석사학위 논문

6LoWPAN 네트워크의 게이트웨이에서의 토폴로지 관리

Topology Management of Gateway in 6LoWPAN Networks

2013년 6월

숭실대학교 대학원

정보통신공학과

박 영 기

석사학위 논문

6LoWPAN 네트워크의 게이트웨이에서의 토폴로지 관리

Topology Management of Gateway in 6LoWPAN Networks

2013년 6월

숭실대학교 대학원

정보통신공학과

박 영 기

석사학위 논문

6LoWPAN 네트워크의 게이트웨이에서의 토폴로지 관리

지도교수 김 영 한

이 논문을 석사학위 논문으로 제출함

2013년 6월

숭실대학교 대학원

정보통신공학과

박 영 기

박 영 기의 석사학위 논문을 인준함

심 사 위 원 장　　정　　윤　　원 인

심 사 위 원　　김　　영　　한 인

심 사 위 원　　노　　동　　건 인

2013년 6월

숭실대학교 대학원

목 차

표 목 차

그 림 목 차

국문초록

6LoWPAN 네트워크의 게이트웨이에서의 토폴로지 관리

박영기

정보통신공학과

숭실대학교 대학원

현재, IP기반의 무선센서네트워크는 환경조사, 빌딩관리, 산업화공장내 기기의 상태 정보 및 사용자 헬스케어등의 분야에 적용하기 위해 활발히 연구가 진행 되고 있다. 이러한 환경에 적용된 센서는 노드의 상태 정보 및 노드간의 네트워크 연결 정보를 위해 센서네트워크의 토폴로지 정보를 필수적으로 제공해야한다. 외부의 모니터링 시스템 또는 게이트웨이에서 센서노드의 정보를 수집하고 있다. 이러한 정보수집은 주기적으로 발생하게 되며, 센서네트워크의 자원을 지속적으로 사용하기 때문에 비효율적이다. 본 논문에서는 센서네트워크의 토폴로지 관리를 효율적으로 할 수 있게 하는 센서게이트웨이 구조를 설계하고 구현하였다. 구현된 시스템은 센서환경을 위해 설계된 CoAP프로토콜을 지원할 수 있도록 하였으며 CoAP을 통해 센서노드의 추가, 변경 등을 관측할 수 있는 기능을 추가하였다. 설계, 구현된 시스템의 정보전달 지연시간 등의 분석을 통해 성능을 고찰하였고 효율적으로 토폴로지관리가 가능함을 보였다.

ABSTRACT

Topology Management of Gateway in 6LoWPAN Networks

PARK, YOUNG KI

Department of Information and Telecommunication Engineering

Graduate School of Soongsil University

Currently, IP based Wireless Sensor Networks have been studied widely to apply in various fields such as environmental monitoring, building management, industrial plant equipment management (e.g. equipment's status information), and health care monitoring. For these applications, the network topology information and sensors' status are needed to be monitored in the Internet site. Therefore, a gateway between the Internet and sensor networks is required and plays an important role to collect sensors' information. The frequent information collection for sensors' status will result in inefficient network resource utilization. In this paper, we design and implement a sensor gateway which supports efficiently the sensor network topology management. The implemented system supports CoAP protocol and is designed with features for adding and changing

sensor nodes with CoAP protocol. Through the experiments and analyses, we demonstrate that our system achieves a good performance with a low request delay time and manage the sensor network topology efficiently.

제 1 장 서론

무선 센서네트워크 분야에서 외부의 모니터링시스템 연동과 센서 상태 정보 수집 기술이 활발히 연구 되고 있다. 연동을 위해서는 새로운 스택(stack)을 정의하는 대신에 센서와 외부네트워크간의 프로토콜 변환을 통한 송·수신이 필요하다. 이는 기존네트워크 기술을 그대로 이용하기 때문에 센서망으로 손쉬운 접속을 제공한다. 외부 네트워크 연동은 인터넷 프로토콜 또는 웹 프로토콜인 HTTP를 이용하여 송·수신 한다.[1][2]

무선 센서네트워크는 IEEE802.15.4 기반의 IPv6 over Low-power WPAN(6LoWPAN) 기술과 저전력 무선 라우팅 기술인 IPv6 Routing Protocol for Low power and Lossy Networks(RPL) 이 있다. 6LoWPAN 기술은 노드의 이웃탐색 방법과 RPL 프로토콜에 의해 최적의 라우팅 경로를 설립하는 기반 기술 이다.[3][4] 이와 함께 센서의 리소스 정보를 제공하기 위해 IETF CoRE 워킹그룹에서 6LoWPAN의 상위 응용계층을 표준화 한 Constrained Application Protocol(CoAP) 프로토콜을 제공 한다.[5]

CoAP은 Representational State Transfer(REST)기술을 지원하며, 센서노드의 리소스 정보를 HTTP와 같은 GET/POST/PUT/DELETE 메소드를 이용하여 데이터를 전송한다.[6][7]

외부 인터넷 망에 연결된 모니터링 시스템의 네트워크 프로토콜과 무선 센서네트워크의 프로토콜 차이로 인해 양단간의 네트워크 상황에 맞는 데이터 처리를 연구하게 되었으며, 이종간의 네트워크 연결을 위해 게이트웨이가 필요하게 되었다.

게이트웨이는 기본적으로 외부의 인터넷 망에 연결된 모니터링 시스템으로부터 요청 정보를 수신하며, 이는 연결되어 있는 6LoWPAN Border

Route(6LBR)를 통해 노드의 리소스 정보 또는 노드간의 연결 구성 정보를 요청하게 된다.

센서노드 리소스 수신 방법은 외부 네트워크에서 정보 요청을 수립하는 경우와, 게이트웨이에서 주기적으로 센서네트워크에 정보를 요청하는 방법인 폴링(Polling)이 있다. 센서네트워크 관리를 위한 토폴로지 구성 정보수집에 폴링 요청방식을 사용 할 경우, 센서에서 지속적으로 사용되는 무선 시그널링으로 인해 한정된 자원으로 동작하는 센서노드의 비효율적 배터리 소모가 발생 할 수 있다. 무선 센서는 대부분 한정된 자원으로 무인의 다양한 환경에서 장시간 동안 자율적으로 작동해야 함으로 효율적인 무선자원 사용을 고려해야 한다.

본 논문은 센서네트워크 망의 토폴로지 정보제공을 위해 노드의 트리 구성 정보를 기존 하향식 검색방법이 아닌 노드 자신이 최상위 부모로 데이터를 전송하는 상향식 방법을 제안한다. 센서네트워크의 토폴로지 정보 제공은 노드간의 데이터경로를 시각적으로 보여 줄 수 있으며, 노드의 물리적인 오류 및 링크 오류로 인한 센서네트워크 구성 정보를 이전 상태와 비교하여 노드의 단절 및 관리에 대한 효율적인 방안을 제공해 준다.

제 2 장 관련 연구

2.1 배경

최근 환경 및 장소에 관계없이 언제 어디서나 접속 가능한 무선센서네트워크 기술이 전 세계적으로 활발하게 연구 되고 있다. 무선센서의 설치편리성과 확장성으로 인해 빌딩, 공장 자동화 및 환경변화 측정 등의 융합 IT 기술과 결합 하여 새로운 산업 전개 발전 되고 있다.

유비쿼터스 환경의 PAN(Personl Area Network)에서 사용되는 대표적인 무선센서네트워크의 기술은 ZigBee와 6LoWPAN 기술이 있다. 동일한 물리계층을 사용하는 두 기술은 확장성에 있어 차이가 있다. ZigBee의 경우 16bit 네트워크 주소로 외부의 PAN과 연결시 노드의 주소 중복에 한계가 있는 반면, 6LoWPAN은 16bit 및 64bit 네트워크 주소의 유일성을 보장함으로써 전방위 네트워크 구조 설계가 가능 하다.

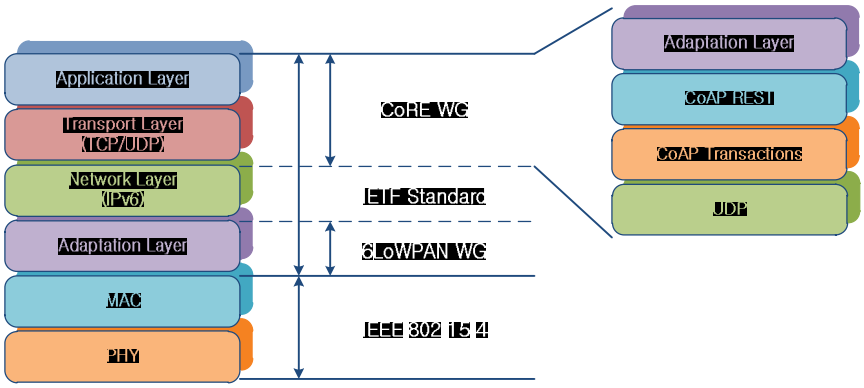
[표 2-1] ZigBee와 6LoWPAN 센서노드 기술 비교

	ZigBee	6LoWPAN
물리계층	IEEE 802.15.4	IEEE 802.15.4
네트워크 적용	홈오토메이션, 로컬네트워크	모든 네트워크 적용
주소 할당	16bit 네트워크 주소 할당 (외부 PAN과 연결 시 노드의 중복문제 내포)	16bit short 및 64bit 네트워크 주소 할당 (네트워크 중복 없음)
게이트웨이	모든 계층이 기존 IP구조와 다르기 때문에 게이트웨이 프로토콜 변환 처리 복잡성 증가	기존 IP 구조에 6LoWPAN 적응계층 추가로 인해 해당 프로토콜 변화 처리 필요
표준화	ZigBee Alliance 가입 사용자에게만 기술 배포	6LoWPAN WG에서 표준화 기술 배표

이로 인해 확장성 및 표준화 기술 개발을 위해 6LoWPAN 연구가 이루어 졌으며, 이와 더불어 외부 네트워크 연결을 위한 게이트웨이 기술 연구도 동시에 진행 되어 왔다. 게이트웨이 기술은 외부의 모니터링 시스템에서 센서네트워크와의 연결을 위해 프로토콜 스택 변환 처리 및 메시지 송수신, 노드의 상태 정보를 저장 처리 한다. 노드의 상태정보 및 연결구조 등에 대해 외부의 모니터링 시스템 또는 게이트웨이 자체에서 처리 하게 되며, 상태의 지속성을 유지하기 위해 일정간격으로 데이터 요청이 필요하다. 이와 같은 방법은 센서자원 사용의 비효율적 요인이 된다. 이로 인해 센서노드의 관리 및 네트워크 구조 관리를 위해 새로운 방안이 제시 되어야 한다.

2.2 무선센서 네트워크 프로토콜

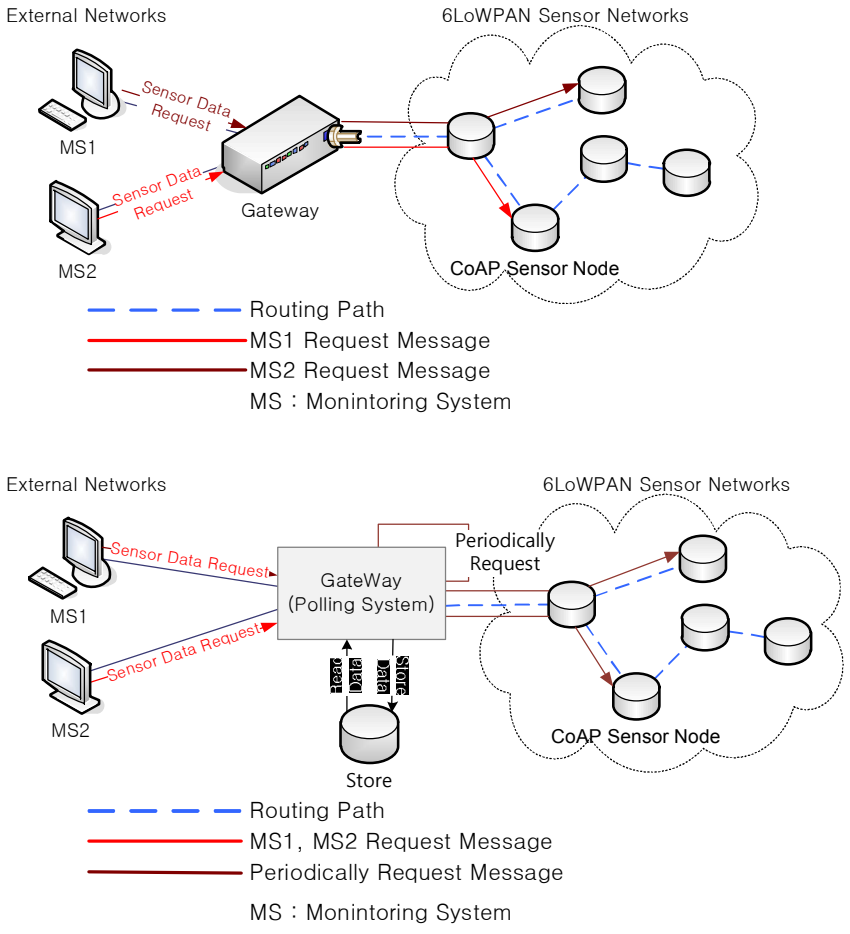
무선센서 네트워크 프로토콜은 [그림 2-1]과 같다. 물리계층(PHY)과 MAC계층의 저전력 저속도 무선 개인 통신망을 위해 IEEE 802.15 워킹그룹에서 표준스택으로 정의 하여 관리 하고 있다.[4] 기본적으로 인터넷 프로토콜 스택 계층과 유사하나 저전력 무선센서 라우팅을 위해 MAC 계층과 네트워크 계층 사이에 6LoWPAN 워킹그룹에서 정의한 적응계층(Adaptation Layer)의 추가와 전송계층과 응용계층 사이에 CoRE 워킹그룹에서 정의한 CoAP 관련 프로토콜 스택이 추가 되어 있다. 6LoWPAN 적응계층은 무선센서의 라우팅 프로토콜인 RPL과 노드의 이웃탐색(Neighbor Discovery Protocol) 방법에 대해 정의를 하고 있다. CoAP 스택은 센서노드의 리소스(온도, 습도, 조도등 센싱데이터) 이벤트처리를 위한 응용계층으로써 센서보드로 부터 검출된 리소스 정보를 이웃 및 최상위 부모노드로 전송을 처리하는 REST 기반의 프로토콜 스택 이다.



[그림 2-1] IP기반 무선센서 네트워크 프로토콜

2.3 센서네트워크 게이트웨이 구조

이번장에는 외부의 네트워크망에서 게이트웨이를 거쳐 무선 센서네트워크 망으로의 접속방법을 설명 한다. 그리고 원격접속을 위한 게이트웨이 프로토콜 구조에 대해 기술 한다.[8][9]



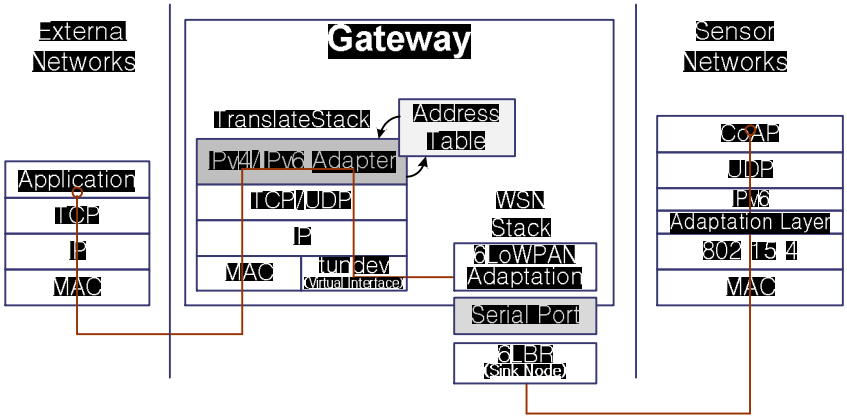
[그림 2-2] 센서네트워크 연동 구조

현재 외부네트워크와 무선 센서네트워크에서 사용 되는 프로토콜 스택의 차이로 인해 직접적으로 메시지를 송수신 할 수 없다. 이러한 문제로 외부네트워크와의 연동을 위해 게이트웨이를 이용한 접근방법이 제시 되었다. 게이트웨이는 외부 네트워크에 대해 인터넷프로토콜 소켓서버를 제공 하여 메시지 전송 하거나, HTTP와 같은 웹프로토콜을 이용하여 메시지를 전송하게 된다. [그림 2-2]은 외부 모니터링 시스템에서 게이트웨이를 통해 센서노드로 직접적인 요청 방법과, 게이트웨이에서 폴링에 의한 센서노드에 주기적 요청 방식을 보여 준다. 폴링에 의해 수집된 센서노드 정보는 게이트웨이의 캐시에 구조화 되어 저장 되며, 저장된 캐시데이터는 외부 모니터링 시스템을 위한 관리 정보로 사용 된다.

외부네트워크와 게이트웨이는 인터넷프로토콜인 IPv4/IPv6 소켓 또는 HTTP 프로토콜을 통해 메시지를 송수신 하게 되며, 센서노드와의 통신은 게이트웨이에 연결된 최상위 노드인 6LoWPAN Border Router(6LBR)와 시리얼통신을 통해 메시지를 송수신 하게 된다. 6LBR은 하나의 도메인내 최상위의 부모노드이며 IPv6 Global 주소에 대한 Prefix 정보설정을 수행 하는 Coordinator 노드이다.

2.4 프로토콜 연동구조

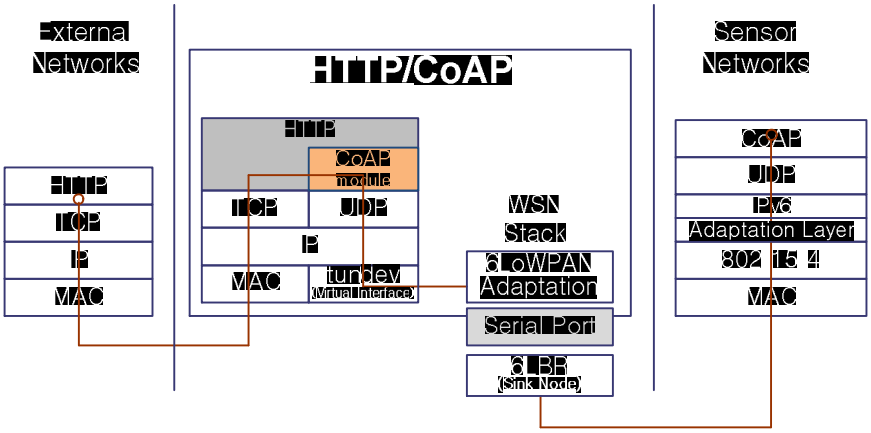
외부 모니터링 시스템과 센서네트워크 연동 게이트웨이 프로토콜 구조는 IPv4/IPv6 6LoWPAN과 웹 프로토콜인 HTTP-CoAP Proxy 서버를 이용한 무선 센서네트워크 접근 방법 등이 사용되고 있다. [그림 2-3]는 인터넷프로토콜 소켓을 이용한 IPv4/IPv6 6LoWPAN 주소변환방법 네트워크 구조를 보여 주고 있다.



[그림 2-3] 주소변환 게이트웨이 구조

모니터링 시스템은 IP소켓을 이용하여 게이트웨이의 특정 포토로 메시지를 요청하게 된다. 요청 메시지를 수신한 게이트웨이는 IPv4/IPv6 Adapter의 주소테이블에서 목적지 노드에 해당하는 주소로 변환 작업이 이루어진다. 이후 센서노드에 적합한 메시지를 생성하여 Virtual Interface인 tundev로 데이터를 송신하게 되면 6LoWPAN Adaptation Layer, USB시리얼 포트, 6LBR노드를 거쳐 무선 센서네트워크로 데이터 전송이 완료 된다.

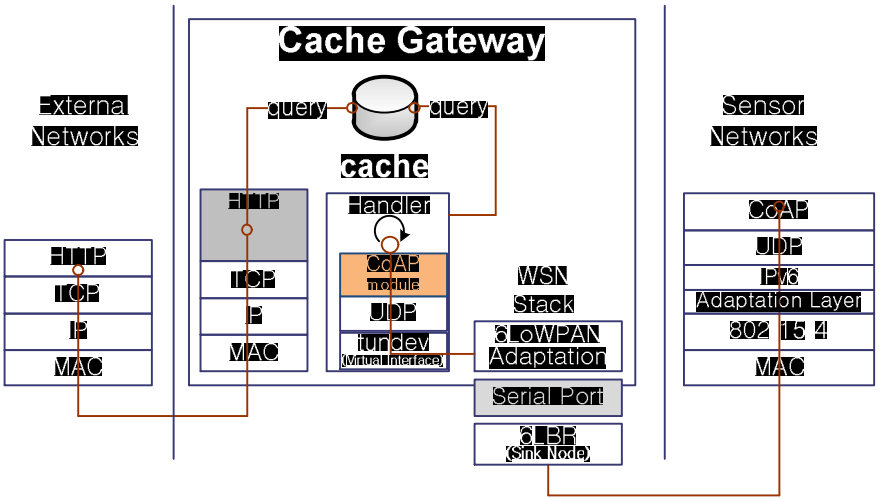
HTTP-CoAP Proxy는 HTTP 메시지를 수신 받아 CoAP 프로토콜을 사용 하여 센서노드에 CoAP 서버로 데이터를 전송 한다. [그림 2-4]은 웹 표준화 프로토콜을 이용한 게이트웨이 구조 이다. HTTP는 잘 알려진 표준화 프로토콜로써 원격지 모니터링 시스템에서 게이트웨이 접속 방법을 제공해 준다.



[그림 2-4] HTTP/CoAP 프록시 서버 구조

[그림 2-3], [그림 2-4]의 프로토콜 구조에서 센서노드의 토폴로지 구성 정보를 수집하기 위해 데이터를 요청하게 되며, 노드의 링크 오류가 발생 할 경우 비정상적인 대기 현상이 발생하게 된다. 이러한 문제로 인해 게이트웨이에서의 폴링 기능과 센서정보 데이터의 캐시 저장을 통해 해결 하려는 연구도 진행 되었다. [그림 2-5]는 센서네트워크에 대한 주기적 요청 핸들러와 캐시 저장 기능이 포함된 게이트웨이의 구조를 보여준다.[10] 게이트웨이는 캐시관리와 센서노드의 정보를 검색하기 위해 주기적 요청처리 기능을 담당하는 핸들러가 포함되어 있다. 캐시는 저장

및 삭제 등의 질의메시지를 처리하기 위한 데이터베이스이며, 주기적 요청 핸들러는 센서노드로부터 정보를 수신한 후 데이터를 분류하여 캐시 데이터베이스에 저장하는 기능을 담당 한다. 게이트웨이는 모니터링 시스템 요청에 대해 캐시에 저장된 데이터를 반환 하거나 캐시관리 핸들러에 의해 센서노드의 변경 및 추가사항을 모니터링 시스템으로 전달한다.



[그림 2-5] 주기적 요청을 통한 데이터 캐시화 구조

앞서 언급한 구조를 통해 정보를 수집 하고 데이터를 구조화 하는 방법으로 센서네트워크 토폴로지 구성이 가능하다. 이와 같이 주기적 요청에 의한 센서정보 수집은 센서노드의 토폴로지 변경과 무관하게 작동함으로 지속적인 무선자원 사용을 가중 시켜 배터리 소모에 있어 비효율적일 수 있다. 자원의 효율적인 사용을 위해 모니터링 시스템 또는 게이트웨이에서의 구조설계가 아닌 노드간의 연결 정보를 비동기 메시지 전달을 이용한 네트워크 설계가 필요하다.

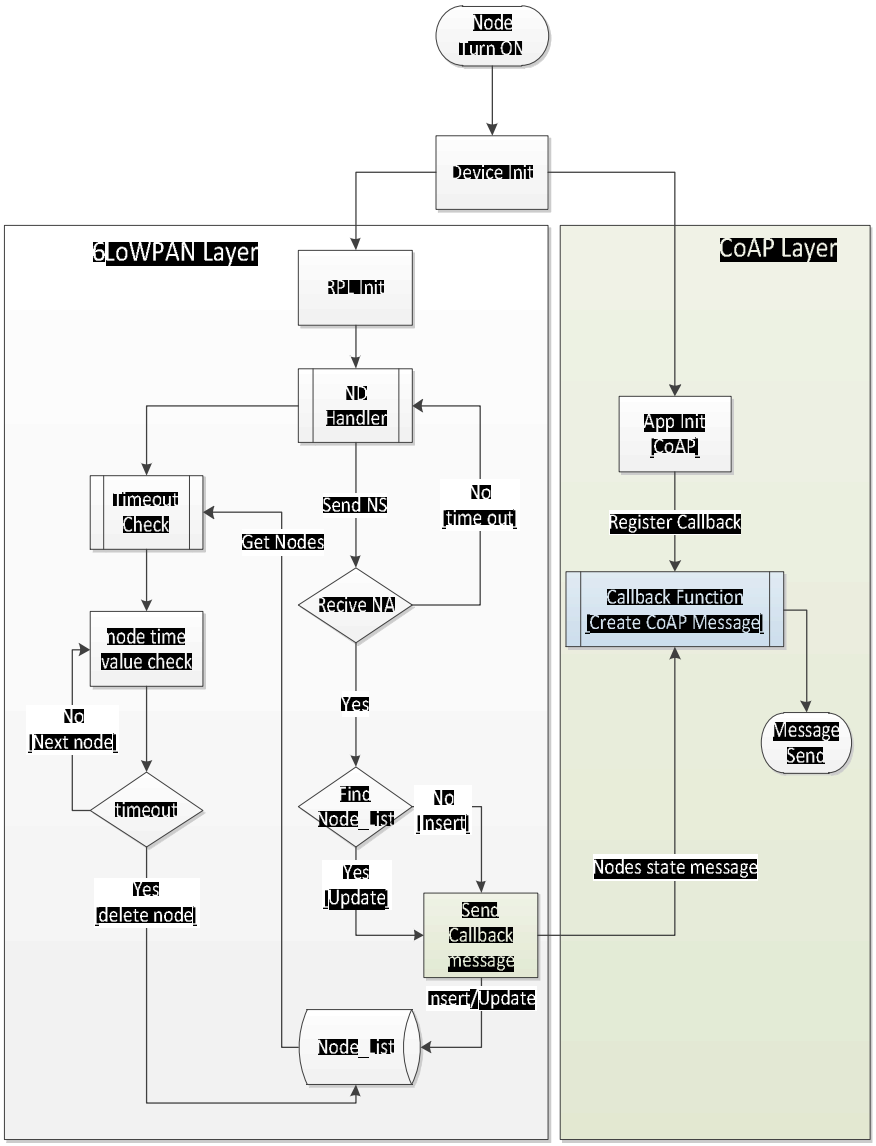
제 3 장 설계

3.1 센서노드 토폴로지 정보제공 설계

외부 모니터링 시스템에서 센서네트워크 토폴로지 정보를 표현하기 위해서는 노드간의 연결 정보를 제공 받아야 한다. 이를 위해 센서노드는 자신의 노드정보 및 이웃노드 정보를 비동기 전송 방식으로 최상위 노드, 즉 게이트웨이로 전송해야 하며, 노드의 추가, 삭제 및 정보변경에 대한 이벤트 처리와 CoAP응용계층을 통한 정보전달이 가능한 시스템으로 설계되어야 한다.

우선 6LoWPAN의 적응계층에서는 이웃탐색 방법으로 무선링크상에 연결된 주변의 이웃노드에게 자신의 MAC주소 정보를 포함한 메시지를 전송하고, 노드들로 부터 응답 메시지 정보를 수신 후 정보를 저장하게 된다. 이러한 이웃탐색 기능은 노드의 링크 상태 및 연결된 노드 정보를 저장한 후 상위계층인 CoAP 계층으로 통지 하게 된다. CoAP에서는 변경된 정보를 최상위 노드에 전송하게 되며, 이는 토폴로지 정보를 구조화 할 수 있는 데이터이다. [그림 3-1]은 센서노드 동작 시 6LoWPAN 적응계층과 CoAP계층 사이에서의 정보 전달 구조이다.

최초 센서노드 부팅 시 노드의 기본적인 부분들이 초기화 된다. 이와 동시에 6LoWPAN 적응 계층에서는 라우터 프로토콜인 RPL 기능을 초기화 하며, RPL에 포함된 Neighbor Discovery Handler(ND 핸들러) 가 동작 하게 된다. ND 핸들러에서는 멀티캐스트로 Neighbor Solicitation(NS) 메시지를 전송하며, 그 응답으로 Neighbor Advertisement(NA) 메시지를 전송 받게 된다. 수신된 NA 메시지의 노드주소가 Node_List 테이블에 존재 하는지 검색 하게 된다.



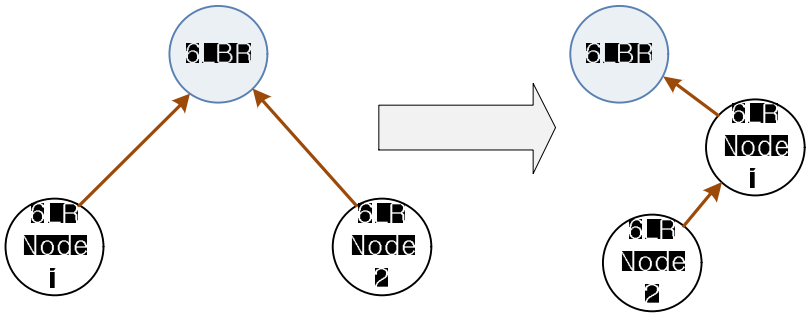
[그림 3-1] 센서노드의 동작 방법

Node_List에 매칭되는 주소가 있는 경우 그 주소에 대한 상태를 업데이트 처리 하게 되며, 존재 하지 않는 경우 새로운 노드 발견으로 저장

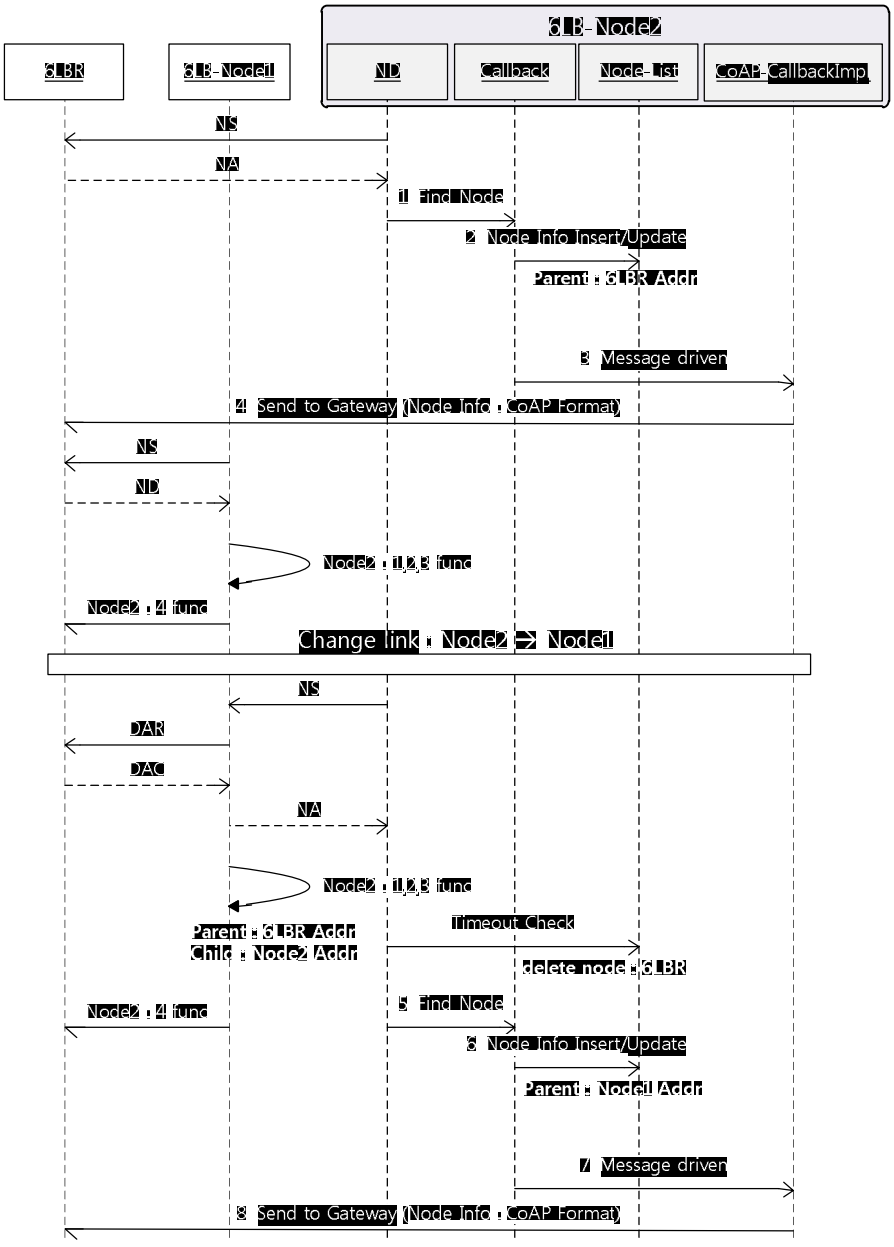
하게 된다. NS/NA 메시지 송·수신, 그리고 발견된 노드에 대한 정보 처리와 같은 일련의 작업들은 ND 핸들러에서 주기적으로 일어난다. ND 핸들러에서 NS/NA 기능 이외에 TimeoutCheck 기능이 주기적으로 동작한다.

TimeoutCheck는 Node_List 테이블에서 존재하는 모든 노드의 정보를 가져와 각 노드의 참조된 시간정보를 확인하여 일정시간 동안 참조되지 않은 노드의 정보를 제거한다. 이와 같이 ND핸들러에서는 노드의 추가, 삭제와 같은 이벤트 발생 시 상위계층으로 통지하기 위해 콜백(Callback)함수를 제공해야 한다. CoAP계층에서는 콜백함수를 구현함으로써, ND핸들러에서 제공하는 이벤트 정보를 수신 할 수 있다. CoAP계층의 등록된 콜백함수에서는 적응계층에서 전달된 노드의 주소정보를 CoAP 메시지 포맷으로 재정의하여 최상위 노드로 전송하게 되며, 이 정보를 이용하여 센서네트워크의 토폴로지 정보를 구성하게 된다.

[그림 3-2], [그림 3-3]은 노드의 링크변화에 대한 6LoWPAN 적응계층에서의 이벤트 발생 흐름을 보여준다. 6LR Node1과 6LR Node2는 ND핸들러에서 콜백 기능이 동일하게 포함된 6LoWPAN Router 노드이다. 이 노드는 최상위 부모노드인 6LoWPAN Border Router(6LBR)로 링크연결을 요청한다.



[그림 3-2] 센서노드 링크 변화

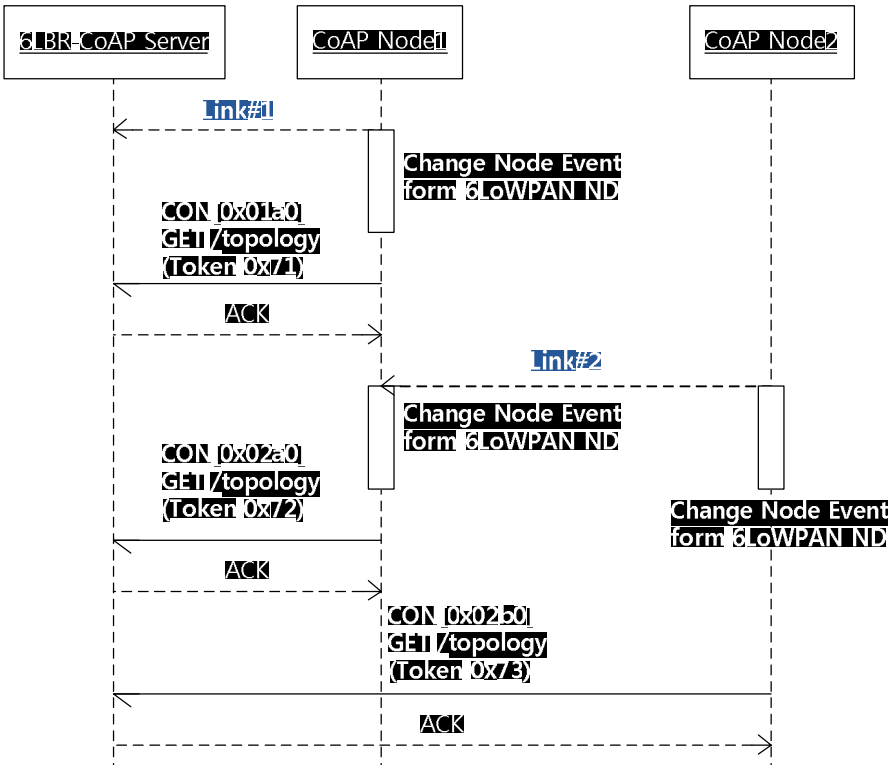


[그림 3-3] 링크변화에 대한 비동기 메시지 전송 흐름도

최초 6LR Node2는 ND핸들러에 의해 6LBR과 NS/NA 메시지 송수신을 하게 된다. 6LBR로부터 수신된 NA 메시지에서 부모주소인 6LBR 주소정보와 참조시간이 포함된 구조체 정보를 Node-List에 저장한다. 각 주소별 참조시간은 일정시간 동안 참조가 없는 경우, 해당 정보를 제거하기 위해 사용되는 정보이다. 노드의 주소저장이 완료되면 변경사항에 대해 CoAP계층에서 등록한 콜백함수를 호출하게 된다. 이 함수에서는 Node-List를 참조하여 자신노드와 연결된 부모노드, 자식노드의 주소정보를 최상위 노드인 6LBR로 보내게 된다. 6LBR에서 수신한 정보는 요청된 소스주소, 그리고 Payload에 포함된 부모노드와 자식노드의 주소를 게이트웨이로 보내게 된다. 노드의 링크가 Node2에서 Node3으로 변경되어도 동일한 방법으로 이벤트를 처리 하게 된다.

3.2 CoAP 프로토콜 비동기 전송 설계

CoAP 노드에서 최상위 부모노드로 전달되는 메시지는 부모노드, 그리고 연결된 자식노드의 주소이다. [그림 3-4]는 CoAP 프로토콜을 이용한 메시지 전송 방법이다.



[그림 3-4] CoAP노드 주소 전송

메시지 요청은 응답이 필요한 Confirmable(CON)을 사용한다. 본 제안에 따른 CON 메시지 전송은 토폴로지 정보가 최상위 부모노드에 정상적으로 전송되었는지 ACK응답을 요구하는 전송방법이며, ACK응답을 받지 못한 경우 최대 3회까지 메시지 재전송 루틴이 실행된다.

[그림 3-5]은 노드의 링크상태 변경에 따른 CoAP 메시지 전송에 대한 정의를 보여 준다.

- Link#1
REQUEST
Header: GET (T=CON, Code=1, MID= 0x01a0)
Token: 0x71
Uri-Host: "coap://[aaaa::212:7400:13b7:677c]"
Uri-Path: "topology"
Payload: "CoAP Server addr"

RESPONSE
Header: (T=ACK, Code=0, MID=0x01a0)

- Link#2
REQUEST
Header: GET (T=CON, Code=1, MID= 0x02a0)
Token: 0x72
Uri-Host: "coap://[aaaa::212:7400:13b7:677c]"
Uri-Path: "topology"
Payload: "CoAP Server addr""CoAP Node2 addr"

RESPONSE
Header: (T=ACK, Code=0, MID=0x02a0)

REQUEST
Header: GET (T=CON, Code=1, MID= 0x02b0)
Token: 0x73
Uri-Host: "coap://[aaaa::212:7400:13b7:677c]"
Uri-Path: "topology"
Payload: "CoAP Node1 addr"

RESPONSE
Header: (T=ACK, Code=0, MID=0x02b0)

[그림 3-5] 이벤트 발생에 대한 CoAP 링크포멧

[표 3-1] 센서 토폴로지 구성을 위한 CoAP 메시지 포멧

정의	설명	값
Method	요청에 대한 리소스 생성	GET
Transaction	요청에 대한 ACK 여부	ACK
MID	중복 메시지 검출용 ID	16bit hex
Token	요청에 대한 응답 일치 확인	0 to 8 bytes long
Uri Host	최상위 노드의 IPv6 주소	[aaaa::x]
Uri Path	요청 리소스 처리 위치	[topology]
Payload	전송 데이터	주소리스트

Payload는 전송 메시지 데이터로, 토폴로지 구성을 위한 센서주소의 리소스 정보로 구성 된다. 정보구조는 각각의 주소들에 대해 16바이트의 리스트 형식으로 정의 되어 있다. 이와 같은 CoAP 메시지를 정의하여 전달함으로써 송수신을 위한 리소스 메시지 생성 및 해석이 쉬워지며, 표준화된 메시지 사용으로 센서네트워크 토폴로지 관리에 효율적인 방법을 제공해 준다.

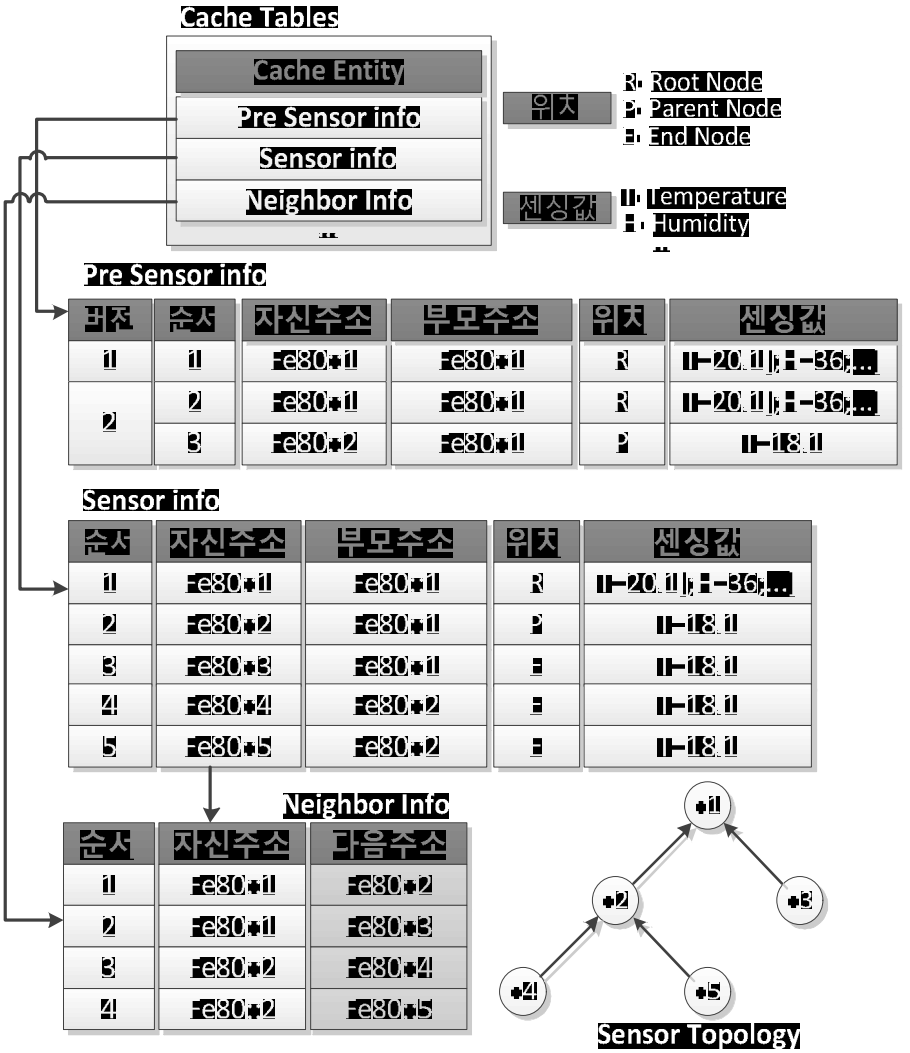
3.3 캐시 설계

센서네트워크의 노드로부터 수신된 CoAP 메시지 정보를 외부의 모니터링 시스템으로 전송하기 위해서는 데이터를 분석하여 파싱(Parsing) 해야 한다. 파싱된 데이터를 모니터링 시스템으로 직접적으로 전달할 경우 모니터링 시스템에서 해당 데이터를 정상적으로 수신 받지 못하거나 또는 모니터링 시스템이 게이트웨이에 접속 하였을 때 센서네트워크 정보를 실시간으로 받지 못 할 수 있다. 이러한 이유로 노드에서 전송되는 토폴로지 메시지 정보를 캐시에 저장 하여 서비스를 제공 해야 한다.

[그림 3-6]은 토폴로지 구성을 위한 캐시구조체이다. 캐시 구조체에는 Sensor Info와 Neighbor Info 그리고 Pre Sensor Info 구조체로 되어 있다. Sensor Info는 주 구조체이며, 최상위 노드에 연결된 모든 센서들의 리스트로 구성되어 있다. 구조체 항목은 센서 자신의 주소, 부모주소, 상태값, 그리고 센서의 센싱정보로 구성되어 있다. 이 정보들은 하나의 엔트리로 구성되며, 상태값은 부모노드의 여부를 판단하는 값이다. Neighbor Info는 Sensor Info 엔트리에 대한 부 구조체이며, 자신의 주소에 연결된 자식노드에 대한 주소값이 정의된 구조체이다. Pre Sensor Info는 토폴로지 정보에 대한 이전 정보를 가진 구조체 이다. 이 구조체는 노드의 생성 및 변경에 대해 이전 토폴로지 구성 전체를 스냅샵(Snapshot) 하며, 버전정보로 그룹화하여 저장한다. 저장되는 그룹의 개수는 최대 열개이며, 열개이상의 그룹정보를 저장 할 경우 가장 낮은 버번 정보의 그룹 데이터를 제거하고 삽입 하게 된다.

모니터링 시스템을 위한 토폴로지 구성은 Sensor Info구조체의 데이터를 이용하여 구성 할 수 있다. Sensor Info의 위치항목에서 최상위 부모노드(R)를 검색하여 자신주소를 가져 온다. 다음으로 부모주소 항목에서

최상위 부모노드의 주소를 가진 엔트리들을 모두 가져온다. 이로 인해 최상위 부모노드에 연결된 자식 노드를 구성 할 수 있으며, 자식노드의 자신주소를 이용 하여 부모주소항목에서 일치되는 데이터를 재차 검색하게 된다. 이러한 재귀적 방법으로 데이터를 검색 하면서 센서네트워크 망에 대한 토폴로지를 구성하게 된다.



[그림 3-6] 게이트웨이 캐쉬 구조체

현재의 토폴로지 구성과 비교하기 위한 이전 토폴로지 검색은 Pre Sensor Info구조체에서 최상위 값의 버전정보를 가진 그룹데이터가 직전의 토폴로지 정보 이며, 이전 토폴로지 구성을 위한 토폴로지 검색은 Sensor Info 검색 방법과 동일하게 진행 된다.

제 4 장 구현

[표 4-1]은 6LoWPAN 계층에서 노드의 연결상태 변경에 따른 콜백함수 호출에 대한 구현 부분 이다.

[표 4-1] 노드상태에 따른 콜백 함수

함수	설명
neighbor_state_changed	- ND의 NS/NA 메시지 따른 노드발견 호출 - TimeoutCheck에 따른 노드삭제 호출
state_callback	neighbor_state_changed의한 변경요청 처리
nbr_state_subscribe	CoAP계층에서 콜백함수 등록 처리

[그림 4-1]는 노드간의 연결상태 확인을 위한 TimeoutCheck 구현 함수이다. TimeoutCheck 함수는 30000ms 간격으로 “nbr_cache”테이블에서 자신의 노드와 연결된 노드의 참조된 시간을 조사한다. 참조 시간은 input packet에 의해 업데이트되는 정보이며, 오랫동안 업데이트가 이루어 지지 않을 경우, 해당 주소로 icmp6 메시지를 전송하게 된다. icmp6 메시지 전송 후에도 참조시간 변경되지 않는다면, 해당 데이터의 사용유무 필드값을 FALSE로 변경하게 된다. 사용유무가 FALSE로 변경된 데이터는 삭제된 데이터로 인식하며, 재연결 되기 전에는 더 이상 참조 하지 않는다.

```
typedef void (*nbr_state_t)(uint8_t ismode);
static nbr_state_t s_callback = 0;

// Register Callback func
void nbr_state_subscribe(nbr_state_t s) {
     if(s_callback == NULL) {
        s_callback = s;
     }
}

// Register callback function call
void state_callback(uint8_t ismode) {
     if(s_callback) {
          s_callback(ismode);
     }
}

// Neighbor state change callback func
void neighbor_state_changed(nbr_t *nbr){
  if(nbr->isused) { // Adding neighbor
     process_add_neighbor(nbr);
  } else { // Removing neighbor
     process_removeing_neighbor(nbr);
  }
  // function call to app
  state_callback(nbr->isused);
}

 void nbr_timeout(void)
{
#define TIME_SEC 30000

  u8_t i;
  for(i = 0; i < DS6_NBR_LIST; i++) {
    if(nbr_cache[i].isused &&
       (nbr_cache[i].ipaddr.u8[15]
         + nbr_cache[i].ipaddr.u8[16]) > 0)
    {
         clock_time_t curr_time =  clock_time();
         clock_time_t off_old_time =
           nbr_cache[i].last_time + TIME_SEC;
```

```
        if (off_old_time < curr_time)
        {
          nbr_update_time(nbr_cache[i].ipaddr);
          icmp6_send(nbr_cache[i].ipaddr,
                     ICMP6, DAO_ACK, 4);
        }
      }
    }
}
```

[그림 4-1] 노드상태에 따른 콜백과 TimeoutCheck 구현 함수

[그림 4-2]는 CoAP계층에서의 콜백등록, 처리함수를 보여 주고 있다. 노드상태의 정보수집을 위한 콜백함수는 CoAP 관련기능이 초기화 된 후 등록하게 된다. 이후 ND핸들러에서 노드상태 이벤트 발생 시 등록된 콜백함수를 호출하게 되며, 구현된 콜백함수에서는 최상위 노드로 전송할 CoAP 패킷 데이터를 초기화 시킨다. URI의 헤더정보는 “topology”옵션태그로 정의되어 있으며, 6LBR의 CoAP서버에서 해당 옵션태그와 매칭 하여 처리하게 된다. CoAP 패킷에 포함되는 데이터는 부모노드와 자식노드의 주소이다. 부모노드는 DAG트리에서 자신의 노드와 연결된 주소를 검색하며, 자식노드는 이웃노드 정보테이블에서 사용 가능한 주소를 검색해서 가져온다. 이 데이터는 CoAP패킷의 Payload 값으로 정의된다. CoAP 패킷의 데이터 정의가 완료되면 UDP 소켓을 사용하는 “packet_send” 통해 최상위 부모노드로 전송하게 된다. CoAP 프로토콜로 전송 받은 부모노드는 연결되어 있는 게이트웨이로 데이터 전송이 이루어진다.

```
void app_init(void) {
  nbr_state_subscribe(nbr_callback);
}


void nbr_callback(uint8_t ismode) {
  /*default coap + node addr*/
#define MAX_PAYLOAD_LEN 164
#define COAP_BUFF_SIZE 300

  int i, j=1;
  int addr_size, data_size = 0;
  static int xact_id;
  uint8_t buf[MAX_PAYLOAD_LEN];
  uint8_t paybuf[64];
  memset(&buf, 0, sizeof(buf));
  memset(&paybuf, 0, sizeof(paybuf));

  if (init_buffer(COAP_BUFF_SIZE)) {
    dag_t *dag;
    parent_t *pp;
    coap_packet_t* req
    init_packet(req);
    coap_set_method(req, COAP_GET);
    req->tid = xact_id++;
    req->type = MESSAGE_TYPE_CON;
    coap_set_header_uri(req, "topology");

    dag = get_any_dag();
    if (dag) {
      pp = dag->preferred_parent;
      if(pp != NULL) {
        nbr_t *nbr = nbr_lookup(&pp->addr);

        if(nbr != NULL) {
          addr_size = sizeof(nbr->ipaddr);
          memcpy(&paybuf,
                 &nbr->ipaddr,
                 sizeof(nbr->ipaddr));
```

```
            for(i = 0; i < NBR_NB; i++) {
              if(nbr_cache[i].isused) {
                if (nbr->ipaddr
                        == nbr_cache[i].ipaddr)
                    continue;

                memcpy(&paybuf[addr_size*j],
                        &nbr_cache[i].ipaddr,
                        sizeof(addr_size));
                j++;
              }
            }
            req->payload = &paybuf;
            req->payload_len = sizeof(nbr->ipaddr) * j;
          }
        }
      }
      data_size = serialize_packet(req, buf);
      packet_send(client_addr, buf, data_size);
      delete_buffer();
    }
}
```

[그림 4-2] 응용계층에서의 CoAP 메시지 전송 구현 함수

제 5 장 성능 분석

이번 장에서는 기존에 제안된 외부 네트워크 또는 게이트웨이에서의 주기적 요청 방법과 본 논문에서 제안한 방법을 비교 분석한다. 비용분석은 무선 시그널링 부분과 에너지 소모량 부분으로 구분하여 분석하였다.

5.1 시그널링 비용

시그널링 비용을 계산하기 위해 토폴로지 구성을 위한 노드에서 송수신 되는 모든 시그널링과 배터리사용 비율을 고려하였다. 각 방법에 대한 총 시그널링 비용은 C_{schme}로 나타내며, 메시지 발생에 따른 총 송수신 처리비용은 P_c로, 메시지 데이터 처리에 대한 배터리 총 소모량은 B_q로 정의하였다. 이에 대한 표현은 [식 5-1]과 같다.

$$C_{schme} = P_c + B_q \qquad \text{[식 5-1]}$$

노드의 초기시간부터 특정시간까지 토폴로지 구성을 위해 최상위 부모노드에서 말단노드로 검색하는 경우 또는 노드가 부모노드로 발생시키는 메시지 총 처리비용인 P_c는 [식 5-2]로 정의 된다.[11]

$$P_c = N_t[P_t(T_{on} + T_{st}) + P_{out}(T_{on})] + N_r[P_r(R_{on} + R_{st})] \qquad \text{[식 5-2]}$$

- $P_{t/r}$: 송수신시 발생하는 소모량
- P_{out} : 송신기의 출력 전력
- T/R_{on} : 특정 시간의 송수신기 값
- T/R_{st} : 시작 시간의 송수신기 값
- $N_{T/R}$: 특정 시간의 누적 송수신 메시지 수

데이터 처리에 대한 배터리 총 소모량은 [식 5-3]으로 나타내며, S_{nodes}는 노드들에서 발생한 총 메시지 수로, P_p는 단위 메시지에 대한 에너지 소모량을 나타낸다.

$$P_{quant} = S_{nodes} \times P_p \qquad \text{[식 5-3]}$$

단위 메시지에 대한 에너지 소모량인 P_p는 [식 5-4]로 정의 된다.

$$P_p = CV_{dd}^2 f + V_{dd} I_0 e^{V_{dd}/n' V_T} \qquad \text{[식 5-4]}$$

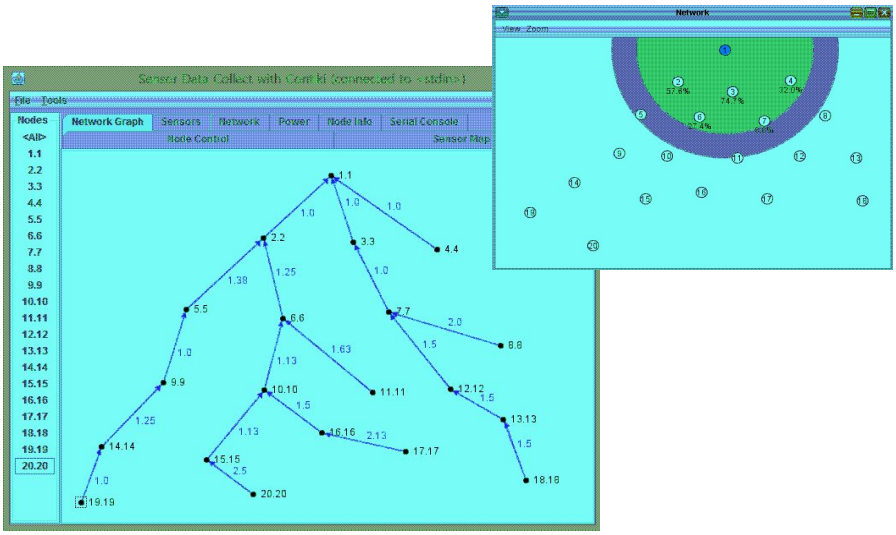
- C : 총 변환 용량 값
- V_{dd} : 전압 변동 값
- f : 특정 시간의 송수신기 값
- $V_{dd} I_0 e^{V_{dd}/n' V_T}$: 누전으로 인한 전력 손실

제안방식은 초기 시그널링이 발생하여 토폴로지 구성이 완료될 시점까지의 시그널링이 증가하지만 노드의 링크변화가 없는 경우 시그널링이 발생하지 않기 때문에 배터리 소모량이 누적 되지 않는다. [표 5-1]와 같

은 구성으로 적용 하여 비교 분석 하였으며, [그림 5-1]은 센서네트워크에 대한 Cooja 시뮬레이션 실행 환경을 보여 준다.

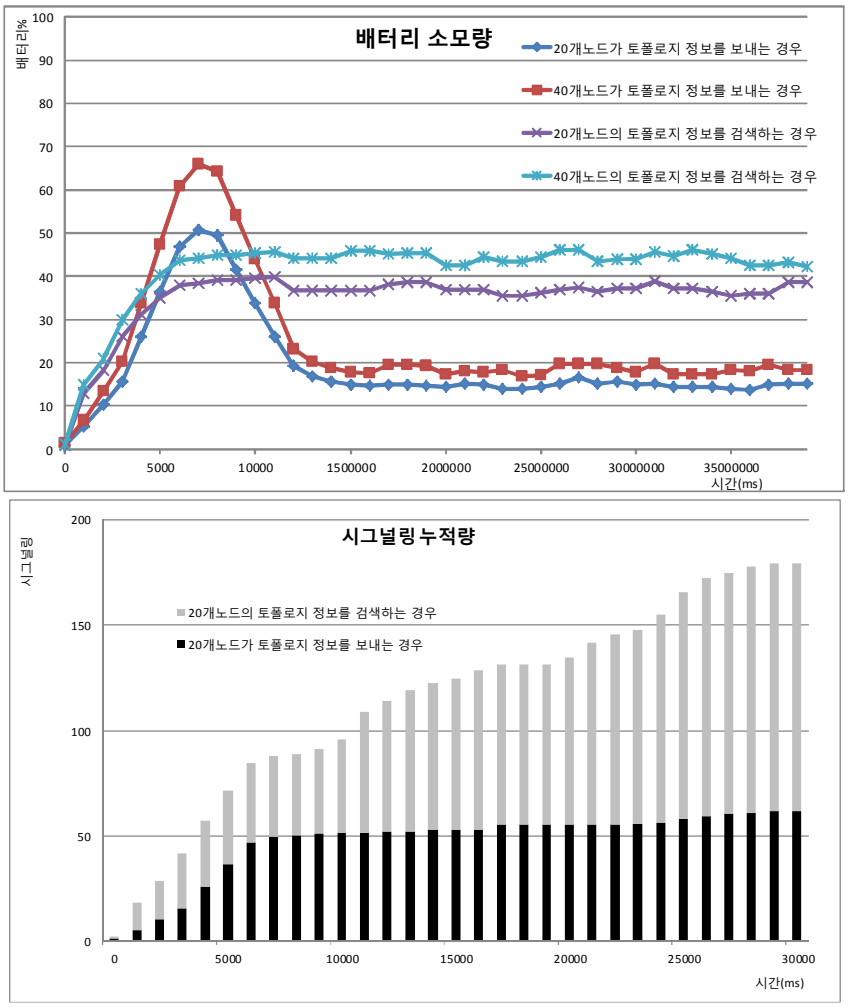
[표 5-1] 비교분석 구성

정의	설명
센서 OS	센서네트워크 오픈소스인 Contiki OS 2.6
시뮬레이션	Cooja for Contiki OS 2.6
센서노드	20, 40
요청 방법	- 최상위 노드에서 주기적 요청 - 센서노드에서 최상위 노드에게 요청



[그림 5-1] 센서네트워크 망의 구조

비교 방법은 시뮬레이션에서 6LR 노드에게 토폴로지 구성을 위한 질의·응답 통신요청 방법과 각각의 6LR 노드가 초기화 이후 이웃탐색 시 노드를 발견 하고 6LBR 노드로 토폴로지 정보를 보내는 시그널링과 시간소비에 대한 소비전력을 측정 하였다.



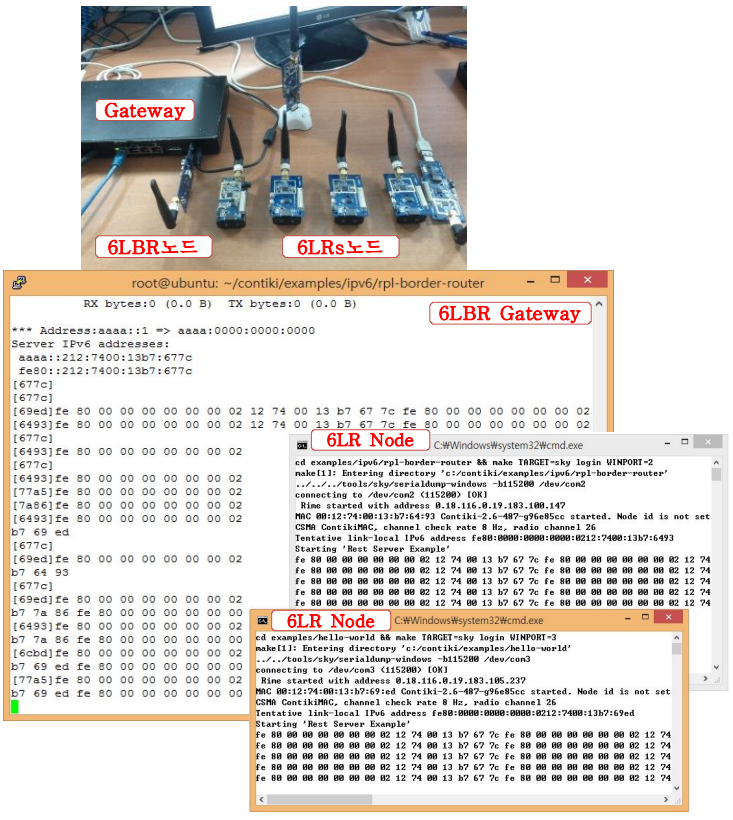
[그림 5-2] 배터리 소모량과 시그널링 누적 성능비교

[그림 5-2]성능비교 표의 주기적 요청의 경우 전체 노드의 검색을 이진트리 검색과 동일한 방법으로 부모노드에서 자식노드의 경로로 탐색하며, 일정간격으로 동일한 탐색방법을 사용하기 때문에 시간이 증가 할수록 무선 시그널링수의 누적과 배터리 소모량이 증가하는 것을 볼 수 있다. 반면에 노드의 정보를 전송하는 방법은 노드의 초기화 작업 이후 이웃탐색에 의한 노드추가 이벤트가 발생 할 때 최상위 부모노드로 정보를

보내게 되며, 노드의 수 및 DAG트리에 따라 초기 다량의 시그널링이 발생한다. 이는 초기 다량의 무선자원 사용에 의한 소모전력이 증가하는 것을 볼 수 있으며, 시간의 증가에 의한 무선 시그널링의 누적 및 소모 전력도 변화 없는 것을 볼 수 있다.

5.2 테스트 베드

실 노드에 대한 실험은 [그림 5-3]과 같이 6LBR의 게이트웨이의 로그 정보와 PC의 시리얼포트에 접속된 6LR의 정보를 보여주고 있다. 6LBR의 로그에서 “[ ]”내 포함된 데이터는 패킷을 전송한 노드의 16바이트중 마지막2바이트며, 나머지 데이터는 CoAP 메시지의 Payload로 16바이트 부모주소, 16바이트 자식노드의 주소로 구성되어 있다. 부모주소와 자식주소는 1:N으로 구성된다. 게이트웨이의 로그에서 센서노드의 링크변동이 없기 때문에 더 이상 정보가 수신되지 않는 것을 확인 하였다.



[그림 5-3] 테스트 베드 구축 실험

제 6 장 결론

모니터링 시스템을 위한 센서네트워크 토폴로지 정보는 센서노드의 오류 및 시스템 코드 업데이트에 시각적인 네트워크 흐름을 보여 줌으로써 관리를 용이하게 해 준다. 이에 본 논문에서는 토폴로지 정보를 구성하기 위해 게이트웨이에서 폴링에 의한 검색이 아닌, 노드의 이웃탐색방법에 의한 노드추가 정보를 게이트웨이로 보낼 수 있도록 설계하고 구현하였다. 그리고 노드의 오류 발생 시 노드삭제 이벤트 정보를 게이트웨이에 보내기 위해 일정시간 동안 참조되지 않은 노드를 관리 할 수 있도록 구현 하였다.

구현은 6LoWPAN 계층에서 발생하는 비동기 이벤트에 대해 콜백함수를 정의하여 제공하며, 상위계층인 CoAP에서 제공된 함수를 구현 하여 CoAP 메시지 포맷으로 최상위 노드로 전송하는 구조이다.

이와 같이 제안구조의 구현을 통해 기존 검색방법보다 무선자원의 사용을 줄임으로써 배터리 사용의 효율화를 증명하였으며, 시뮬레이션을 통해 시간이 지남에 따라 무선시그널이 증가하지 않음을 확인 하였다.

참고문헌

[1] K.A. Emara, M. Abdeen, M.Hashem, “A gateway-based framework for transparent interconnection between WSN and IP network,” EUROCON 2009, EUROCON '09. IEEE, May. 2009.

[2] C. Bormann, A.P. Castellani, Z. Shelby, “CoAP: An Application Protocol for Billions of Tiny Internet Nodes,” Internet Computing IEEE, Volume 16, Issue 2, pp. 62-67, Mar. 2012.

[3] Z. Shelby, S. Chakrabarti, E. Nordmark, C. Bormann, “Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs),” RFC6775, Nov. 2012.

[4] P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, R. Alexander, “RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks,” RFC6550, Mar. 2012.

[5] Z. Shelby, K. Hartke, C. Bormann, “Constrained Application Protocol (CoAP),” draft-ietf-core-coap-14, Sep. 2013.

[6] A. Castellani, S. Loreto, A. Rahman, T. Fossati, E. Dijk, “Best Practices for HTTP-CoAP Mapping Implementation,” http://tools.ietf.org/html/draft-castellani-core-http-mapping-07, CoRE Working Group

[7] Ngoc-Thanh Dinh, Younghan Kim, ‘RESTful Architecture of Wireless Sensor Network for Building Management System,’Special issue in M2M communications - KSII trans., Volume 6, issue 1, pp. 46-63, Jan. 2012.

[8] B. da Silva Campos, J.J.P.C. Rodrigues, L.D.P. Mendes, E.F.

Nakamura, C.M.S. Figueiredo, “Design and Construction of Wireless Sensor Network Gateway with IPv4/IPv6 Support,” Proc. of IEEE International Conference on Communications (ICC) 2011, Kyoto, Japan, Jun. 2011.

[9] V. Aishwarya, V.S. Felix Enigo, “IP based wireless sensor networks with web interface,” Proc. of International Conference on Recent Trends in Information Technology (ICRTIT) 2011, Chennai, India, Jun. 2011.

[10] YoungBag Moon, JongYoung Lee, SangJoon Park, “Sensor Network Node Management and Implementation,” Advanced Communication Technology, 2008. ICACT 2008, Feb. 2008.

[11] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci “Wireless sensor networks: a survey,” Computer Networks, Volume 38, Issue 4, pp. 393-422, Mar. 2002.