논문 2012-07-39

이기종 센서 네트워크를 위한 데이터 중심적 센서 미들웨어

(Data-centric Sensor Middleware for Heterogeneous Sensor Networks)

남 춘 성, 신 동 렬*
(Choon-Sung Nam, Dong-Ryeol Shin)

Abstract : Wireless sensor networks need middleware system for efficiently managing the constrained resource and sensing data because they need different sensing data type and protocol to communicate with heterogeneous sensor networks. Thus this paper proposes data-centric sensor middleware for heterogeneous sensor networks. The proposed middleware have to support various query processing of user applications, high-level request of users, manage heterogeneous sensor systems and universal sensing data type for node and user application.

Keywords : Sensor networks, Data-centric, Heterogeneous sensor, Middleware

Ⅰ. 서론

센서 네트워크는 접근하기 용이하지 못하는 지역에 특정한 목적을 가진 매우 작은 센서 노드들이 환경적인 데이터를 수집하기 위해서 뿌려져 자가 구성으로 이루어진 네트워크이다. 이렇게 작은 센서 노드들은 제한된 자원(cpu 성능, 메모리, 무선 통신 범위 및 대역)을 가지기 때문에 에너지를 유지하기가 주요한 이슈이다[1]. 이러한 센서 노드들은 제한적인 자원으로 인하여 매우 간단한 기능만을 제공한다. 따라서 특정한 지역으로 값을 전달하거나 복잡한 high-level의 센싱 작업을 해결하기 위해서, 제한된 센서 노드들은 센싱 작업을 분배하고 공유할 필요가 있다. 이를 위해 센싱, 센싱 데이터 처리(필터링 및 요약) 그리고 이벤트 처리 등을 수행할 수 있는 미들웨어가 필요하다. 이러한 센서 네트워크 미들웨어의 주요 목적은 개발, 유지, 확장, 센싱 기반의 어플리케이션의 실행을 지원하는 것이다. 이러한 미들웨어는 복잡한 high-level 센싱 작업을 계산하거나, WSN에서 이러한 작업을 위해 통신하기, 각 센서 노드들로 일을 분배하기, high-level 결과를 도출하기 위해 각 센서 노드들의 센서 리딩 값을 종합하기 등이 있다. 또한 상이한 노드들을 다룰 수 있는 적당한 추상화 메커니즘을 제공해야만 한다[2]. WSN을 위한 미들웨어의 범위는 센서 노드만으로 제한하는 것이 아니라, WSN과 연관된 디바이스나 혹은 네트워크를 포함할 수 있다. 센서 네트워크의 데이터 중심적 통신 방식은 RPC(Remote procedure call) 방식의 통신보다는 컨텐츠(Contents) 기반 메시징 시스템과 같은 통신 패러다임(Paradigm)을 가진다. 게다가 이벤트 기반 통신은 전통적인 request-replay 스키마보다 자원제약적인 WSN의 특징과 부합한다. 따라서 일반적으로 통신과 어플리케이션에 특정화된 데이터 프로세싱이 전통적인 시스템보다 WSN 미들웨어에 더 잘 통합될 수 있다[3].

센서 네트워크의 미들웨어는 센서 네트워크 어플리케이션을 지원하고 조직화할 수 있는 필요성을 가진다. 이를 위해선 다음과 같은 여러 가지 디자인 원칙들을 지원해야만 한다. 첫째, 다양한 어플리케이션에 대한 표준화된 서비스를 제공해야만 한다. 둘째, 여러 가지 어플리케이션을 지원하고 조직화하기 위한 실행(runtime) 환경을 제공해야한다. 셋째,

* Corresponding Author (drshin@skku.edu)
Received: 03 Aug. 2012, Revised: 07 Sep. 2012, Accepted: 04 Oct. 2012.
C.S. Nam, D.R. Shin: Sungkyunkwan Univ.

ISSN : 1975-5066

효율적이고 적응적으로 시스템의 자원을 활용할 수 있는 메커니즘을 제공해야 한다. 특히 이러한 기능을 가진 미들웨어는 WSN에서 매우 유용하다. 이유는 WSN은 매우 많은 양의 정보 처리와 엄격한 실행 제약이 있는 네트워크이기 때문에 제약적인 에너지와 프로세싱 자원을 가지는 WSN은 단순하고 쉽게 구현될 수 있는 경량의(lightweight) 디자인이 요구된다[4]. 이러한 원칙을 기준으로 센서 네트워크 디자인 원칙을 세우면, 첫째, 센서 미들웨어는 데이터 프로세싱과 네트워크 내에서 쿼리를 가지는 데이터 중심적인 메커니즘을 제공해야만 한다. 이는 되도록 센서 네트워크의 어플리케이션의 적용을 위해 단순하고 확장할 수 있고 강건함을 유지해야만 한다. 둘째, 어플리케이션의 특성에 맞는 소프트웨어의 디자인과 구현을 해야 한다. 이는 미들웨어에서 제공하는 서비스로 어플리케이션을 통합하는 것이 중요하다. 셋째, 지역적 알고리듬(Algorithm)은 확장성과 강건함을 제공하는데 필수 요소가 되어야 한다. 넷째, 센서 노드의 가능한 자원이 매우 낮기 때문에, 미들웨어자체는 통신과 계산 요구에 맞게 경량화 시켜야 한다. 경량화 요구는 단순하고, 효율적이어야 한다. 마지막으로 제한된 자원 때문에, 실행하는 어플리케이션의 요구는 동시에 만족될 수 없다. 따라서 각 어플리케이션에 대해서 다양한 어플리케이션에 대한 QoS를 충족시킬 수 있는 미들웨어가 요구된다[4].

WSN를 위한 미들웨어의 범위는 센서 노드만으로 제한된 것이 아니라, WSN 과 연관된 디바이스나 네트워크를 포함한다. 어플리케이션을 효율적으로 지원하기 위하여 서버 시스템에 위치하는 Server-side 미들웨어와 센서 네트워크 내부에서 센싱 정보를 효과적으로 관리하기 위하여 센서 노드 및 싱크 노드들에 위치하는 In-network 미들웨어로 구성된다. Server-side 미들웨어는 센서 네트워크로부터 수집한 센싱 데이터를 필터링하고, 그 데이터를 통합, 분석하여 의미 있는 상황 정보를 추출, 저장, 관리, 검색하는 기능을 외부 네트워크에 연결된 응용 서비스에게 제공한다. In-network 미들웨어는 센서 노드와 센서 네트워크의 운영과 관리, 서비스 구현을 용이하고, 환경 변화에 따른 센서 네트워크의 topology 지원, 센싱 데이터의 처리, 저장, 관리, 질의, 이벤트 처리 기능을 제공한다[5, 6]. 따라서 센서 네트워크의 미들웨어를 구성하기 위해서 Server-side 미들웨어와 In-network 미들웨어를 동시에 구현해야 한다. 또한, 이종 간 센서 네트워크를 이용하기 위해서는 표준화된 데이터 형식의 센서 데이터를 취합하고, 사용자 별 요구사항 처리를 담당할 수 있는 이기종 센서 네트워크를 통합관리 미들웨어가 필요하다. 이에 본 논문은 센서 네트워크 미들웨어의 특징과 다양한 센서 네트워크로부터 수집된 데이터를 이용하여 환경 변화에 따른 상황 및 대처를 위한 통합된 데이터 중심적 미들웨어를 제안한다.

이 논문은 2장에서는 각 미들웨어의 기능 및 특징 그리고 문제점에 대해서 알아보고, 3장에서는 제안하는 미들웨어의 모델 및 구현 방법에 대해서 알아본다. 4장에서는 구현된 미들웨어를 기반으로 실제 생활에서 활용되는 다양한 요구사항을 위한 사용자 어플리케이션을 구현한다. 5장에서는 논문에 대한 결론을 맺는다.

Ⅱ. 관련연구

센서 네트워크는 제한적인 능력으로 인하여 지속적인 데이터 전달보다는 이벤트 중심적인 전송방식에 장점을 가진다. 또한, 사용자 어플리케이션과 센서 노드 어플리케이션을 위한 각각의 미들웨어가 존재함으로, 이를 지원하기 위한 방법들이 연구되어 왔다. TinyDB[7]는 TinyOS[8] 기반 쿼리 프로세싱을 위해 센서 네트워크를 가상의 분산 데이터베이스로 간주하는 미들웨어 시스템이다. 기본적으로 TinyDB는 전송될 메시지 수의 감소를 목표로 데이터 수집을 위해 쿼리 시스템을 제공한다. 이는 SQL-like 라는 질의 언어를 지원한다. 이를 위해 Server-side와 in-network 미들웨어가 협력하여 동작하지만, TinyOS 기반의 센서 노드만 이용가능하기 때문에 범용적으로 사용하기 어렵다. TinyDB와 같은 데이터베이스 기반의 미들웨어는 Courgar[9], SINA[10], DSWare[11] 등이 존재한다. Courgar[9]는 센싱 정보를 서버에서 모두 불러와서 그 후에 데이터베이스 기반으로 접근하여 질의를 수행하는 방법이다. 이는 In-network 방식을 배제하고 오직 Server-side 미들웨어를 위한 방식으로 질의 수행에 최적화되어 있다. 하지만, 센서 노드의 데이터를 모두 서버가 수집해야 하기 때문에 센서 노드에 부하가 많이 발생한다. SINA[10]는 Cougar[9]와 비슷하지만 지리적으로 인접한 센서 노드들을 클러스터(cluster)로 묶어서 관리함으로써 센서 노드의 부하를 줄이기 위한 미들웨어 시스템이다. 하지만 이 미들웨어 역시 센싱 정보를 유지해야하기 때문에 클러스터 헤드로 선정된 노드의

에너지 부하가 많이 발생한다. DSWare[11]는 In-network 미들웨어 방식으로 센서 노드의 데이터를 취합하는 방식으로 센서 노드들에 대한 동적인 그룹 관리 방법을 지원하는 미들웨어 시스템이다. 하지만 특정 노드에 대한 의존성이 강하기 때문에 이기종의 센서 노드들에 대한 추상화된 인터페이스를 제공하지 못 한다. 이벤트 기반 프로그래밍 모델을 지원하는 Impala[12]는 손쉬운 어플리케이션 적용성, 업데이트, 내고장성, 에너지 효율성을 기반으로 디자인되었다. 즉, 센서 노드의 기능을 동적으로 쉽게 갱신할 수 있는 미들웨어 시스템이다. 하지만, Impala[12]는 데이터 취합을 적용할 수 없고, 이기종 간의 네트워크 이질성을 고려하지 않았기 때문에 다양한 센서 네트워크에서 적용하기 어렵다. Agilla[13]는 스택(stack) 구조를 지닌 모바일 에이전트(agent) 기반 미들웨어 시스템이다. 스택 기반 구조를 통해 미들웨어 코드를 줄인 Agilla[13]는 명령을 통해 코드의 복제 및 이동할 수 있고, 다수의 어플리케이션들에게 동시에 지원이 가능하다는 장점을 가진다. 하지만, 에이전트 활동을 모니터링하거나 적합한 에이전트의 접근을 허용하기 위한 인증에 대한 정책이 없고, 한정된 자원을 갖는 센서 노드가 프로그램들을 읽거나 유지하는데 있어 어려움을 가진다.

위와 같이 다양한 미들웨어가 연구되었지만, 사용자 어플리케이션과 센서 어플리케이션을 동시에 지원할 수 있는 미들웨어는 존재하지 않았다. 또한 이 기종간의 센서 네트워크를 지원하기 위한 통합된 데이터 포맷이 제공되지 않았기 때문에 현존하는 센서 네트워크의 활용과 나아가 새로운 센서 네트워크를 구성에 있어서 협업을 어렵게 만드는 요인이었다. 따라서 본 논문은 이를 보완할 수 있는 이기종간 센서 네트워크를 위한 데이터 중심적인 센서 미들웨어를 디자인하고 구현한다.

Ⅲ. 데이터 중심적 미들웨어 설계

센서 네트워크는 일반적으로 제한된 자원으로 인하여 데이터 중심적인 메커니즘으로 설계하여야 한다. 따라서 노드와 싱크간의 미들웨어를 구축하기 위해서는 센싱 데이터를 Publish/Subscribe 방식으로 데이터를 전송할 필요가 있다. 또한, 수집된 데이터는 다양한 센서의 종류와 다양한 로우데이터(raw data) 형태를 가지기 때문에 이를 통합 관리하여 공통된 형태의 데이터 포맷(format)을 가지고

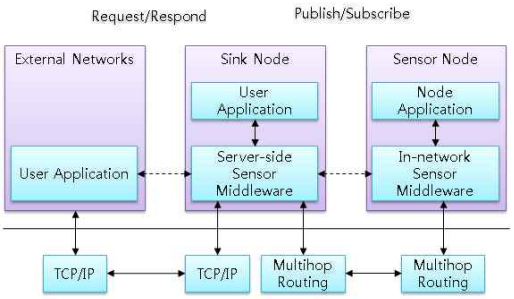

그림 1. 데이터 중심적 미들웨어 구조
Fig. 1 Data-centric middleware architecture

활용할 필요가 있다. 이러한 공통된 데이터 포맷은 센서 네트워크를 이용하는 사용자 즉, 외부 및 싱크 노드의 어플리케이션 이용자에게 일률적인 형태로 제공할 수 있기 때문에 다양한 사용자의 요구에 맞게 적용하여 사용할 수 있다. 따라서 본 논문의 데이터 중심적 미들웨어 설계는 In-network 미들웨어와 Server-side 미들웨어로 나누어서 설계한다. 이는 그림 1과 같다. 그림 1에서 센서 네트워크내의 센서 노드들은 각 노드 어플리케이션이 In-network 미들웨어를 기반으로 구성되고, 둘 간의 통신은 Publish/Subscribe 방식을 사용한다. 이는 이전의 연구[14]에서 구현된 방식을 사용한다. 수집한 싱크 노드 혹은 서버는 Server-side 미들웨어를 기반으로 사용자 어플리케이션과의 질의응답을 위해 구성한다. 그리고 사용자 어플리케이션은 Server-side에서 제공하는 통합된 센싱 데이터 포맷을 이용하여 어플리케이션을 구현한다.

1. In-network 미들웨어

센서 네트워크의 어플리케이션 통신 방법은 이벤트 기반의 통신이다. Request/Respond 방식으로 동작하는 매커니즘은 데이터의 폴링(polling)을 위해서 프로세싱을 지속적으로 사용해야 하며 불필요한 데이터의 전송을 필요로 하기 때문에 특정 상황에 반응하는 센서 네트워크의 속성에는 적합하지 않기 때문이다. 따라서 센서 네트워크에서는 이벤트 중심의 매커니즘이 필요하며 이는 센서 네트워크 미들웨어의 중요한 기능 중 하나이다. 상황 이벤트 기반의 센싱 데이터 처리 매커니즘은 비동기식 통신을 지원하고, 센서 노드와 싱크노드 사이에 송신자를 분리하고 비수신자와 전동기적으로 동작하게 함으로써 전체적으로 데이터 처리의 효율을 높일 수 있게 된다. 따라서 이벤트 기반 방식의 센서 네

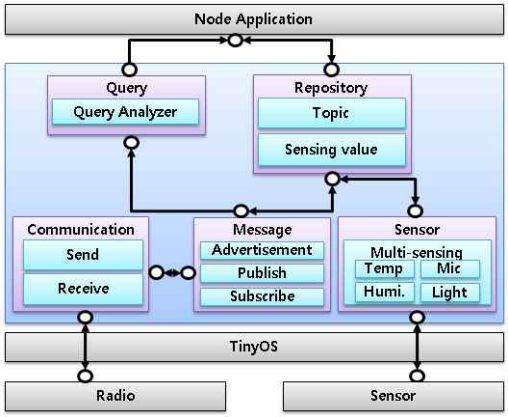

그림 2. In-network 미들웨어 구조
Fig. 2 In-network middleware architecture

트워크는 데이터를 가진 노드가 데이터를 publish하고 특정 센싱 데이터를 필요로 하는 베이스 스테이션(혹은 sink 노드)이 해당 노드에 subscribe를 하는 publish/subscribe 방법이 사용된다. 이러한 publish/subscribe 방식은 비동기적인 특성과 멀티포인트(Multi-point) 통신 속성을 지닐 수 있는 것이다[15-17].

In-network 미들웨어는 센서 노드의 능력에 맞게 데이터 형태를 생성하고, 이를 싱크노드로 'advertise'를 메시지를 통해 전달한다. 이유는 사용자 어플리케이션의 요구에 맞는 데이터를 publish하는 역할을 가지기 때문이다. 그림 2와 같이, 센서 노드 안에 TinyOS[8]와 노드 어플리케이션 사이에 위치한다. 이는 다섯 개의 컴포넌트로 구성된다. 먼저, Sensor 컴포넌트는 센서 어플리케이션으로부터 요청받은 센서 데이터를 수집하고 형태를 파악하여 토픽(Topic)을 생성한다. Repository 컴포넌트는 Sensor 컴포넌트로부터 받은 토픽과 이웃 노드로부터 수집한 Updated-topic을 저장하고, Sensor 컴포넌트로부터 수집된 센싱 데이터를 저장하는 역할을 한다. Message 컴포넌트는 센서 노드의 토픽을 전송하기 위한 'advertise' 메시지를 생성하고, Server-side 미들웨어로 부터 수신한 'subscribe' 메시지에서 사용자의 요구사항 즉 쿼리(query)을 추출한다. 또한 Subscribe와 연관된 센싱 데이터를 전송하기 위해 'publish' 메시지를 생성한다. Query 컴포넌트는 수신된 쿼리를 분석하여 센서 노드의 어플리케이션에 맞게 태스크(task)를 형성한다. 마지막으로 Communication 컴포넌트는 센서 노드가 생성한 메시지를 싱크 노드로 송신하고

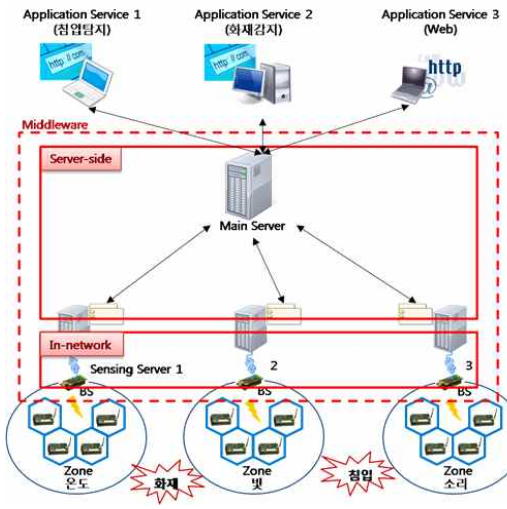

그림 3. Server-side 미들웨어 구조
Fig. 3 Server-side middleware architecture

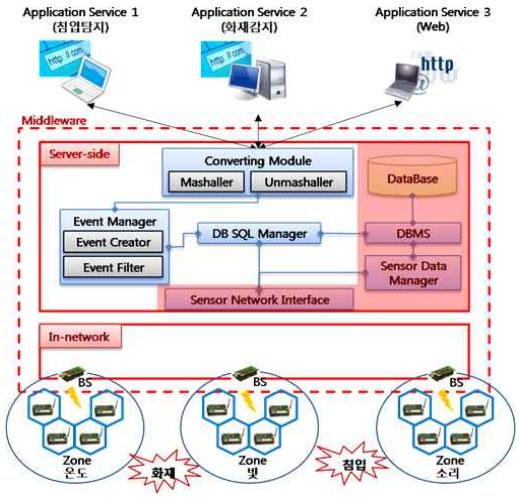

그림 4. Server-side 미들웨어 컴포넌트
Fig. 4 Server-side middleware component

Server-side 로부터 메시지를 수신하는 역할을 한다. 하지만, In-network에서의 publish/subscribe의 방식만으로는 publish/subscribe 형태의 미들웨어로 구성하여 각 노드의 어플리케이션을 개발할 경우, 사용자의 요구사항을 쿼리 형태의 명령어로 전환하여 노드가 사용자의 요구사항에 맞는 역할을 수정하여 데이터 전송을 가능하게 하였지만, 센서 네트워크가 아닌 다른 네트워크와의 접속 혹은 다양한 센서 네트워크와의 데이터 교환은 공통된 데이터

형태를 가지지 않기 때문에 불가능하다. 따라서 센서 네트워크의 데이터를 수집 및 관리하는 Server-side 미들웨어에서는 데이터를 하나의 공통된 형태의 포맷으로 관리할 필요가 있다. 이는 센서 네트워크내의 노드 어플리케이션 뿐 아니라 외부 네트워크 혹은 이기종의 센서 네트워크와의 연동을 지원해야 한다.

2. Server-side 미들웨어

Server-side 미들웨어는 In-network 미들웨어와 상호작용할 수 있는 부분과 사용자 어플리케이션과 상호작용할 수 있는 부분으로 나뉜다. 이는 그림 3에서와 같은 형태이다. 그림 4에서 같이 센 Server-side 미들웨어에서 센싱 서버에는 센서 데이터 매니저 컴포넌트와 센서 네트워크 인터페이스 컴포넌트를 구성한다. 센서 데이터 매니저 컴포넌트는 노드로부터 수집된 데이터를 파악하고 센싱 데이터를 메타 데이터 형태로 생성하여 등록, 검색, 삭제, 저장과 같은 역할을 센싱 서버에서 수행한다. 또한 센싱 서버와 메인 서버간의 통신을 위해 센서 네트워크 인터페이스 컴포넌트를 제공하여 두 서버간의 통신을 담당한다. 즉, 센싱 서버에 있는 Server-side 미들웨어는 센서 데이터의 수집 및 저장을 담당 한다.

메인 서버의 Server-side 미들웨어는 다양한 사용자의 어플리케이션지원을 위해 센싱 데이터를 통합된 형태로 제공함으로써, 공통된 메시지 타입(XML형태), 사용자 인터페이스(API), 사용자의 요구를 이벤트 패턴형태로의 생성 및 제공 하는 역할이다. 그림 4와 같이, Converting 컴포넌트는 사용자로부터 수집된 XML 메시지를 통합된 형태로 변환하거나 센서 네트워크로부터 수집된 데이터를 XML 형태로 변환하는 컴포넌트이다. Marshaller 메소드(Method)는 XML 메시지를 공용 데이터 형식으로 변환하고, UnMarshaller 메소드는 사용자 요청 결과를 XML 객체로 변환한다. Event manager 컴포넌트는 사용자가 요구하는 이벤트 패턴에 대해 분석하고 저장한다. 이를 위해 어플리케이션 요청에 따른 이벤트 패턴을 생성하는 Event creator 메소드와 이벤트 요청 및 등록된 이벤트 패턴 비교를 위한 Event filter 메소드가 존재한다. DB SQL manager 컴포넌트는 이벤트 패턴과 센서 데이터를 데이터베이스 저장 형식으로 변환하고 이벤트 패턴의 등록, 검색, 삭제, 저장을 담당한다.

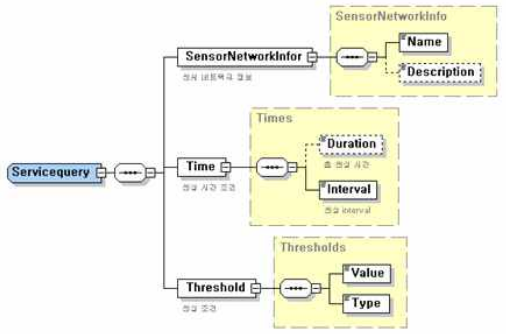

그림 5. XML 스키마 구성 : 서비스쿼리
Fig. 5 XML schema configuration : Service Query

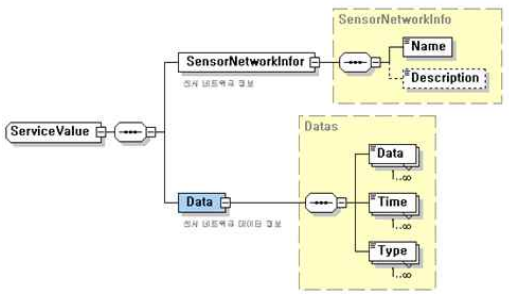

그림 6. XML 스키마 구성 : 서비스 밸류
Fig. 6. XML schema configuration : Service value

Ⅳ. 데이터 중심적 미들웨어 구현

미들웨어 구현을 위해서는 다음과 같은 가정을 세운다. 먼저, 메인 서버와 센싱 서버 간의 통신은 기본적으로 HTTP/IIOP 프로토콜을 이용한 통신을 사용한다. 센싱 서버는 각 센서 네트워크의 베이스 스테이션과 직접적으로 연결되어 배치한다. 메인 서버는 센싱 서버의 위치와 무관하게 설치될 수 있고, TCP/IP를 통해 사용자 어플리케이션과 통신한다. 센싱 서버들은 자신이 포함된 센서 네트워크가 제공하는 데이터에 대해 메인 서버에 등록과정을 거친다. 사용자 어플리케이션은 자신이 원하는 종류의 데이터에 대한 정보를 포함한 이벤트를 메인 서버에 등록한다. 메인 서버는 자신의 하부의 센싱 서버를 관리하며, 사용자 어플리케이션이 요청한 데이터에 대한 이벤트가 발생 시 이를 사용자 어플리케이션에 제공한다.

In-network 미들웨어는 [14]과 같이 센서 노드

그림 7. 센싱 서버 구축
Fig. 7. Sensing server construction

는 무선 모듈로 CrossBow사의 MPR 2400, 센서 보드로는 MTS 300, 게이트웨이로는 MIB 520을 사용하였다. 각 노드의 OS는 TinyOS[8]을 사용하였고, 사용한 언어는 NesC[18]이다. Server-side 미들웨어는 Window XP상에서 C++를 이용하여 구현하였다. 그리고 센싱 서버와 연결되어 WSN과 통신하는 싱크 노드로는 위에서 보았던 센서 노드의 구성을 따랐고, TinyOS에서 제공하는 TOSBASE로 프로그래밍 하였다.

사용자 어플리케이션과의 공통된 데이터 형식을 위해서 XML 스키마(Schema)는 그림 5와 6과 같이 서비스 쿼리와 서비스 밸류(value)을 위한 스키마를 구성한다.

그림 7과 같이 실험을 위해 사용자는 두 가지의 어플리케이션 침입탐지와 화재감지를 구현하고, 집안에 역할이 다른 센서 노드들이 배치된다. 각 센서의 역할(소리, 빛, 온도)는 각 센싱 서버로 각각 수집되어 관리된다. 테스트 환경은 다음 표 1과 같다.

실험은 각 센싱 서버로부터 수집된 데이터가 메인서버에서 통합관리 되어 통합적인 데이터 형태로 어플리케이션에 제공하고 이러한 데이터가 각 어플리케이션에서 요구되는 데이터를 전송하는 것을 실험하였다. 그림 8와 같이 메인 서버는 이기종의 센서 네트워크의 센싱 서버로부터 데이터를 수집하고, 이러한 센서 네트워크의 데이터를 요구하는 어플리케이션으로 데이터를 통합된 형태로 제공한다. 사용자 어플리케이션은 메인 서버로 자신의 역할에 맞는 XML을 보내고 이에 맞는 데이터를 수집하여 사용자에게 보여준다. 이는 그림 9와 같다. 센싱 서버에서 센서 네트워크에서 수집되는 데이터는 센싱 서버에서 관리하기 때문에 이를 위한 어플리케이션을 따로 구현하였다. 이는 그림 10과 같다.

표 1. 실험 환경
Table 1. Test environment

센서노드	Micaz & MTS 310
베이스스테이션	MIB 520, H-mote
센싱서버 1	노트북(core2duo) - 온도
센싱서버 2	노트북(core2duo) - 조도
센싱서버 3	노트북(core2duo) - 소리
메인서버	Core2duo e6300 2G ram
데이터베이스	My_sql 5.1.4

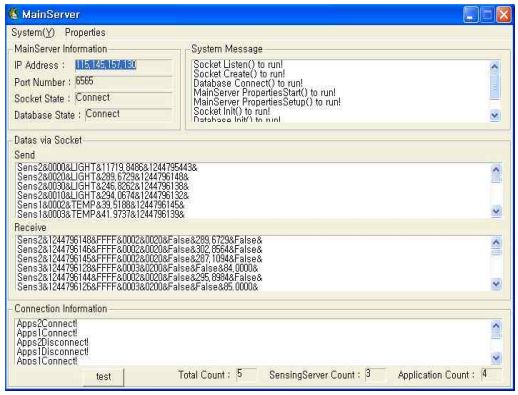

그림 8. 메인 서버
Fig. 8 Main server

그림 9. 화재 경보 어플리케이션
Fig. 9 Fire detection application

센서 노드는 가상의 집안 환경을 기반으로 각각 역할에 맞게 NesC로 프로그램 되어 온도, 빛, 소리를 위해 각각 배치되었고 이는 그림 11와 같이 배치하였다. 그리고 각 센서를 수집하기 위한 센싱 서

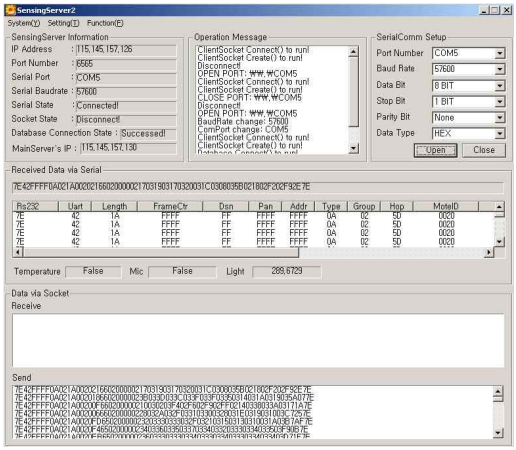

그림 10. 센싱 서버 관리 어플리케이션
Fig. 10 Sensing server management application

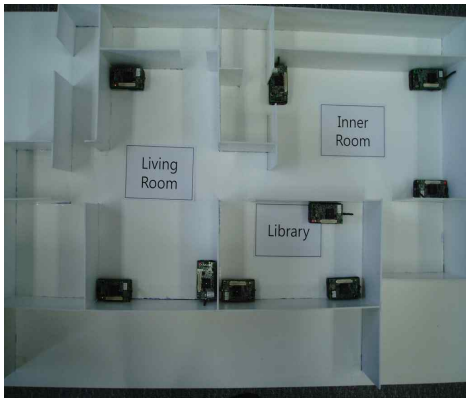

그림 11. 테스트를 위한 노드 배치
Fig. 11 Node arrangement for test

버와 메인 서버는 그림 12과 같이 배치하였다.

실험결과 각 XML로 변환된 사용자 요구사항은 메인서버에서 관리되고, 센싱 서버로부터 수집된 데이터 중에서 사용자의 요구와 맞는 데이터만이 어플리케이션으로 전송됨을 위 그림 10으로 확인하였다. 이를 통해 통합된 데이터 형태를 가지고, 다양한 어플리케이션으로의 확장과 이기종 센서 네트워크를 통합 관리할 수 있는 미들웨어를 구축할 수 있었다.

V. 결론

이기종 간의 센서 네트워크와 다양한 사용자 어플리케이션의 요구를 충족하기 위해서 센서 네트워크의 제한된 자원을 이용한 데이터 중심적 통신과 통

그림 12. 메인서버 및 센싱 서버 배치
Fig. 12. Main & sensing server arrangement

합된 데이터 관리를 통한 확장성이 용이한 미들웨어를 설계 및 구현하였다. 이를 통해 어플리케이션 개발 및 확장이 용이하였다. 이는 개발자에게 통합 API를 제공함으로서 개발 비용을 감소하고, 공통 메시지 타입(XML)의 센싱 데이터 처리 모듈 개발에 따른 어플리케이션 확장성을 제시하였으며, 이기종 센서 네트워크에서 수집된 데이터의 효율적 관리를 가능하게 하였다. 또한, 사용자의 요구 사항의 능동적인 대응이 가능하여, 요구사항의 이벤트 패턴 분석을 통한 정보제공, 이벤트 패턴을 통한 지능형 쿼리 가능, 각 센서 네트워크 협업을 통한 다양한 서비스가 가능해진 것이다.

참 고 문 헌

[1] L.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, " A Survey on Sensor Networks," IEEE Communication Magazine, Vol. 40, No. 2, pp.102-114, 2002.

[2] K. Romer, O. Kasten, F. Mattern, "Middleware challenges for wireless sensor networks," ACM SIGMOBILE Mobile Communication and Communications Review, Vol. 6, No. 2, 2002.

[3] M.S. Kim, K.S Kim, Y.J. Lee, "Trend of USN Middleware Technology," ETRI: Trend Analysis of Electronic & Telecommunication, Vol. 22, No. 3, pp.67-79, 2007.

[4] Y. Yu, B. Krishnamachari, V.K. Prasanna, "Issues in designing middleware for wireless sensor networks," IEEE Network, Vol. 18, No. 1, pp.15-21, 2004.

[5] S. Hadim, N. Mohamed, "Middleware: middleware challenges and approaches for wireless sensor networks," Distributed

Systems Online, IEEE, Vol. 7, No. 3, pp.1-26, 2006.
[6] W.B. Heinzelman, A.L. Murphy, H.S. Carvalho, M.A. Perillo, “Middleware to support sensor network applications,” Network, IEEE, Vol 18, No. 1, pp.6-14, 2004.
[7] S.R. Madden, M.J. Franklin, J.M. Hellerstein, W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," Journal of ACM Transaction on Database Systems, Vol. 30, No. 1, pp.122-173, 2005,
[8] TinyOS communities, TinyOS specification, http://www.tinyos.net
[9] Y. Yao, J.E. Gehrke, "The Courgar Approach to In-Network Query Processing in Sensor Networks," SIGMODE RECORD, Vol. 31, No. 8, pp.9-18, 2002.
[10] C. Srisathapornphat, C. Jaikaeo, C.C. Shen. “Sensor Information Networking Architecture,” Proceedings on International Workshops on Parallel Processing, pp.23-30, 2000.
[11] S. Li, S.H. Son, J.A. Stankovic, "Event Detection Services Using Data Service Middleware in Distributed Sensor Networks," Telecommunication Systems, Vol. 26, pp351-368, 2004.
[12] T. Liu, M. Martonosi, "Impala : A Middleware System for Managing Autonomic, Parallel Sensor Systemsm" Proceeding on ACM SIGPLAN Symp. Principles and Practice of Parallel Programming, pp.107-118, 2003.
[13] C.L. Fok, G.C. Roman, C. Lu, "Agilla : A Mobile Agent Middleware for Self-adaptive Wireless Sensor Networks," Journal of ACM Transactions on Auntomous and Adaptive System, Vol. 4, No. 16, pp.1-25, 2009.
[14] C.S. Nam, H.J. Jeong, D.R. Shin, "Design of Wireless Sensor Networks Middleware using the Publish/Subscribe Paradigm," Proceeding on IEEE International Conference on Serive Operations and Logistics, and Informatics, Vol. 1, pp.559-563, 2008.
[15] P. Eugster, P. Felber, R. Guerraoui, A. Kermarrec, “The Many Faces of Publish/Subscribe,” ACM Computing Surveys, Vol. 35, No. 2, pp.114-131, 2003.
[16] E. Souto, G. Guimaraes, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz, A message-oriented middleware for sensor networks," Proceeding on the 2nd workshop on Middleware for pervasive and ad-hoc computing(MPAC 04), pp.127-134, 2004.
[17] Y. Huang, H. Garcia-Molina, “Publish/Subscribe in a mobile environment,” Wireless Networks, Vol. 10, pp.643-652, 2004.
[18] D. Gay, P. Levis, D. Culler, E. Brewer “nesC 1.1 Language Reference Manual,” 2003.

저 자 소 개

남 춘 성

2005년 상명대 소프트웨어 학사.
2007년 숭실대 컴퓨터 석사.
2011년 성균관대 전자전기컴퓨터 박사.
현재, 성균관대학교 박사후연구원
관심분야: 센서네트워크, 센서 미들웨어, VANET.
Email: namgun99@skku.edu

신 동 렬

1980년 성균관대 전자공학 학사.
1982년 KAIST 전기 및 전자공학과 석사.
1992년 Georgia Tech 전기 및 전자공학과 박사.
현재, 성규관대 정보통신공학과 교수.
관심분야: 유비쿼터스 컴퓨팅, 센서네트워크, 모바일 미들웨어.
Email: drshin@skku.edu