한국시뮬레이션학회 논문지
Vol. 29, No. 3, pp. 73-82 (2020. 9)
http://doi.org/10.9709/JKSS.2020.29.3.073
ISSN 1225-5904

무선 센서 네트워크에서 다중 라우팅 프로토콜 사용을 위한 모델링과 시뮬레이션

남수만 · 조대호 · 김형종 †

Modeling and Simulation for using Multiple Routing Protocols in Wireless Sensor Networks

Su Man Nam · Tae Ho Cho · Hyung Jong Kim†

ABSTRACT

In the fourth industrial revolution, wireless sensor networks (WSNs) are an important element of collecting and analyzing data in a variety of environments without human intervention. This sensor network is greatly affected by topology and routing protocols. Routing protocols, which affect energy consumption, are executed after deploying sensor nodes. Once built, they are difficult to change. Before the WSN is deployed, a routing protocol is carefully selected in view of various environments and the performance of the protocol is evaluated. In this paper, we propose a model to simulate multiple routing protocols using a discrete event system specification (DEVS). The DEVS-based proposed model simulates various situations without changes and structures of the its model as algorithms of the routing protocols are implemented in its coordinators model. To verify normal behaviors of the proposed model, the number of report delivery and the energy consumption of the sensor network were compared using representative protocols LEACH and Dijkstra. As a result, it was confirmed that the proposed model executes normally in both routing protocols.

Key words : Wireless Sensor Network, Discrete Event System Specification, Multiple Routing Protocols, Routing Configuration

요 약

4차 산업혁명에 있어 무선 센서 네트워크(Wireless Sensor Networks; WSNs)는 사람의 개입 없이 다양한 환경에서 데이터를 수집하고 분석하는 중요한 요소이다. 센서 네트워크는 토폴로지와 라우팅 프로토콜에 따라 네트워크 수명에 크게 영향을 받는데, 그중 라우팅 프로토콜은 일단 한 번 구축되면 운영 중에 변경하기 위해 많은 자원(에너지 등)이 소모된다. 네트워크 구축 전 동작과 성능을 예측하기 위해 다양한 시뮬레이터들이 제안되었음에도 불구하고, 라우팅 프로토콜들에 초점을 맞춰 시뮬레이션할 수 있는 도구는 부족한 현실이다. 본 논문은 DEVS(discrete event system specification)를 사용하여 다수의 라우팅 프로토콜을 시뮬레이션할 수 있는 WSN 모델을 제안한다. DEVS 기반의 제안 모델은 코디네이터 모델에 라우팅 프로토콜의 알고리즘을 구현하게 되면 모델의 구조와 변경 없이 다양한 상황에 대하여 시뮬레이션할 수 있다. 제안 모델의 정상적인 동작 확인을 위해 WSN의 대표적인 프로토콜인 LEACH와 Dijkstra를 사용하여 보고서 전달 수와 네트워크의 에너지 소모를 비교하였다. 그 결과 제안 모델은 두 라우팅 프로토콜에서 정상적으로 동작함을 확인하였다.

주요어 : 무선 센서 네트워크, 이산 사건 시스템 명세서, 다중 라우팅 프로토콜, 라우팅 설정

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2020-2018-0-01799) / 이 연구는 2017년도 정부(교육부)의 재원으로 한국연구 재단의 지원을 받아 수행된 연구(NRF-2017R1D1A1B0 3034644)임.

Received: 3 February 2020, **Revised**: 2 September 2020, **Accepted**: 2 September 2020

† **Corresponding Author**: Hyung Joung Kim
E-mail: hkim@swu.ac.kr
Seoul Women's University Information Security

1. 서론

최근 4차 산업혁명 시대의 도래로 다양한 환경에서 데이터 수집과 분석은 중요한 화두이다. 데이터 수집과 분석을 위해 사용되는 무선 센서 네트워크(Wireless Sensor Networks; 이하 WSNs)는 자동화된 원격 정보 수집을 목적으로 다양한 응용(예: 과학적, 의학적, 군사적 등) 개발

에서 폭넓게 활용된다[1,2]. WSN에서는 센서 노드들이 이벤트를 감지(sensing), 계산(computing), 그리고 무선 통신(wireless communication)하고, 베이스스테이션(base station 이하 BS)는 센서들로부터 이벤트 보고서(report)를 수집하여 사용자에게 정보를 제공한다. 하지만, 수많은 센서가 포함된 네트워크는 토폴로지와 라우팅 프로토콜에 따라 크게 영향을 받는다. 특히, WSN의 라우팅 프로토콜은 센서 노드들이 배포된 후 구축되면 많은 에너지 소모 등으로 변경하기가 어렵다. 이를 위해 모델링과 시뮬레이션(modeling and simulation)[3] 등의 기술을 활용하여 네트워크 구축 전에 동작과 성능 예측이 중요하다[4-6].

WSN의 동작과 성능을 예측하기 위해 그동안 다양한 시뮬레이터들이 제안되었다. 그 중 센서 네트워크 시뮬레이터를 위해 널리 사용되는 TOSSIM[5], Avrora[6] 등이 있다. 다만, 이들은 응용 프로그램들의 동작에 대해서만 평가하여 실행 시간과 전력 소모를 예측할 수 없거나 하나의 쓰레드로 동작되기 때문에 많은 수의 센서 노드들을 시뮬레이션하기가 어렵다.

[4]에서는 WSN 시뮬레이터 구축을 위해 센서 노드에 설치된 임베디드 SW 정확한 동작 확인, 노드의 제한된 자원에 따른 실행 시간 및 에너지 소모 예측, 많은 수의 센서들의 시뮬레이션이 요구된다. 이 요구들을 만족시키기 위해 이산 사건 시뮬레이션 방법을 주로 사용한다[4,7]. 이산 사건 시뮬레이션을 위해 사용하는 DEVS(Discrete Event System Specification, 이하 DEVS)는 계층적으로 형식적(formalism)인 모델에서 사건의 발생에 따라 상태를 변화시켜 대상 시스템을 시뮬레이션한다. 특히, DEVS에서는 커널 모델(broadcast model, controlled model 등)을 사용하여 한 모델에서 여러 개의 동형 모델로 생성할 수 있다. 따라서 한두 개의 센서 유형으로 여러 개 동형 모델을 생성해야 하는 WSN 시뮬레이터를 구현하기 위해 DEVS 커널 모델을 사용하는 것이 적합하다[7].

기존에 발표된 DEVS 기반의 WSN 시뮬레이터들[7,8]과 범용 시뮬레이터들[5,9-11]은 시뮬레이터에서 지원하는 라우팅 프로토콜들만 사용할 수 있기 때문에 다양한 프로토콜[12,13]을 자유롭게 구현하여 시뮬레이션하기가 어렵다. 그 결과 제안된 시뮬레이터들이 WSN의 여러 성능을 측정하기 위해 활용됨에도 불구하고 여러 라우팅 프로토콜을 선택하여 시뮬레이션하기에는 한계가 있다.

본 연구에서는 DEVS 커널 모델인 컨트롤드 모델(controlled-model)을 사용하여 한 코디네이터 모델에서 다수의 WSN 라우팅 프로토콜을 적용할 수 있는 시뮬레이션 모델을 제안한다. 제안하는 DEVS 기반의 WSN 모델은 코디네이터 모델을 사용함으로 다양한 라우팅 프로토콜 알고리즘을 자유롭게 구현하고 시뮬레이션 전에 한 프로토콜을 선택할 수 있다. 제안 모델은 코디네이터 모델 외에 동형 모델로 구현된 센서 모델들과 BS 모델을 사용한다. 이때 센서 모델들 사이에는 코디네이터가 위치하며 선택된 라우팅 프로토콜을 기반으로 다음 센서 모델을 결정하게 된다. 다시 말해, 한 소스(source) 센서가 보고서를 전달하면 코디네이터 모델은 네트워크의 라우팅 정보에 따라 다음 목적(destination) 노드를 선택한다. 따라서 제안 모델은 DEVS 모델의 구조와 변경 없이 구현된 다수의 프로토콜들 중 한 라우팅 알고리즘의 선택에 따라 시뮬레이션을 수행한다. 제안 모델의 정확한 동작 확인을 위해 코디네이터 모델에는 대표적인 라우팅 프로토콜들(LEACH[1], Dijkstra[7,14])을 구현했다. 그리고 두 라우팅 프로토콜 위에서 동작하는 센서 모델들이 전달하는 보고서 수와 에너지 소모를 비교하였다.

본 논문의 구성은 2장에서 기존 라우팅 프로토콜들을 간략히 소개하고, 3장에서는 이산 사건 시뮬레이션의 대표적인 DEVS를 설명하겠다. 4장에서는 제안 모델인 DEVS 기반의 WSN 모델을 소개하고 5장에서는 구현한 모델을 사용하여 실험한 결과들을 제시하고 분석하겠다. 마지막으로 6장에서는 전반적인 결론을 요약한다.

2. 관련연구

본 장은 WSN 시뮬레이터들을 살펴보고 다음으로 WSN의 대표적인 라우팅 프로토콜들을 소개한다.

2.1 WSN 시뮬레이터

NS-3는 NS-2의 불필요한 커플링, 복잡한 구조 및 낮은 유연성의 단점을 보완했다[9]. NS-3은 개별 이벤트 기반 시뮬레이션에서 작동하는 오픈 소스 네트워크 시뮬레이터이다. NS-3는 Zigbee 등 여러 통신 모듈을 지원하고, 여러 네트워크 트래픽 모니터 도구들(FlowMonitor[14], Wireshark[15] 등)을 사용할 수 있다. 이 시뮬레이터는 다양한 네트워크 시뮬레이션에서 널리 사용되고 있지만 센서 네트워크의 프로토콜들(예: 라우팅 프로토콜, 보안 프로토콜 등)을 사용하기에는 한계가 있다.

OMNET++[10]은 주로 네트워크 시뮬레이터 구축에 사용되는 모듈식 개별 이벤트 네트워크 시뮬레이션 프레임워크이다. OMNeT++은 사전 정의된 연결을 사용하여 모듈 간의 통신을 의무화함으로써 NS-3에 비해 복잡성이 낮다. 이 프레임워크는 무선 네트워크 시뮬레이션 구축을

위한 확장 가능한 모듈식 및 컴포넌트 기반 C++ 시뮬레이션 라이브러리 프레임워크로 간주된다. OMNeT++는 NS-3과 같은 대형 센서 노드로 시뮬레이션을 수행하기에 어렵다.

TOSSIM[5]은 NEST(Network Embedded Software Technology) 프로젝트에서 개발되었다. TOSSIM은 센서 노드에서 TinyOS 및 애플리케이션의 동작을 확인하고 무선 네트워크의 알고리즘을 평가한다. 그렇지만 TOSSIM은 특정 소프트웨어에만 의존하고 있어 확정성이 어렵다.

CupCarbon[11]은 다중 에이전트 및 개별 이벤트 시뮬레이션을 기반으로 하는 스마트 시티 및 사물 인터넷 무선 센서 네트워크(SCI-WSN) 시뮬레이터이다. 이 시뮬레이터를 사용하면 OpenStreetMap의 디지털 지리적 인터페이스에서 센서 네트워크를 모델링하고 시뮬레이션 할 수 있다. 시뮬레이터의 목표는 모니터링, 환경 데이터 수집 등을위한 분산 알고리즘을 설계, 시각화, 디버그 및 검증하는 것이다. CupCarbon은 지리적 위치에 따라 다양한 상황을 시뮬레이션하고 모니터링 할 수 있지만 센서 네트워크의 다양한 라우팅 프로토콜[12,13]을 사용하기는 어렵다.

상기 시뮬레이터들은 특성에 따라 다양한 분야에서 사용되고 있지만 센서 노드의 각 구성 요소에 대한 암시적 시뮬레이션, 모듈식 측면을 사용하여 모델링하기는 어렵다[8]. 또한 다양한 환경 조건에서 센서의 동작을 관찰하기도 어렵다. 이러한 문제를 해결하기 위해 DEVS[8]를 사용한 센서 네트워크 시뮬레이션 대한 모델링 연구가 시작되었다. 그러나 이러한 연구는 특정 상황에 대한 센서 네트워크의 시뮬레이션만 구현했다. 예를 들어 [17]에서 센서 모델들은 Cell-DEVS[17,18]을 기반으로 구현되었는데 셀의 위치만 사용하여 모델링 및 시뮬레이션이 가능하지 특정 라우팅 프로토콜 적용은 어렵다. Cell-DEVS는 복잡한 셀룰러 모델의 효율적인 실행을 제공함에도 불구하고 센서 필드를 셀 단위로 나누어야하므로 센서 네트워크의 모든 라우팅 프로토콜을 구현하는 데 한계가 있습니다. 따라서 DEVS 기반 WSN 시뮬레이터에서 다양한 라우팅 프로토콜을 구현하고 선택하여 효과적인 모델링 및 시뮬레이션을 할 수 있어야 한다.

2.2 WSN 라우팅 프로토콜

WSN의 라우팅 프로토콜은 센서 노드들이 감지한 정보를 BS까지 효율적으로 전달하는 경로 설정 기술이다. 이 라우팅 프로토콜은 LEACH(Low-energy adaptive clustering hierarchy)[1], Dijkstra[7] 등과 같이 네트워크의 확장성과 저전력 운영에 많은 기법이 제안되고 있다[12,13,19]. WSN의 라우팅 프로토콜은 크게 계층적 위치 기반 라우팅으로 나뉜다[13,20]. 본 장은 계층적 라우팅 프로토콜의 대표적인 LEACH와 클러스터 기반의 Dijkstra[7,14]에 대해서 설명한다.

2.2.1 LEACH

LEACH[1]는 계층적 라우팅 프로토콜의 대표 적으로써 센서 필드에 뿌려진 다수의 센서 노드들을 몇 개의 클러스터로 나뉜다. 각각의 클러스터는 다수의 멤버 노드(Member Node; 이하 MN)와 한 개의 클러스터 헤드(Cluster Head; 이하 CH)로 구성된다. Fig. 1과 같이 MN들은 이벤트가 발생하면 이벤트 데이터(이벤트의 위치, 시간, 유형 정보)를 생성하여 CH에 전달한다. CH는 자신의 MN들로부터 수신 데이터를 수집하고 수집된 데이터를 기반으로 보고서를 만든다. 그다음, CH는 생성된 보고서를 BS에 직접 전달한다.

LEACH에서는 에너지 소모에 따른 CH의 수명은 전체 센서 네트워크 수명과 연관되어 있다. 상대적으로 BS로부터 멀리 떨어진 CH는 보고서를 전달을 위해 많은 에너지를 소모해야 한다. 이를 위해 LEACH에서는 라운드 단위(셋업 단계, 안정 상태 단계)로 운영하여 CH를 선출하고 이를 MN들에 공지한다.

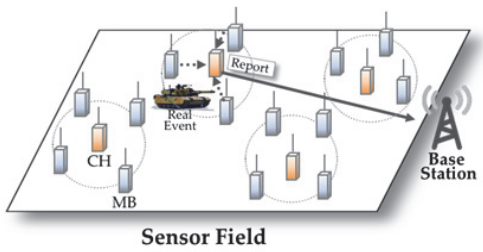

Fig. 1. WSN for LEACH

2.2.2 클러스터 기반 Dijkstra

클러스터 기반 Dijkstra[7,14]는 네트워크를 클러스터 단위로 분할하여 가장 짧은 경로를 따라 CH들을 통해 다중 홉 방식으로 보고서를 전달한다. 모든 센서 노드가 센서 필드에 배치된 후 각 클러스터에서 한 노드는 CH로 선택된다. 모든 클러스터는 고유한 클러스터 ID가 있다. 한 클러스터는 한 홉 내에서 한 개의 CH와 L개의 멤버 노드를 구성한다. 각각의 CH는 Dijkstra 알고리즘을 기반으로 BS까지 가장 짧은 경로를 발견한다. 경로를 발견된 후 CH들은 소스 CH와 BS 사이에서 업스트림(upstream)

방식으로 다음 전달 CH의 ID(identification)를 얻고, 자신의 홉 카운트(hop count)를 계산한다. Fig. 2와 같이 이벤트가 발생하면 한 소스 노드는 보고서를 생성한 후 중계 CH들을 통해 이 보고서를 BS에 전달한다.

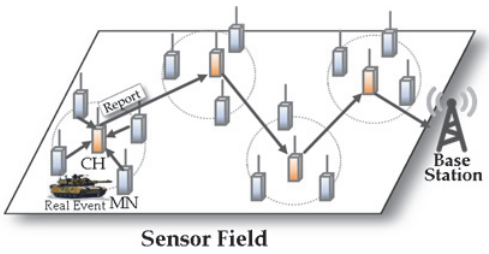

Fig. 2. WSN for Dijkstra based on Clusters

3. DEVS

Zeigler를 통해 개발된 DEVS 형식론[3,22-24]은 일반적인 시스템들을 분석하기 위해 계층적이고 모듈화된 개별 이벤트 모델을 표현하는 이론적이고 잘 정의된 수단이다. DEVS는 모델의 재사용성(model reusability)을 증대시키고 운용 의미론(operational semantics)을 내포하고 있어 실세계 시스템의 실행을 용이하게 묘사할 수 있는 특징을 갖는다. DEVS 형식론에서는 원자 모델과 결합 모델로 정의한다. 따라서 모델링이 완성되면 시스템은 원자와 결합 모델로 구성된 하나의 큰 결합 모델로 표현된다. 그리고 이 시스템은 레벨별로 계층성 있는 모델 구조를 갖게 된다[22].

원자 모델은 한 시스템의 입력과 출력, 상태들, 시간, 그리고 함수들을 갖는다. 시스템의 함수들은 현재 상태와 입력에 따라 다음 상태와 출력을 결정한다. 원자 모델의 형식은 다음 구조와 같다.

$$M = \langle X, S, Y, \delta_{int}, \delta_{ext}, \lambda, t_a \rangle$$

위 형식에서 X는 이산사건 입력 집합, S은 일련의 이산사건 상태의 집합, Y는 이산사건 출력 집합, δ_{int}은 내부 상태 천이 함수, δ_{ext}은 외부 상태 천이 함수, λ는 출력 천이 함수, t_a은 시간진행함수이다.

결합 모델의 구조는 다음과 같다.

$$CM = \langle X, Y, \{M_i\}, EIC, EOC, IC, select \rangle$$

위 형식에서 X는 이산사건 입력 집합, Y는 이산사건 출력 집합, $\{M_i\}$는 모든 이산사건 컴포넌트 모델들의 집합, EIC는 외부 입력 연결 관계, EOC는 외부 출력 연결 관계, IC는 내부 연결 관계, $select$는 같은 시간에 존재하는 사건을 발생하는 모델들에 대한 선택 함수이다.

이러한 결합 모델은 더 큰 결합된 모델과 합쳐질 수 있거나 더 복잡한 모델을 만들기 위해 몇몇 원자 모델들과 결합할 수 있다[22].

WSN 을 DEVS 모델로 구축하기 위해서는 DEVS 커널 모델(kernel model) 사용이 효과적이다[7]. 이 커널 모델은 결합 모델로써 한 자식 모델을 통해 여러 개의 동형 모델(isomorphic models)을 생성한다. DEVS 커널 모델은 브로드캐스트 모델(Broadcast-models), 하이퍼큐브 모델(Hypercube-models), 셀룰러 모델(Cellular-models), 컨트롤드 모델(Controlled-models)이 있다.

그 중 컨트롤드 모델 다른 커널 모델의 서브 모델들과 달리 중앙의 한 코디네이터(coordinator)에서 여러 모델을 제어하는 방식이다. 이 코디네이터는 컨트롤드 모델에서 동작하는 원자 모델인 컨트롤러(controller)로써, 컨트롤드 모델의 자식 중 어떤 모델들에게 출력할지 결정한다.

DEVS로 센서 네트워크의 동일한 센서들을 구현하기 위해서는 커널 모델을 사용하여 한 모델을 동형 모델로 생성해야 한다. 커널 모델로 생성된 여러 모델은 라우팅 프로토콜을 사용하여 연결 관계를 정의하는 것이 필요하다. 300개의 모델이 센서 노드로 구현되었다면 단일 경로로 설정하여도 300×2(입력, 출력)개의 연결이 필요하다. 이를 브로드캐스트 모델로 구현된다면 그 연결 관계는 기하급수로 늘어난다. 브로드캐스트 모델에서는 한 자식 모델의 출력이 다른 자식 모델들의 입력이 되기 때문이다. 브로드캐스트 모델로 구현된 DEVS 시뮬레이션에서 실행할 경우 CPU, 메모리 등과 같은 컴퓨팅 자원이 소모된다. 따라서, 수많은 센서를 DEVS 커널로 표현하기 위해서는 내부 커플링이 가변적으로 사용하는 컨트롤드 모델이 적합하다.

DEVS를 소프트웨어로 구현하기 위해서는 ADEVS[25], CD++[18] 등이 개발되었는데 본 논문에서는 자체 개발한 DEVS[26]를 사용했다.

4. DEVS 기반의 WSN 모델

본 장에서는 DEVS 기반의 WSN 모델에 대해서 개략적으로 설명하고, 이 모델의 행위를 표현하는 원자 모델들을 상세하게 살펴본다.

4.1 개요

제안 모델은 컨트롤드 모델을 사용한 DEVS 기반의 WSN 모델이다. Fig. 3은 제안하는 모델(EF-WSN)의 전체 모습이다. EF-WSN은 결합 모델로써 WSN과 EF 모델로 구성된다. WSN 모델은 센서 네트워크의 행위를 표현하는 원자 모델이 아래와 같이 4개가 있다.

- BS(Base Station): 센서 노드들(CH, MN)로부터 데이터 수집 및 분석
- ROUTING_CNTR(Routing Controller): MN와 CH로부터 받은 데이터와 보고서를 라우팅 알고리즘에 따라 다음 노드에 전달
- MN(Member Node): 이벤트 탐지 및 이벤트 데이터 전달
- CH(Cluster Head): MN로부터 전달받은 데이터 수집하고 BS를 향해 보고서 전달

이들의 연결 관계는 컨트롤드 모델의 컨트롤러인 ROUTING_CNTR에서 제어한다. ROUTING_CNTR은 네트워크의 라우팅 정보를 가지고 있으며 두 모델의 관계를 서로 연결한다. 예를 들어, 한 MN이 이벤트를 감지할 경우 MN은 그의 출력 포트를 통해 ROUTING_CNTR에 이벤트 데이터를 보낸다. 데이터를 전달받은 ROUTING_CNTR은 네트워크 정보를 통해 MN의 CH에 메시지를 보낸다. CH는 이 데이터를 받은 후에 보고서를 생성하고 다시 ROUTING_CNTR에 보고서를 보낸다.

EF-WSN 모델의 또 다른 하위 모델인 EF(experimental frame)는 GENR(generator)과 TRANSD(transducer)모델로 구성되어 있다. GENR는 센서 네트워크의 이벤트를 랜덤하게 생성하여 WSN 모델에 보낸다. TRANSD은 GENR에서 생성된 이벤트 메시지와 BS로부터 받은 보고서 메시지를 기록하고 저장한다.

4.2 EF-WSN의 원자 모델

EF-WSN에서는 제안 모델로써, 두 라우팅 프로토콜인 LEACH와 클러스터 기반 모델을 기반으로 WSN 모델로 구현한다. 이 모델에서는 여섯 개의 원자 모델이 있는데, 그 중에서 컨트롤러 모델(ROUTING_CNTR)은 각각의 라우팅 정보를 통해 센서 모델들의 관계를 연결한다. 따라서 ROUTING_CNTR은 정의된 라우팅 프로토콜들 중 하나를 선택함으로 구조 변경 없이 두 프로토콜을 시뮬레이션할 수 있다.

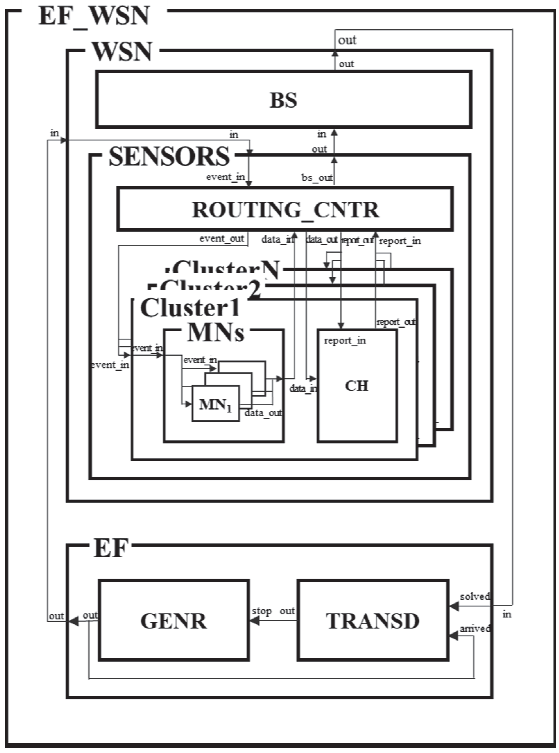

Fig. 3. Proposed Model

Fig. 4는 두 라우팅 프로토콜을 표현하는 ROUTING_CNTR의 상태 천이 모델이다. 이 모델에서는 크게 3번의 상태 천이가 일어난다. 첫 번째, GENR로부터 생성된 이벤트 정보는 event_in 포트를 통해 들어온다. 이를 컨트롤러에서 받고 상태를 passive에서 eventing으로 변환한다. 그리고 이벤트 정보의 위치 정보를 통해 event_out 포트로 한 MN에 전달한다(상태 천이 passive → eventing → passive). 두 번째, 컨트롤러는 MN 모델들로부터 받은 이벤트 데이터가 포함된 패킷을 받은 다음 상태를 passive에서 collecting으로 천이한다. 컨트롤러는 라우팅 정보에 따라 MN의 CH에 이 이벤트 데이터를 전달한다(상태 천이 passive → collecting → passive). 세 번째, 컨트롤러가 CH 모델로부터 받은 보고서를 받을 경우 passive에서 forwarding으로 천이한다. ROUTING_CNTR의 forwarding 상태에서는 라우팅 정보에 따라 받은 보고서를 어떤 모델에게 전달할지 결정한다. 이때, 라우팅 정보가 LEACH로 설정되었다면 보고서는 sensor_out 포트를 통해 BS로 전달된다(passive → forwarding → passive). 반면에, Dijkstra로 설정되었다면 보고서는 report_out 포트를 통해 다중 홉을 통과한다. 그리고 이 보고서는 다음 전달

모델이 BS일 때까지 전달된다.

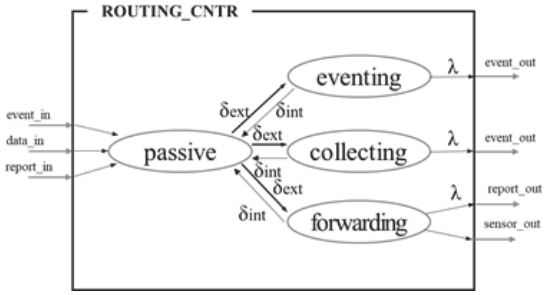

Fig. 4. State transition model of ROUTING_CNTR

컨트롤러 모델의 알고리즘은 주요 함수의 의사 코드는 Table 1, Table 2, Table 3과 같다. DEVS 특성상 모델의 입력을 받으면 표 1과 같이 외부 상태 천이가 먼저 실행되고, 그다음 출력 함수와 내부 상태 함수가 실행된다.

Fig. 5는 MN의 상태 천이 모델이다. MN 모델이 event_in 포트를 통해 컨트롤러부터 이벤트 정보를 받으면 passive 상태에서 sensing 상태로 변환한다. 이때 MN은 이벤트를 감지하고 forwarding에서 이벤트 데이터를 생성하고 data_out를 통해 컨트롤러에 이 데이터를 전달한다.

Fig 6은 CH 모델의 상태 천이를 보여준다. CH 모델에서는 두 개의 입력 포트를 갖는다. data_in는 MN로부터 오는 이벤트 데이터를, report_in는 CH로부터 오는 보

Table 1. Pseudo code of External transition function

```
if state = passive then
  switch( port )
    case event_in:
      location = getLocation(event);
      makeCoupling(location);
      holdIn(eventing, eventing_time);
      break;
    case data_in:
      ch_id = getNextRouting(event_data);
      makeCoupling(ch_id);
      holdIn(collecting, collecting_time);
      break;
    case report_in:
      ch_id = getNextRouting(report);
      makeCoupling(ch_id);
      holdIn(reporting, reporting_time);
      break;
  end swtich;
end if;
```

Table 2. Pseudo code of outuput function

```
switch(state)
  case eventing:
    forwardEvent(event_out, cluster_id);
    break;
  case collecting:
    forwardData(data_out, ch_id);
    break;
  case arriving:
    if( report.dst == 0 )
      forwardReport(bs_out, bs_id);
    else
      forwardReport(report_out, bs);
    end if;
    break;
end switch;
```

Table 3. Pseudo code of internal transition function

```
switch(state)
  case eventing:
  case collecting:
  case reporting:
    passivate();
    break;
end switch;
```

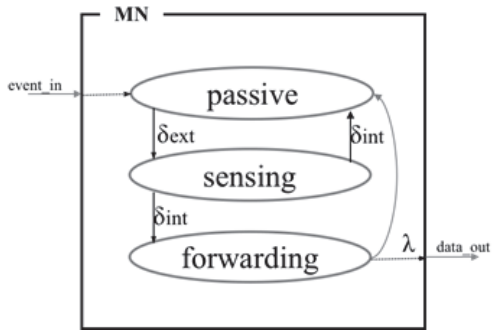

Fig. 5. State transition model of MN

고서를 받는다. 다만, LEACH 프로토콜에서는 CH가 보고서를 바로 BS에 전달함으로 report_in 포트 없이 실행된다. CH 모델이 입력 포토들을 통해 이벤트 데이터 또는 보고서를 받으면 sensing 상태로 변환되는데 이때 이벤트 데이터를 받으면 보고서를 생성하고, 보고서를 받으면 보고서를 분석한다. 그리고 forwarding에서 생성되었거나 받은 보고서는 report_out를 통해 컨트롤러에 전달된다.

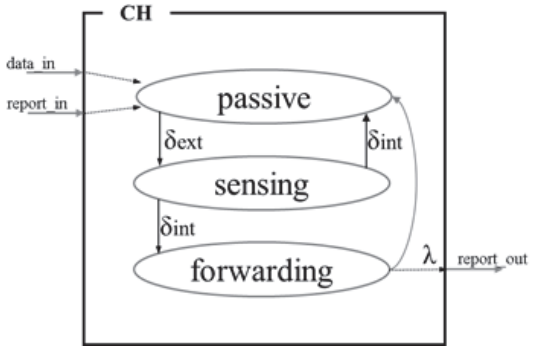

Fig. 6. CH의 상태 천이 모델

Fig. 7은 BS의 상태 천이 모델을 보여준다. BS는 CH 모델로부터 받은 보고서를 받은 후 상태를 analyzing로 천이한다. 그리고 분석 결과는 out 포트를 통해 보내진다.

EF 모델은 두 원자 모델 GENR와 TRANDS을 가지고 있다. GENR는 특정 시간마다 이벤트를 생성하여 WSN 모델에게 전달하고, TRANDS는 BS로부터 받은 보고서를 평가한다.

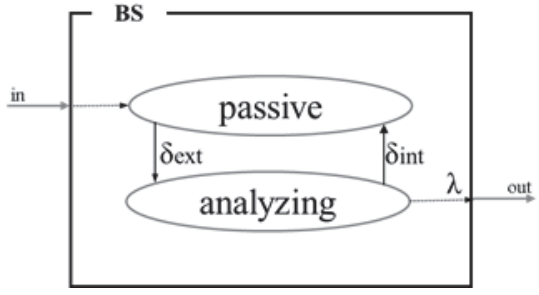

Fig. 7. State transition model of BS

5. 실험평가

우리의 제안 모델은 코디네이터 모델에서 라우팅 프로토콜의 알고리즘을 구현하게 되면 다양한 상황에 대하여 시뮬레이션할 수 있다. 특히, 제안 모델의 구조와 변경 없이 코디네이터 모델은 라우팅 알고리즘 구현만으로 WSN 프로토콜들을 평가할 수 있다. 본 실험에서는 WS의 평면과 계층적 라우팅 프로토콜의 대표인 LEACH와 Dijkstra을 선택하여 제안 모델의 정상적인 동작 확인과 성능을 평가하였다. 특히, Dijkstra 라우팅 프로토콜은 클러스터 기반으로 구현된다[7]. 클러스터 기반의 Dijkstra는 클러스터 단위로 구성된 WSN을 CH들을 중심으로 BS까지 최소 경로로 라우팅을 설정하는 방법이다. 한 센서 필드의 크기는 500x50㎡로써, 그 필드에 100개의 클러스터가 구성된다. 한 클러스터는 한 개의 CH와 네 개의 MN가 랜덤하게 배치된다. 이벤트가 센서 필드 위에서 랜덤하게 발생하게 되면 다수의 MN은 이벤트를 탐지하고 CH에 전달한다. CH는 자신의 탐지한 이벤트 데이터와 비교한 후 36바이트의 보고서를 생성하여 BS을 향해 그 보고서를 전달한다. 이때 LEACH는 BS까지 한 홉 내에 전달되고, Dijkstra는 다수의 CH를 경유하여 멀티 홉으로 전달된다. 센서 노드들이 보고서를 송수신할 때마다 Table 4에 보여준 것과 같이 노드의 에너지를 소모한다. 이때 패킷 손실이 없다고 가정한다. 센서 노드들은 배치가 완료된 후 시뮬레이션 종료 때까지 두 라우팅 프로토콜들을 한 번씩 실행한다. 이때 에너지 소모는 Table 4와 같이 각각 라우팅 프로토콜[1,7]에서 제시한 기준을 따랐다.

Table 4. Parameters

Parameter		LEACH	Cluster-based Dijkstra
Field Size		500x500㎡	
Number of Sensors		500	
Number of CHs		100	
Size	Event Data	8byte	
	Report	36byte	
Energy Con-sumption	Transmission	distance×0.06μJ	16.25μJ
	Reception	distance×0.05μJ	12.5μJ
	Report Generation	51.2μJ	15μJ
Simulation Time		200	
Event Generation		every 2	

Fig. 8은 전달된 보고서 수와 운행된 홉 수를 보여준다. 두 라우팅 프로토콜 모두 MN에서 이벤트를 감지한 후 이벤트 데이터를 그들의 CH에 보낸다. 이벤트 데이터를 받은 소스 CH는 자신의 라우팅 정보에 따라 다음 전달 노드에 보내게 된다. LEACH 라우팅 프로토콜에서는 소스 CH가 생성한 보고서를 BS에 바로 보내게 됨으로 두 홉 내에서 보고서를 전달이 집중되었다. 반면에 클러스터 기반의 Dijkstra은 설정된 가장 짧은 경로를 통해 다중 홉으로 보고서를 전달함으로 운행된 홉 수가 분산되었다.

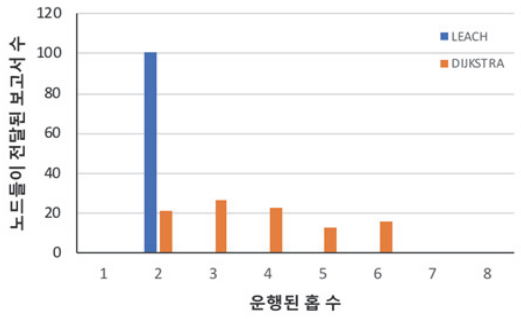

Fig. 8. 전달된 보고서 수와 운행된 홉 수

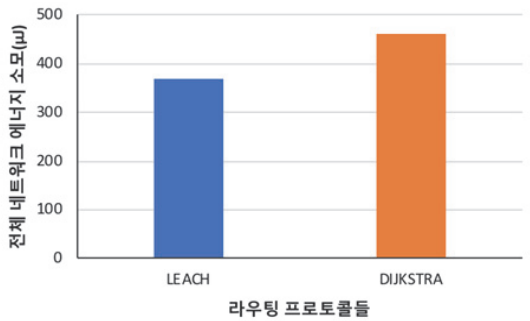

Fig. 9. 전체 네트워크 에너지 소모와 라우팅 프로토콜들

Fig. 9는 두 라우팅 프로토콜의 전체 네트워크 에너지 소모를 보여준다. Fig. 9에서 보는 것과 같이 클러스터 기반의 Dijkstra보다 LEACH가 에너지 소모가 적은 것을 볼 수 있다. 다만, LEACH에서는 BS로부터 먼 거리에 위치한 특정 CH들에서 많은 이벤트가 집중적으로 발생할 경우 그 CH들은 BS에 전달을 위해 원거리 전송해야만 다시 말해, 전체 센서 네트워크의 수명을 단축할 수 있는 원인이 될 수 있다.

6. 결론

WSN은 다수의 센서 노드와 BS로 구성되어 있는데 다양한 환경에서 데이터를 수집하고 분석한다. 센서 네트워크는 토폴로지와 라우팅 프로토콜에 따라 다양한 요소들(에너지 소모, 전달 수 등)이 영향을 받는다. WSN을 실제 구축하기 전에 시뮬레이터를 통해 목적에 맞는 시뮬레이션을 하는 것이 중요하다.

본 논문은 이산 사건 시뮬레이션인 DEVS를 사용하여 WSN에서 다중 라우팅 프로토콜(LEACH, 클러스터 기반 Dijkstra)을 위한 한 모델을 제안하고 시뮬레이션했다. DEVS 기반의 제안 모델은 두 프로토콜을 시뮬레이션 하기 위해 구조 변경 없이 라우팅 알고리즘만을 변경한다. 실험 결과를 통해 전체 네트워크의 에너지 소비는 클러스터 기반의 Dijkstra보다 LEACH가 낮았다. LEACH에서는 소스 CH에서 BS로 바로 전달되는 것을 확인했다. 향후, WSN의 다양한 라우팅 프로토콜에 맞는 모델을 개발하고 이들을 서로 비교할 것이다.

References

[1] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” IEEE Trans. Wirel. Commun., Vol. 1, pp. 660-670, 2002.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” Comput. Networks, vol. 38, pp. 393-422, 2002.

[3] B. P. Zeigler, “Hierarchical, modular discrete-event modelling in an object-oriented environment,” Simulation, vol. 49, pp. 219-230, 1987.

[4] 김방현, 김종현, “유비쿼터스 응용 개발을 위한 센서 네트워크 시뮬레이터”, 한국정보과학회 정보과학논문지, 13권 pp. 358-370, 2007.

[5] P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM: Accurate and scalable simulation of entire TinyOS applications,” SenSys'03 Proc. First Int. Conf. Embed. Networked Sens. Syst., pp. 126-137, 2003.

[6] B. L. Titzer, D. K. Lee, and J. Palsberg, “Avrora: Scalable sensor network simulation with precise timing,” 2005 4th Int. Symp. Inf. Process. Sens. Networks, IPSN 2005, vol. 2005, pp. 477-482, 2005.

[7] S. M. Nam and T. H. Cho, “Context-Aware Architecture for Probabilistic Voting-based Filtering Scheme in Sensor Networks,” IEEE Trans. Mob. Comput., vol. 16, no. 10, pp. 2751-2763, 2017.

[8] T. Antoine-Santoni, J. F. Santucci, E. De Gentili, and B. Costa, “DEVS-WSN: A discrete event approach for Wireless Sensor Network simulation,” AICCSA 08 - 6th IEEE/ACS Int. Conf. Comput. Syst. Appl., pp. 895-898, 2008.

[9] Nsnam, “NS-3.” https://www.nsnam.org/. [Accessed: 11-Feb-2020].

[10] OMNeT++, “OMNeT++.” https://omnetpp.org/.

[Accessed: 11-Feb-2020].

[11] K. Mehdi, M. Lounis, A. Bounceur, and T. Kechadi, “CupCarbon: A multi-agent and discrete event Wireless Sensor Network design and simulation tool,” SIMUTools 2014 - 7th Int. Conf. Simul. Tools Tech., pp. 126-131, 2014.

[12] G. Carneiro, P. Fortuna, and M. Ricardo, “FlowMonitor - a network monitoring framework for the Network Simulator 3 (NS-3),” 2012.

[13] Wireshark, https://www.wireshark.org/. [Accessed: 25-Feb-2020].

[14] T. Antoine-Santoni et al., “DEVS-WSN: A discrete event approach for Wireless Sensor Network simulation,” AICCSA 08 - 6th IEEE/ACS Int. Conf. Comput. Syst. Appl., pp. 3189-3200, 2009.

[15] G. Wainer, “CD++: A toolkit to develop DEVS models,” Softw. - Pract. Exp., 2002.

[16] W. Dijkstra, “A note on two problems in connexion with graphs,” Numer. Math., 1959.

[17] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, “Energy-efficient routing protocols in wireless sensor networks: A survey,” IEEE Commun. Surv. Tutorials, vol. 15, no. 2, pp. 551-591, 2013.

[18] K. Akkaya and M. Younis, “A survey on routing protocols for wireless sensor networks,” Ad Hoc Networks, vol. 3, no. 3, pp. 325-349, 2005.

[19] F. Li, A. Srinivasan, and J. Wu, “PVFS: A Probabilistic Voting-based Filtering Scheme in Wireless Sensor Networks,” Int. J. Secur. Networks, Vol. 3, pp. 173, 2008.

[20] B. P. Zeigler, Object-oriented simulation with hierarchical, modular models: intelligent agents and endomorphic systems. Academic press, 2014.

[21] B. P. Zeigler, T. H. Cho, and J. W. Rozenblit, “A knowledge-based simulation environment for hierarchical flexible manufacturing,” IEEE Trans. Syst. Man, Cybern. Part ASystems Humans., vol. 26, no. 1, pp. 81-90, 1996.

[22] T. G. Kim and B. P. Zeigler, “The DEVS Formalism: Hierarchical, Modular Systems Specification in an Object Oriented Framework,” in Winter Simulation Conference Proceedings, 1987, pp. 559-566.

[23] J. Nutaro, “ADEVS.” https://web.ornl.gov/~nutarojj/adevs/. [Accessed: 10-Feb-2020].

[24] S. M. Nam, “DEVS-based-WSN.” https://github.com/sumannam/DEVS-based-WSN.

남 수 만 (ORCID : https://orcid.org/0000-0002-8595-3307 / sumannam@gmail.com)

2009 한서대학교 컴퓨터정보학과 이학사
2013 성균관대학교 전기전자컴퓨터공학과 공학석사
2017 성균관대학교 전기전자컴퓨터공학과 공학박사
2017~ 2018 아주대학교 의료정보학과 연구강사
2018~ 현재 (주)두두아이티 M&S연구소 책임연구원

관심분야 : 유체역학, 시뮬레이션, 국방 M&S, 선박제어

조 대 호 (ORCID : https://orcid.org/0000-0001-6222-2096 / thcho@skku.edu)

1983 성균관대학교 전자공학과 공학사
1987 University of Alabama 전자공학과 공학석사
1993 University of Arizona 전자 및 컴퓨터공학과 공학박사
1995~ 현재 성균관대학교 정보통신공학부 교수

관심분야 : 무선 센서 네트워크, 모델링 시뮬레이션, 지능 시스템, 모델링 방법론, 네트워크 보안 시뮬레이션, 전사적 자원 관리

김 형 종 (ORCID : https://orcid.org/0000-0002-0608-5397 / hkim@swu.ac.kr)

1996 성균관대학교 정보공학과 공학사
1998 성균관대학교 정보공학과 공학석사
2001 성균관대학교 전기전자 및 컴퓨터공학과 공학박사
2001~ 2007 한국정보보호진흥원 수석연구원
2004~ 2006 미국 Carnegie Mellon University, CyLab 국제공동연구원
2013~ 2014 미국 Carnegie Mellon University, ECE, Visiting Professor
2007~ 현재 서울여자대학교 정보보호학과 정교수

관심분야 : 안드로이드 환경의 IoT서비스 개인정보보호, 블록체인 서비스 성능평가, 클라우드 서비스 보안 모델