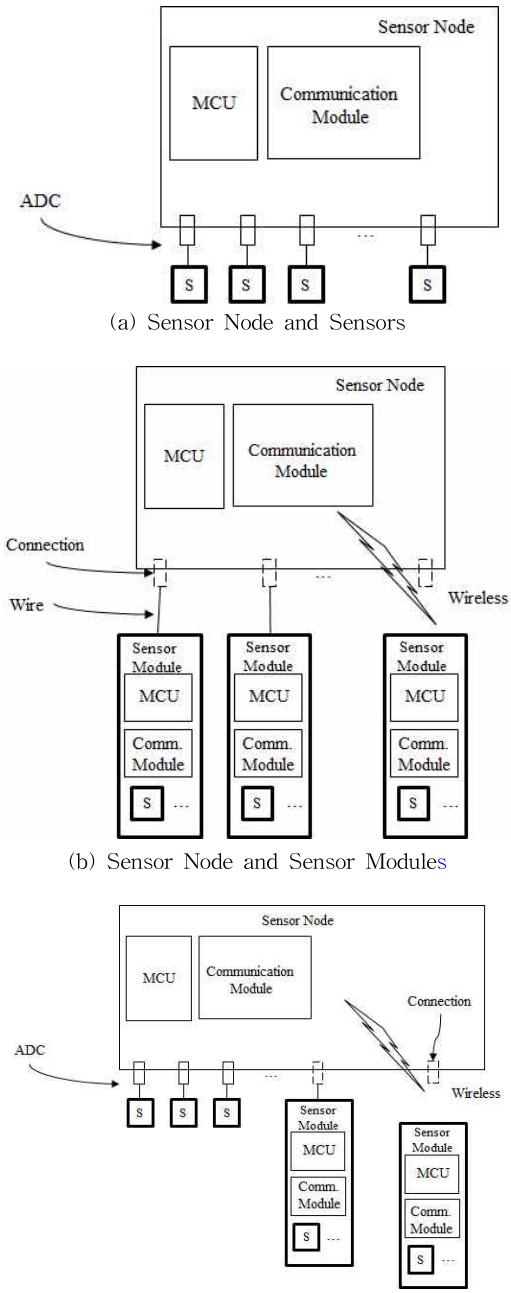


Fig. 1. Architectures of Sensor Node.

위해서는 캘리브레이션 작업이 필요하다. 캘리브레이션 작업을 센서 노드에서 수행할지, 상위 노드에서 수행할지 결정할 수 있다. 일반적으로 센서 노드의 에너지는 한정적이기 때문에 에너지 공급이 원활한 상위 노드에서 캘리브레이션을 수행한다.

• Sensor Data Collection Signal : 센서 노드는 센서 신호를 수신할 때 센서의 종류에 따라 아날로그 신호 혹은 디지털 신호를 수집할 수 있다.

센서 노드를 설계하기 위해 위와 같은 5가지 요소를 고려할 수 있는데, 가장 중요한 점은 가장 적합한 요소의 선택은 한 가지 이상의 경우의 수가 존재한다는 것이다. 이는 실제 센서 노드가 설치될 환경과 목적 등에 따라 선택해야 하는 요소가 다를 수 있기 때문이다. 따라서 센서 노드 설치 환경에 적합한 요소를 선택 하는 것 또한 중요하다.

2.2 센서 노드 아키텍처 분류

센서 노드는 일반적으로 MCU (Micro Controller Unit)와 통신 모듈 그리고 센서로 구성된다. 센서의 경우 제작사에 따라 처리 기능과 통신 모듈이 존재하는 센서 모듈과 수집한 센서 값을 아날로그 신호로만 전송하는 센서로 나눌 수 있으며, 일반적인 센서 노드의 아키텍처는 Fig. 1과 같이 크게 3가지 타입으로 구분할 수 있다. 첫 번째 타입 Fig. 1의 (a)는 처리와 통신 기능이 없는 센서를 가진 센서 노드이다. 이 타입은 센서 자체에 처리 기능이 없기 때문에 ADC (Analog to Digital)를 통하여 센서 노드로 센서 신호만을 전송하게 된다. 따라서 캘리브레이션 작업이 추가적으로 필요하다. 두 번째 타입 Fig. 1의 (b)는 처리 기능과 통신 모듈을 가진 센서 모듈을 가진 센서 노드이다. 센서 모듈은 첫 번째 타입의 센서와 다르게 센서 노드처럼 MCU와 통신 모듈을 포함하고 있어 센서 모듈에서 캘리브레이션 작업을 수행할 수 있다. 이때 센서 노드와 센서 모듈은 상황에 따라 유선 통신, 무선 통신, 유·무선 통신을 사용할 수 있다. 마지막 타입 Fig. 1의 (c)는 첫 번째 타입과 두 번째 타입의 센서와 센서 모듈을 모두 가지고 있는 센서 노드이다.

3. 센서 노드 설계를 위한 상태천이모델

본 절에서는 2절에서 식별한 센서 노드를 설계하

Table 2. State and its Explanation

No.	States	Explanation
1	Start	• Program start state
2	PortCheck	• Port Check state for polling
3	Sleep	• Sleep state of MCU
4	InterruptCheck	• State for interrupt check
5	MessageProcessing	• State for other message processing
6	Data/MessageReception	• State for data or message reception
7	DataTransmission	• State for data transmission to upper node
8	MessageOptionJudgement	• State for immediate or delayed delivery selection
9	DataSave	• State for data-saving

기 위해 고려해야 하는 요소들을 기반으로 실제 센서 노드를 쉽게 모델링 할 수 있도록 센서 노드 구성을 위한 상태천이모델의 구성요소를 식별하고, 상태천이모델 3가지를 제시한다. 본 논문에서는 상태천이모델의 구성요소를 통해 대표적으로 3가지 모델을 제안하였지만, 실제로 이 구성요소들의 조합을 통해 센서네트워크의 설치 환경에 적합한 다양한 모델들을 생성해낼 수 있다.

3.1 센서 노드 설계를 위한 상태천이모델의 구성요소

센서 노드 설계를 위한 상태천이모델의 구성요소는 크게 상태와 이벤트로 분류할 수 있다. 본 절에서는 센서 노드 설계를 위해 실제 사용되는 상태천이 모델의 상태와 이벤트들에 대한 정의와 설명을 다룬다.

상태천이모델을 구성하는 상태의 종류는 Start, PortCheck, Sleep, InterruptCheck, MessageProcessing, Data/MessageReception, DataTransmis-

Table 3. Events and its Explanation

No.	Events	Explanation
1	MessageExistence()	• This event checks whether any message exists in port check state. If true it moves data/message reception state, otherwise, it remains still in port check state
2	MessageReception Completion()	• This event checks whether the message reception is finished completely in data/message reception state. If true the current state moves to data/message source judgement state, otherwise, the state remains still in data/message reception state
3	MessageSource Check()	• This event checks what the type of the received message source is. If message source is the sensor type in data/message source judgement, the current state moves to data/message option judgement state, but if the source type is ETC, it moves to message processing state
4	MessageOption Check()	• If the returned option value as the result of this event is IM in data/message option judgement state, the current state moves to data transmission state, but if it is SV, the current state moves to data save state • IM means immediate transmission, SV means save and transmission
5	FinalData()	• If the saved data is final in data save state, the current state moves to data transmission state, otherwise, it moves to port check or sleep state
6	InterruptCheck()	• If the return value of interrupt check() is reception, the current state moves to data/message receipt state, but if ETC, it moves to message processing state
7	WFI	• If data transmission is completed in data transmission state, the current state moves to WFI (Wait for Interrupt) state
8	Processing Completion	• If data transmission is completed in message processing state, the current state moves to sleep state
9	Interrupt	• If interrupt occurs in sleep state, the current state moves to interrupt state

sion, MessageOptionJudgement, DataSave로 총 9가지이며, 이에 대한 설명은 Table 2와 같다.

각 상태가 다음 상태로 이동하기 위해서는 이벤트 발생이 필요하다. 여기에 필요한 이벤트는 Message Existence(), MessageReceptionCompletion(), MessageSourceCheck(), MessageOptionCheck(), Final Data(), InterruptCheck(), WFI, Processing Completion, Interrupt로 총 9가지가 있으며, 자세한 설명은 Table 3과 같다.

3.2 폴링 기반의 상태천이모델

상태천이모델 중 가장 첫 번째 모델은 폴링 기반의 상태천이모델이다. 이 모델은 PortCheck 상태에서 신호가 감지되면 데이터를 수신하고, 메시지 옵션 상태에 따라 즉시 상위 노드로 전송할지, 저장 후 전송할지 결정하게 된다. 처음 시작하면 포트를 확인하는 상태(PortCheck)가 되며, 메시지가 수신된 경우(Message Existence()==TRUE) 메시지를 수신하는 상태(Data/MessageReception)로 이동하게 된다. 이때 메시지 수신이 완료되면(MessageReception Completion()==TRUE) 메시지가 센서로부터 도착한 센서 메시지인지, 상위 노드에서 도착한 메시지인지 확인하는 상태(Data/MessageSourceJudgement)로 이동하게 되며, 센서로부터 도착한 메시지인 경우(MessageSource Check()==Sensor) 센서 데이터를 즉시 상위 노드로 전송할 것인지, 통합하여 전송할 것인지 판단하는 상태(Data/MessageOptionJudgement)로 이동하게 되고, 옵션 값이 즉시 전송인 경우(MessageOption Check()==IM) 데이터를 전송하는 상태(DataTransmission)로 이동 후 데이터 전송이 완료되면(Data TransmissionCompletion) 처음 상태로 돌아가며, 옵션 값이 추후 전송인 경우(MessageOptionCheck()==SV) 데이터를 저장하는 상태(Data Save)로 이동 후 처음 상태로 이동하게 된다. 이때 마지막 데이터(FinalData()==TRUE)인 경우 데이터를 전송하는 상태로 이동 후 데이터 전송이 완료되면 처음 상태로 이동한다. 만약 상위 노드로부터 도착한 메시지인 경우(MessageSourceCheck()-==ETC) 해당 메시지를 처리하는 상태(Message Processing)에서 메시지 처리 후 처음 상태로 돌아가게 된다. 이에 대한 상태천이모델과 알고리즘은 Fig. 2, 3과 같다.

3.3 인터럽트 기반의 상태천이 모델

인터럽트 기반의 상태천이 모델은 폴링 기반의 상태천이 모델과 다르게 데이터 혹은 메시지를 대기하는 동안 인터럽트 방식을 사용하는 모델이다. 이 모델은 처음 시작하자마자 Sleep 상태에 있으며, 인터럽트가 발생할 경우에만 다음 절차를 수행할 수 있다. 인터럽트 기반의 상태천이 모델은 폴링 기반의 상태천이 모델과 비슷하지만 처음에 포트를 확인하지 않고 바로 절전 모드(Sleep State)로 들어가게 된다. 이때 인터럽트가 발생한 경우 인터럽트가 데이터 수신에 대한 인터럽트이면(InterruptCheck()== Receipt) 데이터를 수신하는 상태(Data/Message Reception)로 이동하고, 기타 인터럽트이면(InterruptCheck()==ETC) 해당 인터럽트를 처리하는 상태(MessageProcessing)상태로 이동하게 된다. 이것을 제외한 내용은 폴링 기반의 상태천이모델과 동일하다. 이에 대한 상태천이도와 알고리즘은 Fig. 4, 5와 같다.

3.4 폴링과 인터럽트 기반의 상태천이모델

이 모델은 폴링과 인터럽트 방식을 모두 사용하는 기법이다. 프로그램 처음 수행 시 절전 모드를 수행하며, 일정 주기마다 폴링 방법으로 데이터를 받아오고 그 이외에는 인터럽트 방식으로 데이터를 받아온다. 이때 센서와 센서 노드 간에는 폴링 방법으로, 센서 모듈과 센서 노드 간에는 인터럽트 방법으로 데이터를 받아올 수 있다. 이 방법은 폴링 기반의 상태천이모델과 인터럽트 기반의 상태천이모델을 혼합하여 사용한 것으로 해당 설명은 3.2절과 3.3절을 참고하면 된다. 이에 대한 상태천이모델과 알고리즘은 Fig. 6, 7과 같다.

4. 성능평가 및 분석

본 절에서는 센서 모델의 타입 중 센서 노드와 센서 모듈, 센서가 복합적으로 연결되어 있는 타입을 선정하고, 상태천이모델 3가지에 대한 성능평가를 실시한다. 성능평가는 총 3가지로 센서 노드에서 소모되는 에너지와 전력을 측정한다.

4.1 평가환경 및 평가 절차

상태천이모델에 대한 성능평가를 위해 Table 4와 같이 센서 노드, 센서, 소프트웨어 툴, 멀티미터 등을

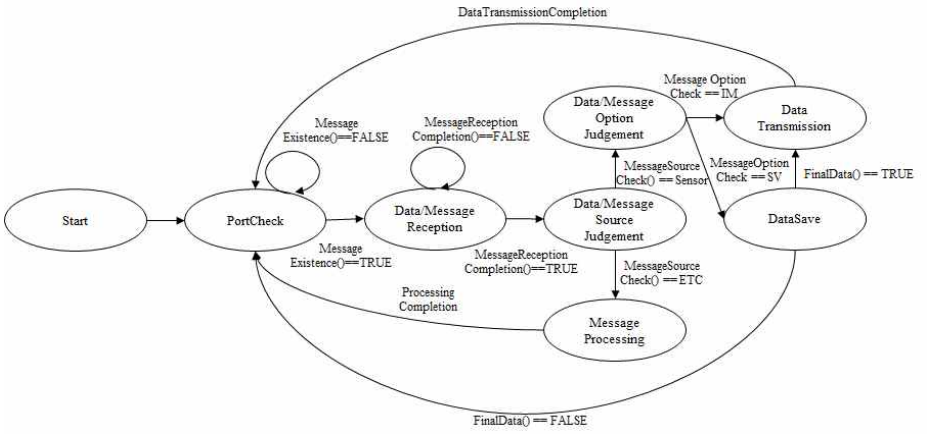


Fig. 2. State Transition Model based on Polling.

```
STATE = Port Check;
WHILE(1){
   SWITCH(STATE){
      CASE PortCheck:
         IF(MessageExistence() == TRUE) THEN
            STATE = Data/MessageReception;
         BREAK;
      CASE Data/MessageReception:
         IF(MessageReceptionCompletion() == TRUE) THEN
            STATE = Data/MessageSourceJudgement;
         BREAK;
      CASE Data/MessageSourceJudgement:
         IF(MessageSourceCheck() == Sensor) THEN
            STATE = Data/MessageOptionJudgement;
         ELSE
            STATE = MessageProcessing;
         BREAK;
      CASE Data/MessageOptionJudgement:
         IF(Message OptionCheck() == IM) THEN
            STATE = DataTransmission;
         ELSE
            STATE = DataSave;
         BREAK;
      CASE Message Processing:
         Message Processing();
         STATE = PortCheck;
         BREAK;
      CASE Data Transmission:
         Data Transmission();
         State = PortCheck;
         BREAK;
      CASE Data Save;
         Data Save();
         IF(FinalData() == TRUE) THEN
            STATE = DataTransmission;
         ELSE
            STATE = PortCheck;
         BREAK;
   }
```

Fig. 3. Algorithm of State Transition Model for Polling

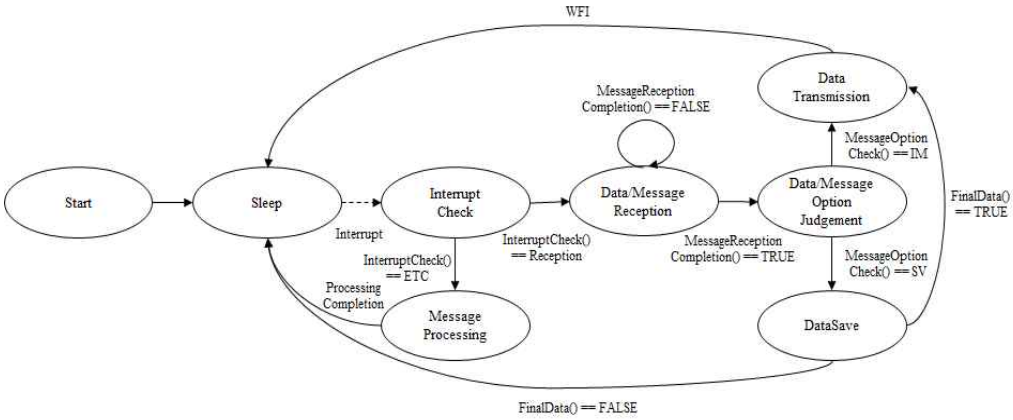


Fig. 4. State Transition Model for Interrupt.

```
STATE = Sleep;
WHILE(1){
   SWITCH(STATE){
      CASE Sleep:
         WFI();
         STATE = InterruptCheck;
         BREAK;
      CASE InterruptCheck:
         IF(InterruptCheck() == Reception) THEN
            STATE = Data/MessageReception;
         ELSE
            STATE = MessageProcessing;
         BREAK;
      CASE MessageProcessing:
         MessageProcessing();
         STATE = Sleep;
         BREAK;
      CASE Data/MessageReception:
         IF(MessageReceiptCompletion() == TRUE) THEN
            STATE = Data/MessageOptionJudgement;
         ELSE
         BREAK;
      CASE Data/MessageOptionJudgement:
         IF(MessageOptionCheck() == IM) THEN
            STATE = DataTransmission;
         ELSE
            STATE = DataSave;
         BREAK;
      CASE DataTransmission:
         Data Transmission();
         STATE = Sleep;
         BREAK;
      CASE DataSave:
         IF(FinalData() == TRUE) THEN
            STATE = DataTransmission;
         ELSE
            STATE = Sleep;
         BREAK;
   }
}
```

Fig. 5. Algorithm of State Transition Model for Interrupt.

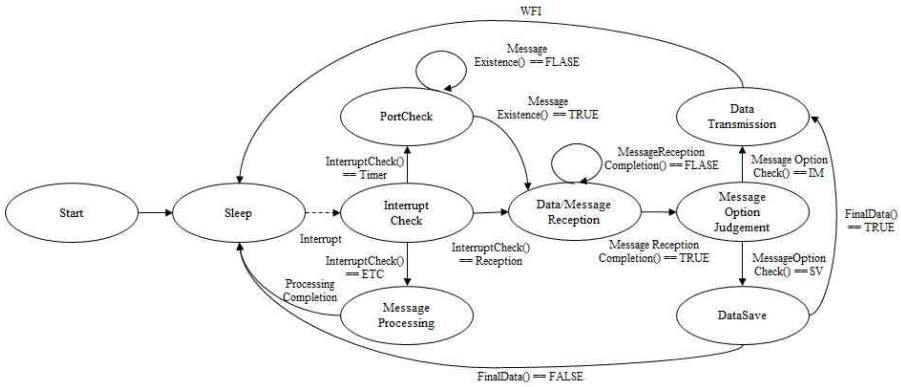


Fig. 6. State Transition Model for Polling and Interrupt.

```
STATE = Sleep;
WHILE(1){
   SWITCH(STATE){
      CASE Sleep:
         WFI();
         STATE = InterruptCheck;
         BREAK;
      CASE Interrupt Check:
         IF(InterruptCheck() == Reception) THEN
            STATE = Data/MessageReception;
         ELSE IF(InterruptCheck() == Timer) THEN
            STATE = PortCheck;
         ELSE
            STATE = MessageProcessing;
         BREAK;
      CASE Port Check:
         IF(MessageExistence() == TRUE) THEN
            STATE = Data/MessageReception;
         BREAK;
      CASE MessageProcessing:
         MessageProcessing();
         STATE = Sleep;
         BREAK;
      CASE Data/MessageReception:
         IF(Message ReceiptCompletion() == TRUE) THEN
            STATE = Data/MessageOptionJudgement;
         ELSE
         BREAK;
      CASE Data/MessageOptionJudgement:
         IF(MessageOptionCheck() == IM) THEN
            STATE = DataTransmission;
         ELSE
            STATE = DataSave;
         BREAK;
      CASE Data Transmission:
         Data Transmission();
         STATE = Sleep;
         BREAK;
      CASE Data Save:
         IF(FinalData() == TRUE) THEN
            STATE = DataTransmission;
         ELSE
            STATE = Sleep;
         BREAK;
   }
}
```

Fig. 7. Algorithm of State Transition Model for Polling and Interrupt.

Table 4. Using Tools and Specification

No.	Tools	Spec.
1	Software Tool	• IAR Embedded Workbench
2	Sensor Node	• Cortex - M3, 512KB Flash Memory, 64KB RAM
3	Communication Modules	• Zigbee Module • RS-485 Module
4	Sensor	• Dummy sensor(the number of sensors: 4)
5	Multimeter	• HPM - 100A

사용하였다. 센서 노드에서 상위 노드로 데이터를 전달하기 위해 지그비 통신을 사용하였고, 센서와 센서 모듈간에는 RS-485 통신을 사용하였다.

성능평가는 다음의 3가지 조건에 적합한 센서 노드 프로그램을 실제 개발하여 Table 4에서 나열한 각종 툴을 이용해 성능평가를 실시하였다.

1. 센서 데이터 수신 주기에 따른 센서 노드의 에너지 소모 측정
2. 센서 개수에 따른 센서 노드의 에너지 소모 측정
3. 센서 데이터 통합 유/무에 따른 센서 노드의 에너지 소모 측정

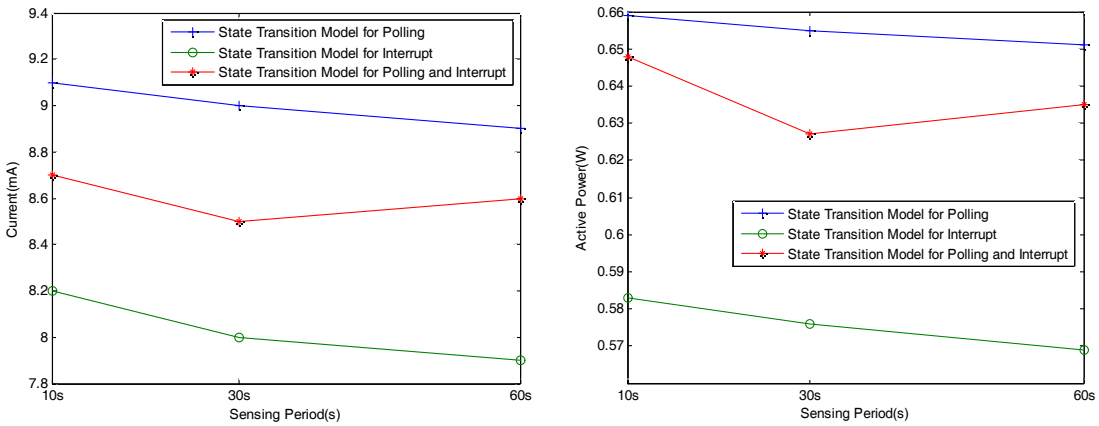
센서 데이터 수신 주기에 따른 센서 노드의 에너지 소모는 각각 10, 30, 60초에 따라 측정을 하였고 센서 개수에 따른 센서 노드의 에너지 소모는 센서의 개수를 1, 2, 4개로 증가시키며 측정을 하였다. 마지막으로 센서 데이터 통합 유/무에 따른 센서 노드의 에너지 소모는 10초 간격으로 센서 데이터를 즉시 전송하는 경우와 센서 데이터 5개를 모아 한번에 전송하는 경우로 구분하여 측정하였다.

성능평가를 위한 성능평가 절차는 먼저 각 상태천이모델에 해당하는 프로그램이 입혀진 센서 노드 등을 준비하고 멀티미터를 이용하여 1시간 동안의 소모 전류와 전력을 측정하였다.

4.2 평가결과 및 분석

4.3.1 센서 데이터 수신 주기에 따른 에너지 소모

센서 데이터 수신 주기를 각각 10초, 30초, 60초로 설정하여 소모된 센서 노드의 에너지를 측정한 결과 Fig. 8과 같은 결과를 얻었다. 기본적으로 폴링 방법의 경우 계속해서 노드가 활동하고 있기 때문에 전류와 전력 소모가 가장 높았으며, 인터럽트 방식의 경우 센서 데이터를 수집하고, 상위 노드로 전송할 때를 제외하면 슬립 모드를 수행하기 때문에 전류와 전력 소모가 가장 적었다. 폴링과 인터럽트 방식을 같이 사용한 경우 폴링 방법 보다는 적은, 인터럽트



(a) Current (mA) (b) Active Power (W)

Fig. 8. Comparison of Each Model Energy Consumptions According to Sensor Data Reception Period.

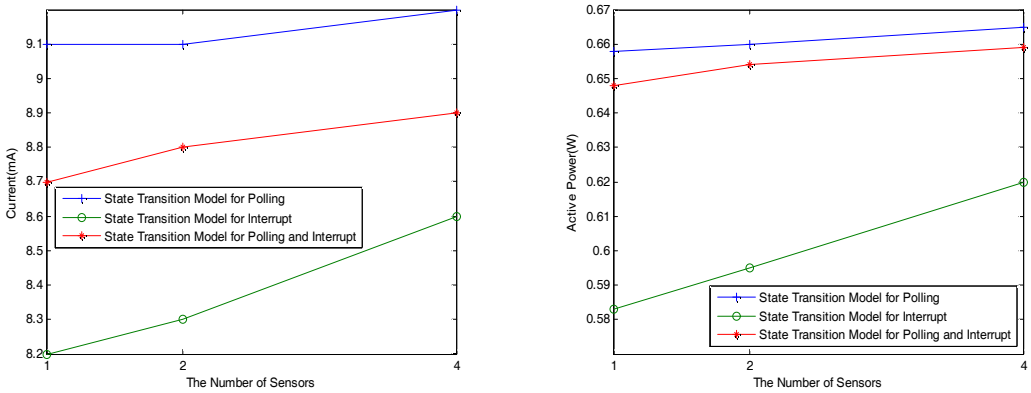
방법 보다는 많은 전류와 전력 소모가 발생하였다. 또한, 센서 데이터 수신 주기가 길면 길수록 3가지 방법 모두 전류 및 전력 소모가 대체적으로 줄어드는 것을 확인할 수 있다. 이는 센서 데이터를 수집하고, 상위 노드로 전달하는데 있어 가장 많은 에너지를 소모하기 때문이다.

4.3.2 센서 개수에 따른 에너지 소모

센서 개수를 각각 1, 2, 4개로 증가시켜 소모된 센서 노드의 에너지를 측정한 결과 Fig. 9와 같은 결과를 얻었다. 폴링, 인터럽트, 폴링 및 인터럽트를 혼합하여 사용한 방법 모두 센서의 개수가 커지면 커질수록 더 많은 전류와 전력을 소모하였다. 센서 개수에 따른 에너지 소모도 마찬가지로 센서의 개수가 증가할수록 센서 데이터 수집을 위한 통신과 상위 노드로 전송을 위한 통신 횟수가 증가하기 때문이다.

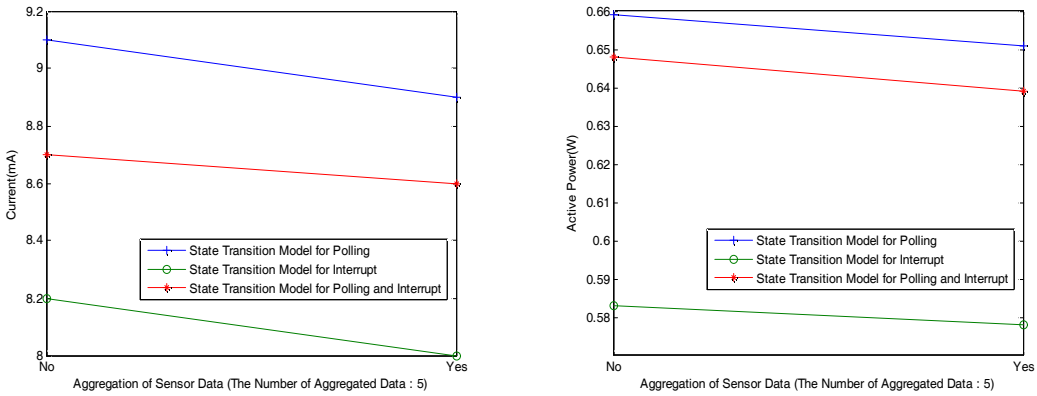
4.3.3 센서 데이터 통합 방법에 따른 에너지 소모

센서 데이터 통합 방법에 따른 에너지 소모의 경우 센서 데이터 통합을 하지 않은 경우 10초 간격으로 계속해서 센서 데이터 수집 후 상위 노드로 전송하며, 센서 데이터 통합을 한 경우 10초 간격으로 센서 데이터를 5회 수집 후 상위 노드로 전송한다. 이 경우 Fig. 10과 같이 센서 데이터를 통합하여 전송한 경우가 통합하지 않고 전송한 경우보다 전류와 전력 소모가 더 적은 것으로 나타났다. 이는 센서 데이터를 상위 노드로 전송할 때 발생되는 에너지 소모가



(a) Current (mA) (b) Active Power (W)

Fig. 9. Comparison of Each Model Energy Consumptions According to The Number of Sensor.



(a) Current (mA) (b) Active Power (W)

Fig. 10. Comparison of Each Model Energy Consumptions According to Sensor Data Aggregation.

5회에 1번씩 발생하기 때문이다.

5. 결론 및 향후 연구

센서 노드는 센서네트워크의 주된 역할 중 하나인 센서 데이터를 수집하는데 있어 직접적인 역할을 하며, 센서네트워크 전체 성능에 영향을 크게 미치는 요소 중 하나이다. 따라서 센서 노드의 설계를 어떻게 하는가에 따라 센서네트워크 에너지 소모, 통신 오버헤드 등이 높아질 수도, 낮아질 수도 있다. 하지만 기존까지의 연구는 대부분이 센서네트워크 설치 환경에 따라 적합하게 센서 데이터를 어떻게 수집할 것인가에 대한 연구에 초점이 맞춰져 있었다. 따라서 본 논문에서는 센서네트워크에서의 센서 노드 설계를 위한 요소를 식별하고 식별한 요소를 바탕으로 센서 노드의 3가지 상태천이모델을 제안하였으며, 이와 관련된 연구는 기존에 연구되지 않은 최초로 시도한 연구이다.

상태천이모델은 프로그램의 설계 및 코드로의 변환을 쉽게 해줄 뿐만 아니라 상태의 변화를 쉽게 파악할 수 있게 해주며, 상태천이모델 생성 시 자동으로 프로그램 코드로 변환해줄 수 있는 장점을 가진다. 실제 상태천이모델에 대한 센서 데이터 수집 주기, 센서 개수, 센서 데이터 통합 유/무에 따른 성능평가를 실시한 결과 인터럽트 방식에서의 성능이 가장 좋게 평가되었다. 하지만 본 논문에서는 센서 노드 설계를 위한 요소와 상태천이모델 구성 요소들을 조합하여 대표적인 3가지 모델을 제안한 것이며, 센서 노드 설계 시 선택해야 할 요소들은 센서네트워크의 설치 환경과 목적 등에 따라 달라질 수 있으므로 센서네트워크의 설치 환경에 적합한 요소들을 선택하는 것 또한 중요하다.

향후 연구로는 센서 노드 설계 시 여러 설치 환경에 적합할 수 있도록 하는 요소의 선택 방법과 상태천이모델을 만들면 센서 노드 프로그램의 기본 틀을 자동으로 생성해주는 것에 대한 연구와 기존에 개발된 여러 종류의 센서 노드들에 대한 기능 및 동작, 특성 등에 대해 본 논문에서 제시한 기능적 관점에서의 문제점을 분석하고 해결하는 연구를 진행하고자 한다.

REFERENCE

[1] D.H. Shin and C.H. Kim, "A Sensor Construction Method in Sensor Node of Sensor Network," *Proceeding of Korea Multimedia Society Conference*, Vol. 20, No. 1, pp. 13-16, 2017.

[2] D.H. Shin and C.H. Kim, "Data Compression Method for Reducing Sensor Data Loss and Error in Wireless Sensor Network," *Journal of Korea Multimedia Society*, Vol. 19, No. 2, pp. 360-374, 2016.

[3] H.W. Nam, S.S. Shin, C.H. Kim, and S.H. Park, "Remote Monitoring System Based on Ocean Sensor Networks for Offshore Aquaculture," *Proceeding of IEEE Oceanic Engineering Society*, pp. 14-19, 2014.

[4] R. Herlien, T. O'Reilly, K. Headley, D.R. Edgington, S. Tilak, T. Fountain, et al., "An Ocean Observatory Sensor Network Application," *Sensors*, pp. 1837-1842, 2010.

[5] D.H. Shin and C.H. Kim, "Sensor Network System for Littoral Sea Cage Culture Monitoring," *KIPS Transaction on Computer and Communication Systems*, Vol. 5, No. 9, pp. 247-260, 2016.

[6] D.S. Kridi, C.G.N. Carvalho, and D.G. Gomes, "Application of Wireless Sensor Networks for Beehive Monitoring and In-Hive Thermal Patterns Detection," *Computers and Electronics in Agriculture*, Vol. 127, pp. 221-235, 2016.

[7] J.Z. Sun and J. Zhou, "Querying Sensor Network with Extended SQL," *Wireless Communications, Networking and Information Security*, pp. 634-638, 2010.

[8] M. Ruiz, E. Alvarez, A. Serrano, and E. Garicia, "The Convergence between Wireless Sensor Networks and the Internet of Thnigs; Challenges and Perspectives: A Survey," *IEEE Latin America Transactions*, Vol. 14, No. 10, pp. 4249-4254, 2016.

[9] N.Y. Yun, J.I. Namgung, H.M. Park, S.H. Park, and C.H. Kim, "The Underwater Environment Monitoring System Based on Ocean

Oriented WSN(Wireless Sensor Network)," *Journal of Korea Multimedia Society*, Vol. 13, No.1, pp. 122-132, 2010.

[10] C.J. Seong and C.H. Kim, "State Transition Model-based Design of Wireless Gateway Types to Connect between a Sub-network of Things and Mobile Internet and their Performance Evaluations," *Journal of Korea Society for Simulation*, Vol. 25, No. 3, pp. 1-14, 2016.

신 동 현

2014년 강릉원주대학교 컴퓨터공학 공학사
2014년~2016년 강릉원주대학교 컴퓨터공학 공학석사
2016년 현재 강릉원주대학교 컴퓨터공학 박사과정

관심분야: Underwater Communication, IoT/IoUT, WSN (Wireless Sensor Network), Data Mining

김 창 화

1985년 고려대학교 수학교육과 이학사
1987년 고려대학교 전산학전공 이학석사
1990년 고려대학교 전산학전공 이학박사
1994년~1995년 University of Toronto, Enterprise Integration Lab. Post-Doc. & Visiting Professor
2002년~2004년 미국 Texas A&M대학 Visiting Scholar
1989년~현재 강릉원주대학교 컴퓨터공학과 교수
2005년~현재 강릉원주대학교 해양센서네트워크시스템 기술연구센터 센터장

관심분야: Underwater Communication and Sensor Network, IoT/IoUT, Distributed System, Intelligent System