DOI: 10.3745/KIPSTC.2009.16C.5.653

무선 센서 네트워크에서 미들웨어 서비스를 제공하는 센서 오버레이 네트워크

김 용 표[†] · 정 의 현[††] · 박 용 진[†††]

요 약

센서 네트워크 미들웨어는 센서 하드웨어와의 연관을 최소화하고, 어플리케이션 개발을 용이하게 하며, 추상적인 데이터 접근 방법을 제공하기 위한 연구들로 많은 연구자들에게 관심을 받고 있다. 그러나 기존의 미들웨어는 실행코드를 해석하기 위한 해석기가 모든 노드에 설치되어야 하며, 이는 추가적인 컴퓨팅 및 통신 오버헤드를 야기하였다. 이를 해결하기 위하여 본 논문에서는 게이트웨이에서 센서 오버레이 네트워크를 이용하여 운용되는 TinyONet-Lite를 제안하였다. TinyONet-Lite에서 가상 센서는 물리적 센서의 가상 대응체로 동작하고, 동적으로 연합하여 오버레이 네트워크인 슬라이스(Slice)를 만들어 미들웨어 서비스를 제공한다. 플랫폼의 효율을 보여주기 위해서 이 논문은 TinyOS를 가진 하드웨어 모트(mote)위에 TinyONet-Lite를 구현했고, 여러 실험을 수행하였다. 실험에 의하면, 기존의 연구와 비교했을 때 TinyONet-Lite는 확장성, 동적서비스 구성이 우수하며, 오버헤드를 줄일 수 있었다.

키워드 : 무선센서네트워크, 오버레이 네트워크, 미들웨어, 가상 대응체

A Sensor Overlay Network Providing Middleware Services on Wireless Sensor Networks

Yong-Pyo Kim[†] · Euihyun Jung[††] · Yong-Jin Park[†††]

ABSTRACT

A research for middleware of WSN can provide sensor applications with avoiding tight coupling of hardware, ease of development, and abstract data access. However, previous works have some limitations which should install their own middleware onto the all sensor nodes resulting in computational and communication overhead. In order to address it, we proposed a virtual sensor overlay network, called TinyONet-Lite which introduced virtual sensors to model a virtual counterpart of physical sensors. These virtual sensors dynamically grouped into an overlay network, Slice, which provides middleware services. We implemented TinyONet-Lite on mote class hardware with TinyOS. In accordance with experiments and comparison with existing researches, TinyONet-Lite was proved to show advantages of extensibility, dynamic service composition and reducing overhead.

Keywords : WSNs, Overlay Network, Middleware, Virtual Counterpart, TinyOS

1. 서 론

1.1 연구 배경 및 목적

무선 센서 네트워크는 기존 무선 애드혹(ad-hoc) 네트워크에서 해결하지 못한 응용범위 확장을 위한 중요한 기술로 고려되고 있다[1-3]. 무선 센서 네트워크는 전장, 교통, 환경 감시, 재난 감시등의 다양한 분야에서 응용되고 있다. 무선 센서 네트워크는 파워, 메모리, 통신 능력에 제한이 있는 수백~수천 개의 작은 센서들로 구성되어 있다. 최근까지, 이러한 제약 상황을 극복하면서 다양한 응용분야에 활용하기 위한 많은 연구들이 에너지 효율적인 관점에서 라우팅, OS의 소형화, 미들웨어를 중심으로 이루어지고 있다.

특히 미들웨어는 데이터 수집에만 초점을 맞추어 연구되었던 단순한 응용을 다양한 미들웨어 서비스를 이용하여 확장하고 하드웨어에 밀접하게 연관되어 있는 구현의 어려움을 해결하기 위한 방법으로 제안되었다. 기존의 미들웨어 연구는 가상머신 기반의 접근 방식과 데이터베이스 기반 접근방식으로 분류할 수 있다. 가상 머신 기반의 접근방식은 해석기(interpreter)와 실행코드로 구성되어 있다. 이진화 된 실행코드는 네트워크로 전파되어 센서 노드의 기능을 변경,

※ 이 논문은 2009년도 두뇌한국 21 사업과 2006년도 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단의 지원을 받아 연구되었음 (KRF-2006-003-D00365).
† 준 회 원 : 한양대학교 전자컴퓨터통신공학과 박사과정
†† 정 회 원 : 안양대학교 컴퓨터학과 조교수(교신저자)
††† 정 회 원 : 한양대학교 전자컴퓨터통신공학부 교수
논문접수 : 2008년 11월 3일
수 정 일 : 1차 2009년 7월 17일
심사완료 : 2009년 7월 20일

확장하는 것이 가능하다. 데이터베이스 기반의 접근방식은 질의(query)와 질의 해석기로 구성되어 속성기반의 데이터 질의를 센서 네트워크에서 데이터베이스와 같이 처리가 가능하다. 가상 머신 기반의 접근방식에서는 확장성과 재프로그래밍(Re-programming)의 장점을 제시했고, 데이터베이스 기반의 접근방식에서는 속성기반의 데이터 접근방식을 제시하여 데이터의 추상적인 처리가 가능함을 보였다[4-6].

이 두 가지 방식은 모두 실행코드 또는 쿼리를 해석하기 위한 해석기를 모든 센서 노드에 설치한다는 가정을 하고 있다. 기존의 방식은 모든 센서 노드가 네트워크로 전파되는 실행코드와 질의를 해석하고 이해하여 센서 네트워크의 확장성을 제공하고 고수준의 미들웨어 서비스를 제공하는데 큰 기여를 했다. 그러나 자원 제약이 심한 센서노드에 해석기를 설치하고 응용의 요청에 따라 전파되는 실행코드로 EEPROM을 자주 갱신하는 것은 추가적인 오버헤드를 발생시킨다. 또한, 실행 코드의 전파는 센싱 데이터를 위한 통신 이외에 추가적인 통신 오버헤드를 생성 시키고, 전용으로 사용하는 가상 머신에 대한 프로그래밍 기술의 이해 없이는 응용을 개발하기 어렵고 이기종의 센서네트워크와 호환될 수 없는 문제점을 노출하였다.

이러한 문제점들을 해결하기 위해서 본 논문에서는 센서 오버레이 네트워크(Sensor Overlay Network)를 이용한 TinyONet-Lite를 제안한다. TinyONet-Lite는 일반적으로 센서 노드보다 자원이 풍부한 싱크노드에 가상 센서로 구성된 오버레이 네트워크를 구성하여 미들웨어 서비스를 제공하도록 설계되었다. 가상 센서는 물리적 센서의 가상 대응체로써, 물리적 센서와 1:1로 연결되어 있으며 외부 응용들에게 미들웨어 서비스를 제공하기위해 가상 센서들이 동적으로 협력하여 오버레이 네트워크인 슬라이스(Slice)를 구성한다. 슬라이스는 가상 센서들로 구성된 네트워크상에 얇은 층(thin layer)처럼 동적으로 구성된 서비스 유닛 모델이다.

여러 가지 목적을 갖는 다양한 슬라이스들이 동시에 수행됨으로써, TinyONet-Lite는 무선 센서네트워크에 다수의 센서 응용에 의한 추상적인 데이터 접근을 동시에 가능하게 한다. 또한 싱크노드를 제외한 일반 센서에는 미들웨어의 설치를 하지 않으므로 오버헤드가 거의 없다. 그리고 추상적인 데이터 접근 방식을 이용하여 데이터 수집을 위한 응용 개발이 용이하고 하부 네트워크에 독립적인 구성이 가능하여 호환성이 뛰어나다. 또한 가상 센서의 기능을 손쉽게 확장할 수 있어 확장성도 매우 뛰어난 장점을 갖는다.

본 논문의 구성은 다음과 같다. 2장에서는 TinyONet-Lite의 디자인 구성과 구현에 대해 설명하고 슬라이스와 SOML(Slice Order Markup Language)에 대해 설명한다. 3장에서는 TinyONet-Lite을 평가하기 위하여 실험을 통해 기존의 미들웨어와 비교, 분석한다. 마지막으로 4장에서는 결론을 맺는다.

1.2 관련 연구

기존의 미들웨어 연구는 다양한 기법들을 제시하여 미들웨어 서비스를 제공하는데 큰 기여를 하였다. 이러한 연구는 크게 가상머신 기반과 데이터베이스 기반 접근 방식으로 나누어진다[1,4,5].

가상머신 기반 미들웨어는 가상 머신, 해석기, 그리고 모바일 코드로 구성되어져 있다. Mate[7]는 독자적인 바이트코드 해석기를 가지고 TinyOS에서 수행되고, 재프로그래밍 기법을 제공한다. 가상머신 언어로 짜인 최대 24개의 단위 명령으로 구성된 "캡슐" 이라고 부르는 프로그램이 노드위에 올려져 하나의 기능(Task)으로 수행하게 된다. 센서 네트워크에 새로운 기능이 요구되었을 때 새로운 기능을 담은 "캡슐"을 통해 기능을 확장할 수 있기 때문에 확장성이 매우 우수하다. 그러나 "캡슐"을 해석하기 위한 컴퓨팅 오버헤드가 발생하고 재프로그래밍에 의해 모바일 코드가 EEPROM에 쓰여지는 것을 막을 수 없다. 일반적으로 EEPROM에 쓰는 작업은 간단한 메모리 작업보다 더 많은 에너지를 사용하기 때문에 에너지의 소모가 커질 수 있다. 그리고 사용자들은 특정 가상 머신 프로그램언어와 구조에 대해서 지식을 가지고 있어야 하기 때문에 응용 개발이 어렵다.

데이터베이스 기반 미들웨어는 센서 네트워크를 가상 데이터베이스 시스템과 같이 사용되는 네트워크이다[8,9]. 센서노드에서 센싱된 데이터는 각 노드내에 저장되어 있고, 센서노드는 센서 응용으로부터 속성기반의 SQL과 유사한 형태를 갖는 질의를 받으면, 질의 해석을 통해 데이터를 전달한다[10]. Cougar와 SINA는 센서 네트워크를 하나의 분산된 데이터베이스로 정의하고, 쿼리 형식을 사용하여 센서 네트워크의 정보에 접근한다. Cougar에서는 데이터를 수집하고 질의 결과를 센서 노드에 분산시킴으로써 전력 소모를 관리한다. SINA에서는 각 센서가 센서노드의 속성을 데이터시트에 수집하여, 유지시켜놓는다. TinyDB[11]는 질의 처리 시스템으로써 SQL과 같은 인터페이스를 쉽게 사용할 수 있는 센서 노드들로부터 사용자가 관심 있는 데이터에 접근이 가능 할 수 있도록 한다. 다른 TinyOS기반의 솔루션들과 달리 TinyDB는 내장형 C 코드와 같은 프로그래밍언어를 사용해서 구현할 필요가 없다.

데이터베이스 기반의 미들웨어는 쉽게 이해할 수 있는 질의를 통해 센싱 데이터에 대한 추상화된 데이터 접근이 가능하다는 장점이 있다. 따라서 센서 구조나 특정 가상 머신에 대한 프로그래밍 기술 지식 없이도 쉽게 센싱 데이터에 접근할 수 있기 때문에 사용자가 센서 응용을 쉽게 개발 할 수 있다. 그러나 추상화된 질의가 네트워크로 모두 전파되어야 하기 때문에 통신 오버헤드가 발생하게 된다. 또한 센서 노드의 질의 해석기를 확장하기 어렵기 때문에 질의 확장 역시 어렵고 이기종의 센서 네트워크와 동시에 사용할 수 없는 호환성의 문제가 있다.

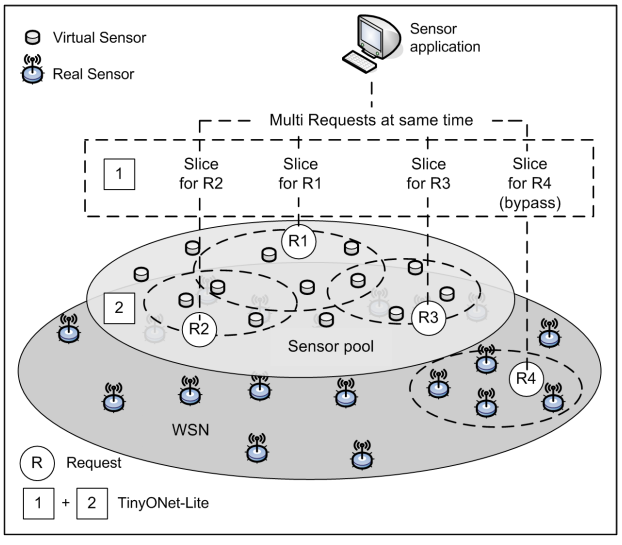
2. 설계 및 구조

2.1 시스템 설계

기존의 미들웨어 연구 분석을 통해 센서 네트워크의 미들

웨어가 갖추어야 할 몇 가지 요구사항을 도출하였다. 첫째, 통신과 컴퓨팅 오버헤드 둘 다 최소화해야 한다. 추상적인 데이터 접근을 위해서 질의 시스템을 채택하면 쿼리가 모든 노드로 전달되어야 하는 통신 오버헤드가 발생하고 가상 머신을 채택하면 가상 머신 내에서 코드 실행으로 인한 컴퓨팅 오버헤드가 야기된다. 둘째, 코드의 제한 없이 상황인지[12]에 기반을 두어 다양한 응용의 요구에 맞게 동적으로 서비스를 구성하고 제공할 수 있어야 한다. 셋째, 무선 센서 네트워크의 기능 확장을 위한 비용(cost)를 최소화해야 한다. 가상 머신 접근방식에서는 EEPROM쓰기와 모바일 코드의 전송을 통해 기능 확장이 용이하지만 데이터베이스 접근 방식에서는 질의해석기를 갱신하기는 매우 어렵다. 또한, 많은 EEPROM 쓰기와 빈번한 모바일 코드의 전송은 센서네트워크에 많은 오버헤드를 발생하게 된다. 넷째, 상호 운용성을 확보해야한다. 기존의 미들웨어들은 전용의 해석기를 설치하기 때문에 기존에 설치된 이기종의 센서 네트워크와는 같이 사용할 수 없다. 센서 네트워크는 많은 노드의 넓은 범위에 설치되기 때문에 이기종의 센서네트워크도 같이 운용될 수 있는 상호 운용성을 제공해야 한다. 마지막으로, 응용 개발이 쉬워야 한다. 센서 응용은 하드웨어와 밀접한 코드로 작성되기 때문에, 미들웨어를 통해 특정 OS나 프로그래밍 언어에 대한 지식 없어도 쉽게 개발할 수 있는 방법을 제공해야 한다[13].

일반적으로 센서 네트워크의 싱크 노드는 다른 센서 노드에 비해 파워, 메모리, 통신, 그리고 컴퓨팅 능력등의 자원이 풍부하다. TinyONet-Lite은 싱크 노드에 물리적 센서 노드와 1:1로 대응되는 가상 센서 네트워크를 구성하고 이를 동적으로 그룹화 하여 미들웨어 서비스를 제공하도록 설계되었다. 응용으로부터의 요청은 슬라이스라는 단위로 처리되며 여러 요청에 대한 슬라이스를 생성하여 동시에 처리가 가능하도록 구현되었다. (그림 1)은 TinyONet-Lite의 개념적 설계도이다.



(그림 1) TinyONet-Lite 개념적 설계
(Fig. 1) Conceptual design of TinyONet-Lite

(그림 1)에서의 각각의 물리적 센서들은 오버레이 네트워크상에 있는 가상 센서들과 1:1로 연결되어 있다. 가상 센서는 물리적 센서의 대응체로써, 물리적 센서로부터 받은 최신의 센싱 데이터를 유지한다. 센서 응용이 데이터 요청을 하면, TinyONet-Lite는 동적으로 가상 센서들로 그룹 지어진 슬라이스를 구성한다. 슬라이스는 응용의 요청을 처리하기 위해 검색 조건에 적합한 가상 센서들로 구성된 오버레이 네트워크이다. 이러한 구조는 센서 노드의 동일한 데이터를 다양한 응용에서 여러 가지 목적으로 동시에 사용할 수 있는 미들웨어 서비스를 제공한다. 예를 들어 같은 건물에서 수집한 센싱 데이터는 와인 보존 슬라이스나 비상사태 관리 슬라이스 둘 다에서 사용될 수 있다. 또 다른 예로는, 한 방에 있는 여러 사용자들이 가상센서를 이용해서 각 사용자들의 목적에 적합한 개인화된 슬라이스를 가질 수 있다.

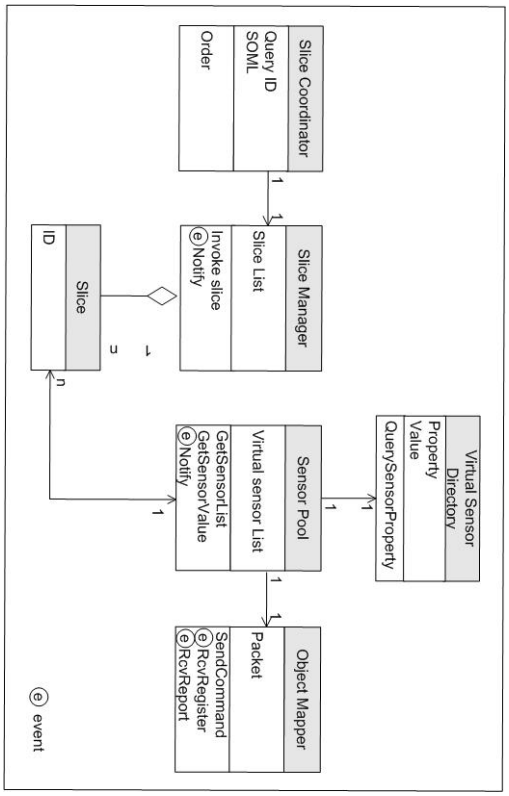
제안한 미들웨어는 몇 가지 장점을 가지고 있다. 첫째, 모든 센서 노드에 특정 미들웨어를 설치할 필요 없이 외부 응용에 미들웨어 서비스를 제공할 수 있다. 둘째, 동일한 센싱 데이터로부터 여러 목적으로 다양한 정보를 동적으로 생성이 가능하기 때문에 동적인 서비스를 쉽게 구성할 수 있다. 마지막으로 이기종의 센서 노드, 응용에 대해 호환성을 제공할 수 있다. 다른 종류의 응용과 센서 네트워크를 위해 오버레이 네트워크를 그냥 통과시키는 방법을 제공하여 전용의 데이터 패킷이 아닌 경우도 제안한 미들웨어를 같이 사용할 수 있다.

2.2 구현 및 운용

2.2.1 가상 센서 오버레이 네트워크

이 논문에서는 (그림 2)처럼 nesC로 구현하여 TinyOS 컴포넌트를 구성했다. Object Mapper, TxRx, Virtual Sensor Directory, Sensor Pool, Slice Coordinator 그리고 Slice Manager는 오버레이 네트워크의 미들웨어 서비스를 제공하기 위해 서로 협력한다.

Object Mapper는 물리적 센서 노드들을 가상 센서들에 연결시켜주는 역할을 한다. 그리고 TxRx와 통신을 하고, 받은 데이터를 Sensor Pool로 전달한다. Sensor Pool에서 데이터 저장은 EEPROM을 사용한다. 그리고 슬라이스를 구성하고 데이터를 처리하기 위해 물리적인 센서 노드와 매핑된 센서 노드 객체의 정보 및 데이터를 처리하는 기본 구조를 가지고 있다. TinyONet-Lite에서 새로운 물리적 센서가 스스로를 등록 할 때, Sensor Pool은 Virtual Sensor Directory에서 해당되는 속성 정보를 읽고, EEPROM에 등록된 센서를 속성테이블에 추가한 후에 가상 센서를 인스턴스화 한다. 물리적 센서에 보고된 데이터는 대응되는 가상 센서 속성테이블에 저장되고, Slice Manager로부터 전달된 조건에 해당되는 센서 데이터 또는 센서 노드 목록을 해당 슬라이스로 전달한다. Virtual Sensor Directory는 각 센서들의 메타 속성정보를 관리한다. 만약 새로운 타입의 센서가 센싱 필드에 추가됐다면, 관리자는 Virtual Sensor Directory에



(그림 2) TinyONet-Lite 클래스
(Fig. 2) Class diagram of TinyONet-Lite

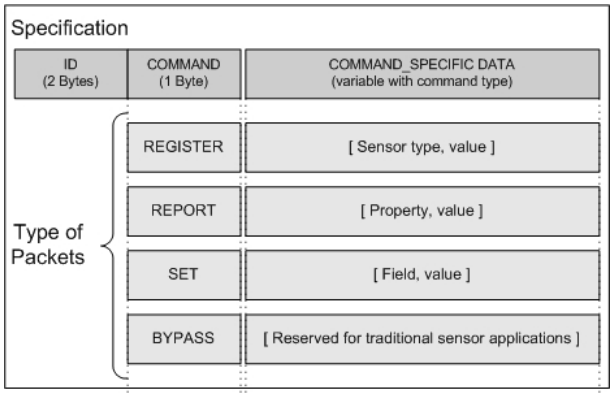
센서 속성정보를 추가한다. Slice Coordinator는 외부 응용으로부터 전달된 요청을 해석하여 Slice Manager에게 조건, 필터 리스트, 질의 ID등을 포함하여 전달한다. Slice Manager는 해당 요청을 슬라이스로 생성하여 Sensor Pool로 질의를 전달한다. 이때 Slice Manager는 슬라이스들을 관리한다.

2.2.2 데이터 패킷

TinyONet-Lite에서 사용되어지는 데이터 패킷은 (그림 3)과 같다.

각 패킷은 센서 노드의 ID, COMMAND, 그리고 특정 명령 전용 데이터(Command specific data)를 갖는다. ID 필드는 각 물리적 센서 노드를 식별하기 위해 두 바이트의 식별자를 갖는다. COMMAND 필드는 한 바이트로 패킷 타입을 의미한다. 패킷 타입은 REGISTER와 REPORT, SET, 그리고 BYPASS의 4종류가 있다. 특정 명령 전용 데이터(Command specific data) 필드의 길이와 내용은 명령(Command)에 따라 다양하다.

센서 노드가 센싱 필드에 새롭게 배포되면, TinyONet-Lite의 Object Mapper에 REGISTER 데이터 패킷을 보낸다. 이 명령은 Sensor Pool로 전달되어 Sensor Pool에서 가상센서를 인스턴스화 시키는데 사용된다. REPORT 데이터 패킷은 센서가 보고할 새로운 데이터가 센싱되어, 이를 전송할 때 사용된다. 보고된 센싱 데이터는 수신된 시간정보



(그림 3) 데이터 패킷 구조
(Fig. 3) Structure of data packet

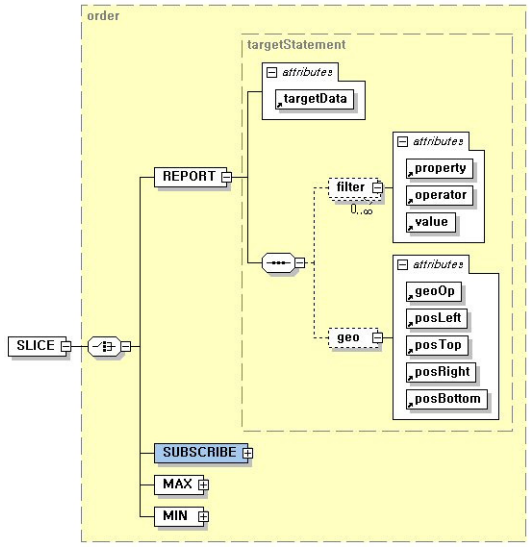
(Time stamp)를 포함하여 가상센서의 속성 테이블에 저장되고 슬라이스의 질의 조건 확인에 사용된다. SET 데이터 패킷은 센서노드의 ID나 위치정보와 같은 센서 속성 정보를 갱신할 때 사용된다. 또한, 네트워크에 있는 이기종의 센서 노드에게 호환성을 제공하려 할 때에 BYPASS 데이터 패킷을 사용할 수 있다. 이 경우 센싱 데이터는 TinyONet-Lite에서 처리되지 않고 바로 센서 응용으로 전달된다.

2.2.3 Slice Order Markup Language

TinyONet-Lite의 슬라이스 처리를 위해 이 논문에서는 Slice Order Markup Language(SOML)을 설계하여 응용의 질의를 전달하는 언어로 사용하였다. SOML은 TinyONet-Lite에서 사용할 수 있게 XML을 변형하여 센서의 속성을 표현하기 적합하도록 생략된 버전으로 설계한 마크업(markup)언어이다.

슬라이스 요소(Element) 내부를 보면, 네 가지 서브 명령 속성(Sub element)이 있다. REPORT, SUBSCRIBE, MAX 그리고 MIN이다. 이 명령은 Slice Manager가 슬라이스 처리를 위해 선택할 수 있다. 예를 들어 만약 REPORT 명령이 SOML 문서에 쓰였다면, REPORT 슬라이스를 구성하기 위해서 Slice Manager는 REPORT 동작을 처리한다. REPORT 명령은 주어진 상태에서 만족스러운 센싱 데이터를 얻기 위해 동작한다. MAX, MIN명령은 센싱된 데이터 각각의 최대값과 최소값을 얻기 위해 추가적으로 리포팅된 서비스를 제공해 주는 좀 더 복잡한 작업을 한다. 반면, SUBSCRIBE 명령은 주어진 상태를 만날 때 까지 기다려서 슬라이스를 만든다. (그림 4)는 본 논문에서 사용한 SOML의 XML 스키마(Schema)이다.

각 명령 속성은 타겟 데이터 속성과 filter 와 geo의 두 개의 서브 속성을 가진다. 타겟 데이터 속성은 명령내린 슬라이스에 관련된 데이터를 말한다. filter는 property, operation 그리고 value 세 가지 속성을 가진다. property는 온도나 습도와 같은 것을 표현한다. operation 속성은 GT(>), GE(>=), EQ(==), NE(!=), LE(<=), LT(<) 이다. value 속성은 필터링한 property의 타겟 값이다. 예를 들어,



(그림 4) SOML 스키마
(Fig. 4) Schema of SOML

만약 센서의 센싱 온도가 30도 이상인 센싱 데이터를 찾는다면, SOML은 다음과 같이 작성된다.

```
<filter property="temperature"
operator="GT" value="30"/>
```

geo 속성은 센싱 데이터의 특정 위치 검색에 사용된다. geoOp, posLeft, posTop, posRight, 그리고 posBottom 다섯 가지의 속성으로 이루어져 있다. geoOp는 지역을 찾기위한 범위(Range)에 대한 INSIDE와 OUTSIDE값을 의미한다. 만약 센서 응용이 질의한 데이터의 geo속성이 left=20, top=20, right=50, bottom=100이고 이 범위에서 INSIDE의 데이터를 얻기를 원할 때, SOML은 다음과 같이 표현된다.

```
<geo geoOp="INSIDE",
posLeft="20", posTop="20"
posRight="50", posBottom="100"/>
```

2.2.4 슬라이스 처리

TinyONet-Lite에서는 외부 응용이 요청한 서비스에 대해 물리적인 센서 네트워크와 1:1로 대응된 가상센서들을 동적으로 그룹화하여 논리적인 오버레이 네트워크를 구성하는데 이를 슬라이스라 명한다. 즉, 외부 응용으로부터 서비스 요청이 들어오면 독립적인 각각의 개별 서비스마다 당 property와 서비스 조건에 해당하는 가상 센서 목록을 구성하고 이것을 바탕으로 데이터를 처리하게 된다. 예를 들어, 아래와 같이 left=20, top=30, right=40, bottom=80의 사각형 지역 내에 위치한, 온도 20도 이하인 가상 센서 목록 중에 최대 습도 데이터를 원하는 SOML을 처리하는 경우 Slice Manager가 Max 값을 얻는 슬라이스를 선택하여 이를 처리하도록 한다.

```
<SLICE>
<MAX targetData="humidity">
<filter property="temperature" operator="LT"
value="20"/>
<geo geoOp="INSIDE"
posLeft="20" posTop="30"
posRight="40" posBottom="80"/>
</MAX>
</SLICE>
```

다른 명령 속성과 다르게 SUBSCRIBE 명령은 질의 기반의 데이터 처리가 아닌 이벤트 기반의 데이터 처리가 필요할때 센싱데이터를 센서 응용에 비동기적으로 통보하기 위해 사용되어진다. 만약 센서 응용이 left=20, top=40, right=50, bottom=80 인 위치에 있는 센서 중에 온도가 20도 이상이고, 습도는 30% 이상인 조건에 일치할때만 이벤트로 알려주기를 원한다면, SOML은 다음과 같이 표현된다.

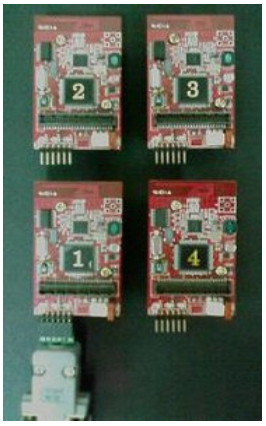
```
<SLICE>
<SUBSCRIBE targetData="humidity">
<filter property="temperature" operator="GT"
value="20"/>
<filter="humisity" operator="LT" value="30"/>
<geo geoOp="INSIDE"
posLeft="20" posTop="40"
posRight="50" posBottom="80"/>
</SUBSCRIBE>
</SLICE>
```

3. 실험 및 평가

제안된 구조는 ATMega 128 CPU와 TinyOS를 사용한 모트(mote)기반의 하드웨어 (한백전자:zigbeX[14]) 환경에서 NesC를 사용하여 구현되었다. (그림 5)는 실제 구현된 모트의 하드웨어이다.

실험에 사용된 모트는 온도, 습도, 조도를 센싱 할 수 있다. 싱크 노드 역할을 하는 하나의 모트에는 TinyONet-Lite가 설치되어 있고 싱크 노드를 제외하고 나머지 모트들은 센싱 데이터를 알려주는 간단한 센서 응용을 설치하였다. TinyONet-Lite 를 효과적으로 실험하기 위해서 세 개의 방에 센서 노드용 모트를 배치하였고 동적 서비스 기능, 확장성 측면에서 두 가지 실험을 수행하였다.

실험에 사용된 모트는 용도가 다른 세 개의 방에 용도에 적합하게 설치되어 있다. 첫 번째 방은, 네 개의 모트와 RFID 리더기가 설치되어있다. 그 방에 있는 두 사람은 개인화 된 RFID를 가지고 있다. 개인화 된 RFID에는 한 사람은 고혈압을 가지고 있는 사람의 데이터를 저장하였고 다른 사람은 건강한 사람의 데이터를 저장하였다. 두 번째 방은 두 개의 모트가 설치되었고 화재위험에 노출될 가능성이 높은 창고로 가정하였다. 마지막 방은 기존의 방식을 사용하는



(그림 5) 실험에 사용된 모트
(Fig. 5) Mote in the experiment

두 개의 전통적인 모트를 설치하여 TinyONet-Lite에서의 호환성을 테스트 하였다.

3.1 동적 서비스 구성

이 실험은 TinyONet-Lite에서 동적인 서비스 구성을 검증하기 위한 실험으로 첫 번째 방에서 고혈압 환자와 건강한 사람에 대한 정보를 이용하여 실험하였다. 첫 번째 슬라이스는 고혈압 환자를 모니터링 하는 슬라이스로 SUBSCRIBE 명령을 이용한 실험이다. 고혈압 환자를 위한 슬라이스에서는 온도가 22도 이상일 때 습도 데이터를 확인하여 경고 이벤트를 알려주도록 하였다. 반면 건강한 사람의 슬라이스는 일반적인 냉난방을 위한 동작을 하도록 하는 기능으로 온도가 25도 이상일 때 습도 데이터를 확인하고 알림 이벤트가 동작되게 하였다. 또한 화재위험에 노출될 가능성이 높은 창고에서는 온도가 80도 이상 상승했을 때 습도 데이터를 가지고 화재 조건을 판단하여 화재 알람을 알려주도록 하였다. 각각의 슬라이스를 설정하기 위한 SOML은 다음과 같다.

이 실험을 통하여 다양한 슬라이스들이 외부 사용자 의 제어 없이 동적으로 생성되어 동시에 수행 가능함을 입증하였다. 또한, 같은 센싱 데이터를 다른 목적으로 설정된 슬라이스에서 공유하며 동적으로 고수준의 정보제공이 가능함을 확인하였다.

```
<SUBSCRIBE targetData="humidity">
<filter property="temperature" operator="GT" value="22"/>
<geo geoOp="INSIDE"
posLeft="0" posTop="0"
posRight="340" posBottom="835"/>
```

〈고혈압 환자의 슬라이스〉

```
<SUBSCRIBE targetData="humidity">
<filter property="temperature" operator="GT" value="25"/>
<geo geoOp="INSIDE"
posLeft="0" posTop="0"
posRight="340" posBottom="835"/>
```

〈건강한 사람의 슬라이스〉

```
<SUBSCRIBE targetData="humidity">
<filter property="temperature" operator="GT" value="80"/>
<geo geoOp="INSIDE"
posLeft="0" posTop="0"
posRight="700" posBottom="835"/>
```

〈화재 위험의 슬라이스〉

3.2 확장성

TinyONet-Lite의 쉬운 확장 가능성을 보여주기 위해서 이 논문에서는 시스템에 AVERAGE 명령을 추가하였다. AVERAGE 명령을 수행하기 위한 SOML을 작성하고 TinyONet-Lite에서 이해하도록 수정하였다. 이 실험에 사용된 SOML은 다음과 같다.

```
<SLICE>
<AVERAGE targetData="temperature">
<filter property="humidity" operator="LT"
value="20"/>
<geo geoOp="INSIDE"
posLeft="20" posTop="30"
posRight="40" posBottom="80"/>
</AVERAGE>
</SLICE>
```

〈AVERAGE 슬라이스〉

이 실험을 위해 Slice Coordinator와 Slice Manager에는 AVERAGE 명령을 해석하기 위한 코드를 추가하였다. 두 번째 방에 AVERAGE 명령을 이용하기 위한 SOML을 적용하였을때 Sensor Pool에서 수집된 데이터를 이용하여 슬라이스에서 센서의 습도값 평균을 조사하기 위해 left=20, top=30, right=40, bottom=80의 지역 내에 있는 센서들 중에 습도가 20%이내인 센서들만의 온도 데이터에 대한 평균값을 구하는 SOML을 실험하였다. 이 실험을 통하여 제안 시스템은 다른 부분에 영향을 주지 않고 확장이 용이함을 증명하였다.

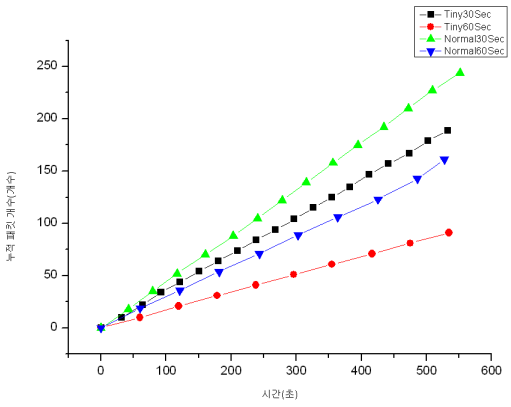
3.3 제안 미들웨어의 정량적 실험

본 논문에서 제안한 미들웨어가 갖는 동적 서비스 구성과 확장성의 장점에 대해서는 앞서 기술한 실험을 통하여 확인할 수 있었다. 그러나 앞선 2 가지의 실험만으로는 제안 미들웨어를 사용하는 시스템과 미들웨어를 사용하지 않는 기존의 센서 네트워크 시스템과의 정량적인 비교를 할 수 없다. 따라서 미들웨어를 사용하지 않는 기존의 시스템과 제안 미들웨어를 사용하는 시스템의 수치화 된 비교를 위해 다음과 같은 추가적인 실험을 수행하였다. 실험은 기존 실험과 같이 3 개의 방에 각각 3 개씩 총 9 개의 센서노드를 설치하였고 제안 미들웨어를 사용하는 시스템과 미들웨어를 사용하지 않는 센서 네트워크 시스템을 구성하였고, 통신은 ZigBee 프로토콜을 이용하였다. 이 실험에서는 응용 프로그램으로부터 센서 네트워크로 데이터를 요청하는 명령을 실행하였을 때 두 개의 다른 구조를 갖는 시스템에서의 전송 패킷 개수와

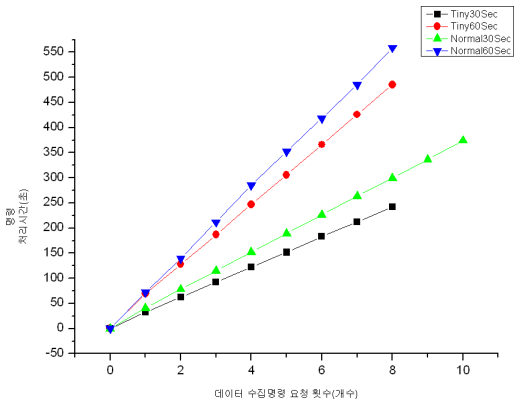
데이터 응답 처리 시간을 측정하여 비교하도록 구성하였다.

(그림 6)은 응용 프로그램에서 센서 네트워크로 데이터를 요청하는 명령을 실행했을 때 이 명령을 처리하기 위한 패킷의 수를 비교한 그래프이다. 제안 미들웨어를 사용한 시스템과 미들웨어를 사용하지 않은 시스템 모두 30초와 60초 간격으로 데이터 요청을 했을 때의 데이터 요청 명령을 처리하기 위해 송수신하는 패킷의 개수를 보면 미들웨어를 사용하지 않은 Normal 60, Tiny 60, Normal 30과 Tiny 30의 순서이다. 기본적으로 TinyONet-Lite는 센서 노드의 데이터 리포트 주기를 기준으로 항상 싱크 노드로 데이터를 전송하고 있기 때문에 응용 프로그램으로부터 데이터 요청 시 싱크노드의 가상 센서에 데이터가 캐싱 되어 있으면 바로 응답이 가능하다. 반면에 미들웨어를 사용하지 않는 Normal 시스템의 경우는 응용 프로그램이 데이터를 요청할 때 각각의 센서 노드로 데이터 요청 패킷을 순차적으로 처리하기 때문에 데이터 요청 명령을 처리하기 위한 패킷의 개수가 많이 필요하게 된다.

또한, (그림 7)에서 보면, 응용프로그램에서 데이터 요청 명령이 센서 네트워크로 명령이 실행된 후 이에 대한 응답이 완료되는 시간을 비교하였다. 데이터 요청 명령 처리 시간 역시 Normal 60, Tiny 60, Normal 30 과 Tiny 30의 순서로 동일한 리포트 주기에 대해 TinyONet-Lite를 사용한 시스템이 더 짧은 명령 처리 시간을 보이고 있다. 이는 TinyONet-Lite를 미들웨어로 사용하는 시스템의 경우 싱크 노드내에 탑재되어 있는 TinyONet-Lite에 센서 데이터들이 캐싱되어 있기 때문에 응용 프로그램으로부터 데이터 요청 명령이 내려오는 시점에서 캐싱 되어 있는 데이터를 이용해 바로 응답할 수 있는 장점이 생긴다. 또한, 데이터 요청 명령이 전달되는 시점에 캐싱 되어 있는 데이터가 최신 데이터가 아닌 경우는 해당 센서 노드에 새로운 데이터를 요청하도록 하여, 항상 새로운 데이터를 사용하도록 하였다. 반면에 Normal 시스템의 경우는 응용 프로그램으로부터 데이터 요청 명령이 실행되는 순간부터 순차적으로 모든 센서 노드에 대해 데이터 요청 명령을 실행하기 때문에 명령 처리 시간이 상대적으로 길어지게 된다.



(그림 6) 데이터 요청 명령에 대한 패킷 수
(Fig. 6) Packet counts for data request



(그림 7) 데이터 요청 명령 처리시간
(Fig. 7) Response time for data request

3.4 기존 미들웨어와 비교

<표 1>은 TinyONet-Lite와 기존의 미들웨어를 실험 결과를 바탕으로 다양한 측면에서 비교한 표이다. 표에서 볼 수 있듯이, TinyONet-Lite는 컴퓨팅 오버헤드, 통신 오버헤드 확장성, 응용 개발의 용이성, 그리고 동적 서비스 제공 등의 측면에서 기존의 미들웨어에 비해 우수하다. 그러나 기존의 미들웨어에 비해 제안 시스템은 싱크 노드에 설치되기 때문에 싱크 노드에서 통신이 단절되었을 때 단일 노드 장애 문제가 발생할 수 있다. 이는 추가 비용을 감안하여 복수의 싱크 노드를 설치하여 해결할 수 있다.

〈표 1〉 기존 미들웨어와 비교
〈Table 1〉 Comparison with existing middleware approaches

	TinyONet-Lite	가상머신 기반의 접근방법	데이터베이스 기반의 접근방법
통신 오버헤드	○	◎	△
코드 처리 오버헤드	◎	△	○
확장성	◎	○	X
응용 개발의 용이성	◎	△	◎
동적 서비스 제공	◎	○	◎

◎: 탁월, ○: 우수, △: 보통, X: 불량

4. 결 론

무선 센서 네트워크에서 미들웨어 연구는 지속적으로 연구자들에게 주목을 받고 있다. 미들웨어는 물리적으로 센서 노드 하드웨어와 응용 서비스 사이에 존재하면서, 센서 하부구조와의 연관을 최소화하고, 추상적인 데이터 접근을 제공해 준다. 그러나 리소스 제한이 많은 센서 노드에 오버헤드가 심한 미들웨어 스택을 설치해야 하는 약점을 갖고 있다. 이를 위해 본 논문에서는 센서 노드들에 추가적인 컴포넌트 설치 없이 추상적인 데이터 접근과 미들웨어 서비스를 제공할 수 있는 센서 오버레이 네트워크를 기반으로 한 TinyONet-Lite을 제안하였다.

TinyONet-Lite의 가상 센서는 물리적 센서의 가상 대응체로 동작한다. 가상 센서들은 1:1로 대응되는 물리적 센서

로부터 받은 데이터를 유지하고, 가상 센서들의 그룹을 생성하여 슬라이스를 동적으로 구성하고 미들웨어 서비스를 제공한다. 제안 시스템은 슬라이스 개념을 통해 센서들의 컴퓨팅, 통신 오버헤드를 줄이고, 확장성, 호환성을 제공하며 응용 개발을 쉽게 할 수 있는 미들웨어 서비스를 제공하여 기존의 연구보다 우수함을 보였다. 향후 싱크 노드에서만 실행 코드가 운용되는 미들웨어를 추가적으로 연구할 계획이다.

참 고 문 헌

[1] I. F. Akyldiz, W. Su, Y. Sankarasubramaniam, and E, Cayirci, "A survey on sensor networks," IEEE Comm.magazine, Vol.40, No.8, pp.102-114, Aug., 2002.

[2] C. Chong and S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," Proc.of IEEE, Vol.91, No.8, pp.1247-1256, Aug., 2003.

[3] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," IEEE Computer, Vol.37, No.8, pp.41-49, Aug., 2004.

[4] K. Römer, O. Kasten, and F. Mattern, "Middleware challenges for wireless sensor networks," ACM SIGMBOBILE Mobile Communication and Communication Review, Vol.6, No.4, pp.59-61, Oct., 2002.

[5] S. Hadim and N. Mohmed, "Middleware: middleware challenges and approaches for wireless sensor networks," IEEE Distributed Systems Online, Vol.7, No.3, pp.1, Mar., 2006.

[6] 최용식, 김성선, 신승호, "유비쿼터스 환경에서 센서 노드의 관리와 망 구성을 위한 RFID 미들웨어 프로토콜에 관한 연구," 한국컴퓨터정보학회논문지, 12권, 3호, pp.155-163, 2007년 7월.

[7] P. Levis and D. Culler, "Mate: A tiny virtual machine for sensor networks," Proc. 10th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), ACM Press, pp.85 - 95, 2002.

[8] E. Souto et al., "A message-oriented middleware for sensor networks," Proc. 2nd International Workshop Middleware for Pervasive and Ad-Hoc Computing (MPAC 04), ACM Press, Vol. 77, pp.127 - 134, 2004.

[9] W. B. Heinzelman et al., "Middleware to support sensor network applications," IEEE Network, Vol.18, No.1, pp.6 - 14, 2004.

[10] 김정이, 이강호, "다중 질의 색인기법과 무선 센서를 이용한 환경정보 모니터링 시스템 구현," 한국컴퓨터정보학회논문지, 12권, 6호, pp.307-312, 2007년 12월.

[11] S.R. Madden, M.J. Franklin, and J.M. Hellerstein, "TinyDB: An acquisitional query processing system for sensor networks," ACM Transactions on Database Systems (TODS), Vol.30, No.1, pp.122-173, Mar. 2005.

[12] 최종화, 최순용, 신동규, 신동일, "지능적인 홈을 위한 상황인식 미들웨어에 대한 연구," 한국정보처리학회논문지, Vol.11A, No.7, pp.529-536, 2004년 12월.

[13] 김정길, 이준환, 박경랑, 김신덕, "통합 RFID 미들웨어의 응답 시간 개선을 위한 효과적인 캐쉬 구조 설계," 한국정보처리학회논문지, Vol.15A, No.1, pp.17-26, 2008년 2월.

[14] Product Information of ZigbeX Ubiquitous Sensor Networks, (http://www.hanback.co.kr/english/sub1.htm)

김 용 표

e-mail : ypkim@hyuee.hanyang.ac.kr
1998년 한양대학교 전자공학과(공학사)
2007년 한양대학교 전자컴퓨터통신공학과 (공학석사)
2007년~현 재 한양대학교 전자컴퓨터통신공학과 박사과정
2001년~2005년 스마트카드테크놀로지 연구원
관심분야 : WSN, 미들웨어, 무선 네트워크, 미래인터넷

정 의 현

e-mail : jung@anyang.ac.kr
1992년 한양대학교 전자공학과(공학사)
1994년 한양대학교 대학원 전자공학과 (공학석사)
1999년 한양대학교 대학원 전자공학과 (공학박사)
1999년~2002년 대우통신 선임연구원
2003년~2003년 가톨릭대학교 컴퓨터학부 초빙교수
2004년~현 재 안양대학교 컴퓨터학과 조교수
관심분야 : WSN, 시맨틱 웹, 데이터마이닝, 미래 인터넷

박 용 진

e-mail : park@hyuee.hanyang. ac.kr
1969년 와세다대학교 전자통신공학과(공학사)
1971년 와세다대학교 전자통신공학과(공학석사)
1978년 와세다대학교 전자통신공학과(공학박사)
1979년~현 재 한양대학교 전자컴퓨터통신공학부 교수
2001년~현 재 와세다대학교 객원 교수
2003년 한국정보과학회 회장
2009년~현 재 IEEE 본부 이사, Region 10 (아시아·태평양 지역) 회장
관심분야 : 컴퓨터통신, 이동 데이터 통신