韓國컴퓨터情報學會 論文誌
第15卷 第9號, 2010. 9.

2010-15-9-4-1

USN 환경에서 실시간 상황인식 QoS 보장을 위한 범용 미들웨어 설계

김광훈*, 최운수*, 이태우*, 이준석*, 구경옥**, 조용환*

Design of General-Purpose Middleware for QoS Guaranteed Context-Aware Services on USN Environment

Guang-Xun Kim*, Woon-Soo Choi*, Tae-Woo Lee*, Joon-Suk Lee*, Kyoung-Ok Koo**, Yong-Hwan Cho*

요 약

유비쿼터스 컴퓨팅 시대가 도래하면서 USN(Ubiqutious Sensor Network) 관련 핵심기술 개발에 대한 관심이 급증하고 있다. 특히, ZigBee, Bluetooth, WLAN, CDMA 등과 같이 다양한 종류의 무선통신 기술과, Nano 계열, Mote계열, NeurFon계열 등과 같은 여러 종류의 센서노드들에 대한 연구가 세계 각 대학과 연구소를 중심으로 진행되고 있다. 이러한 연구 중 본 논문에서는 USN 응용 시스템과 센서노드 하드웨어의 중간 부분에 위치하여, 수많은 센서 데이터들을 수집 및 처리하고 USN 응용서비스의 실시간 상황인식 QoS를 보장할 수 있게 조력하는 USN 미들웨어를 설계하고 그 성능을 시뮬레이션을 통해 증명 한다.

Abstract

Recently, many researchers interested in core technology related to USN(Ubiqutious Sensor Network) with emerging Ubiqutious computing era. Especially, researched about various wireless telecommunication technologies such as ZigBee, Bluetooth, WLAN, CDMA, and various sensor nodes such as nano, Mote, and NeurFon are progressed in universities and research institutes of the world. In this paper, an advanced algorithm is proposed by analyzing problems of circumstance recognition and conventional real-time QoS of USN middleware technology. Finally, the performance of the proposed USN middleware is demonstrated through simulation.

▸ Keyword : 유비쿼터스(Ubiqutious), 센서(Sensor), 미들웨어(Middleware), 상황인식(Context-Aware)

• 제1저자 : 김광훈 교신저자 : 조용환
• 투고일 : 2010. 07. 05, 심사일 : 2010. 07. 18, 게재확정일 : 2010. 08. 11.
* 충북대학교 전자정보대학 ** 강릉영동대학 사무정보과
※ 이 논문은 2009년도 충북대학교 학술연구지원사업의 연구비지원에 의하여 연구되었음
(This paper was supported by the research grant of the Chungbuk National University in 2009).

Ⅰ. 서 론

최근 u-Healthcare, u-Hospital, u-Transpotation, u-City 등과 같이 다양한 형태의 USN(Ubiquitous Sensor Network) 응용서비스가 출현하고 있다. 이러한 USN 응용서비스에는 실세계의 정보를 효율적으로 수집, 전달, 처리 및 신속한 응답을 구하기 위해 위 기능들이 통합된 USN 미들웨어(Middleware) 기술이 필요하다. USN 미들웨어는 이기종 센서 네트워크로부터 수집된 데이터를 필터링하고 통합 분석하여 의미 있는 상황 정보를 추출, 저장, 관리하고 USN 응용서비스로 전달 및 서비스 간 연계, 통합하는 역할을 수행한다. 또한 USN 미들웨어는 센서 네트워크 내에 존재하는 노드들의 제한된 자원(저성능 CPU, 저용량의 메모리, 낮은 통신 대역폭 등)으로 발생할 수 있는 문제점들을 해결하는데 중요한 기능을 담당한다[1].

USN응용서비스가 현 상황을 실시간으로 인식하고 처리하기 위해서는 수많은 센서 데이터들을 수집하고 분석하는 속도가 가장 중요할 것이다. 그러나 센서데이터를 수집하기 위해서는 수많은 문제들이 주어진다. 프로토콜들이 서로 상이한 센서들을 통합하는 문제, 센서 데이터들 간의 우선순위 결정 문제, 낮은 대역폭을 사용하는 환경에서 무수히 많은 센서 데이터를 주기적으로 전송해야하는 문제들이 그것이다[2]. 따라서 앞서 지적한 문제들을 해결할 수 있는 범용적인 USN 미들웨어를 설계하고 이를 통해 USN응용서비스의 실시간 상황인식 QoS를 보장하기 위한 연구가 시급한 실정이다.

Ⅱ.USN 미들웨어

USN 미들웨어는 USN 응용 서비스의 시스템과 센서 네트워크 인프라의 중간에 위치하여 이들의 원활한 상호관계를 가능하게 하는 역할을 수행하며, 설치된 위치에 따라 그림1과 같이 Server-side 미들웨어 와 In-network 미들웨어로 나눈다[1].

Server-side 미들웨어는 싱크노드와 USN 정보시스템 사이에 있으며 기본 기능으로 USN 응용서비스의 다중 질의처리, 센서 정보의 필터링, 통합, 분석 및 메타정보 관리, 상황정보 수집과 상황 정보관리, 지능형 이벤트 처리를 수행하는 컴포넌트들로 구성될 수 있다.

반면 In-network 미들웨어는 센서 노드와 싱크 노드 사이에 있는 미들웨어로서 대부분 센서 노드와 싱크노드 수준에서의 질의 처리, 센서 네트워크 상태에 관한 실시간 모니터링 및 센서 데이터의 획득, 가공 저장, 분석, 그리고 센서 노드의 관리 등을 제어할 수 있는 작은 모듈로 구성될 수 있다[1].

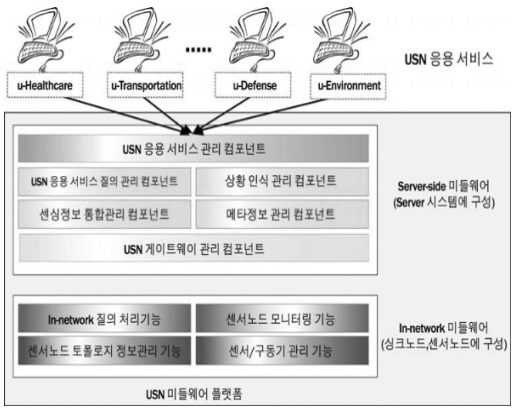

그림 1. USN 미들웨어의 개념적 모델
Fig 1. Conceptual model of the USN middleware

2.1 주요 USN 미들웨어

본 절에서는 USN 미들웨어의 기능적인 분류에 따라 특정 상황이나 응용에 특화되어있는 대표적인 미들웨어들을 소개하고 각각의 특징에 대하여 설명 한다.

2.1.1 Cougar USN 미들웨어

Cougar 미들웨어는 Server-side 미들웨어의 대표적인 예로서 서버시스템이 모든 센싱정보를 데이터베이스를 통해 통합 관리하는 형태를 취하며, USN 응용시스템의 요구하는 질의 사항을 센서노드를 통하여 분산질의 처리 하는 USN 미들웨어이다. Cougar 미들웨어는 모든 센싱 정보를 서버에 취합한 후 DB 기반 접근 방식으로 질의를 처리하며, SQL (Structured Query Language) - like 언어를 지원하여 USN 응용시스템이 센서 네트워크에 존재하는 센싱 정보를 보다 빠르게 수용할 수 있는 환경을 지원한다. 뿐만 아니라 센서 노드들로부터 수집된 센싱 정보들을 통해 얻어진 결과를 데이터스트림 관리 시스템(DSMS)의 쿼리 옵티마이저 (Query Optimizer)를 이용해 USN 응용 시스템의 질의 처리에 대한 최적화 방법을 제안하고 있다[3].

2.1.2. MiLAN USN 미들웨어

MiLAN 미들웨어는 In-network 미들웨어의 한 예로서 센서 네트워크내의 서비스 품질과 관련된 가장 대표적인 미들웨어이다. MiLAN 미들웨어는 USN 응용 서비스에 따른 효

과적인 QoS 요구조건 보장을 주요 목표로 개발 되었다. 이를 위하여 MiLAN 미들웨어는 센서 네트워크 응용계층에서 요구하는 QoS를 미리 센서 노드, 라우팅 및 서버에 기술하고 USN 응용 계층에서 서비스 품질을 요구할 때, 현재 센서 네트워크의 전체 자원과 비교하여 절충한 응답을 응용 계층에게 전송함으로써 최적의 실행 조건을 도출한다. 이러한 특징으로 MiLAN 미들웨어는 USN 응용 계층이 요구하는 QoS를 최대로 만족시킬 수 있다. 또한 MiLAN은 USN 응용계층에서 필요로 하는 QoS를 표현하기 위해 자체 API와 명령 등을 적용대상 네트워크 프로토콜의 명령으로 확장이 가능하기 때문에 여러 가지의 이질화된 물리 네트워크를 연결해줄 수 있다.

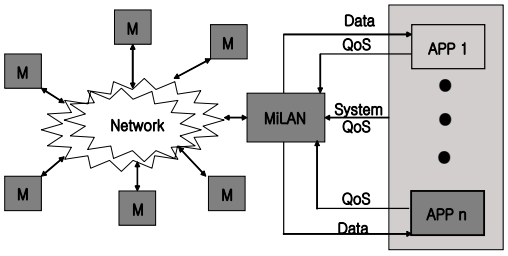

그림 2. MiLAN 시스템의 구조
Fig 2. a MiLAN system's structure

그림 2에서 볼 수 있듯이 MiLAN은 어플리케이션이 원하는 QoS를 만족시키기 위해 센서 네트워크의 응용이 요구한 정보를 시스템의 자원과 비교·분석한다. 또한 네트워크 상태를 파악하여 최적의 전송률을 설정하고 어플리케이션에서 원하는 결과를 제공한다[4].

2.1.3 SensorWare USN 미들웨어

SensorWare 미들웨어는 센서 노드의 효율적인 활용을 목적으로 개발되었으며, 모바일 에이전트를 지원하기 위한 대표적인 USN 미들웨어이다. 또한 SensorWare 미들웨어는 스크립트화 할 수 있는 경량의 런타임 환경을 기반으로 하는 프레임워크이며, 동적 네트워크를 구성을 통해 스크립트들은 한 노드에서부터 다른 노드에 복사하거나 이동할 수 있다. 이를 통해 플랫폼과 독립적으로 센서 노드를 제어할 수 있다.

SensorWare 미들웨어는 센서 네트워크가 설치된 후 관리자의 손길이 닿지 못하는 환경에서 오랜 기간 자율적으로 동작되고 있으며 특정 센서가 사용 중에 용도가 변경될 경우, 센서 노드의 동작 스크립트를 능동적으로 다운로드한다. 뿐만 아니라 응용프로그램, 미들웨어 및 운영체제를 구성하는 모듈을 자동으로 업데이트 할 수 있는 기능 등을 제공한다[5]. 그림 3은 SensorWare의 센서 노드 구조이다.

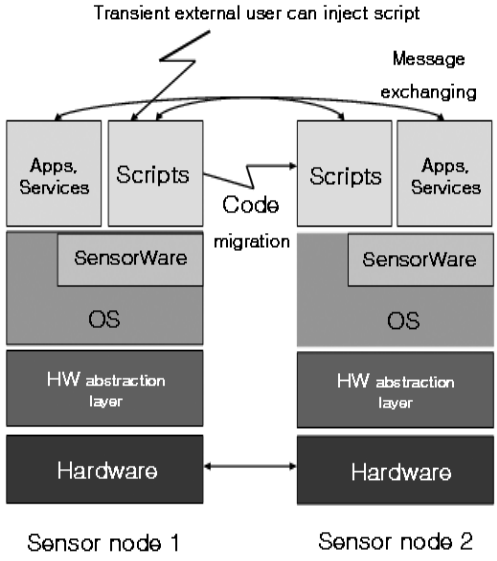

그림 3. 센서 노드 구조
Fig 3. Sensor node architecture

이중 하드웨어와 하드웨어 추상화 계층은 센서 노드의 제일 하위 계층에 존재하며, 운영체제는 표준기능과 멀티 스레드 환경의 서비스를 모든 계층에게 제공한다. 이는 운영체제의 런타임 환경을 통해 SensorWare의 제어 스크립트를 동작시키는 것을 의미하며, SensorWare의 제어 스크립트가 네트워크상에서 동작될 수 있는 환경을 제공한다. 운영체제를 바탕으로 동작되는 SensorWare를 기반으로 모바일 스크립트와 응용서비스는 서로 공존하면서 SensorWare 계층의 일부 기능과 운영체제의 표준 기능 및 서비스를 사용한다.

2.2 주요 USN 미들웨어 분석

본 절에서는 2.1 절에서 살펴본 주요 미들웨어들 간의 특징과 장단점을 분석한다.

첫째로, 센싱된 데이터의 수집 및 처리 정확성을 위하여 개발된 Cougar미들웨어는 SQL-like 질의 언어를 사용함으로써 모든 센싱 정보를 서버에 취합한 후 DB 기반 접근 방식으로 질의 처리하기 때문에 서버에 DB화 되어 존재하는 센서 정보를 보다 빠르게 수용할 수 있는 장점이 있다. 그러나 서버 시스템이 모든 센싱 정보를 유지하고 있어야 하며 센서 노드들 또한 센싱 정보를 모두 서버로 전송하기 때문에 서버의 부하가 커진다는 단점과, 반복적으로 센서데이터를 서버로 전송하는 전송 부하로 인해 전송 지연이 크다는 단점이 있다.

둘째로, USN 응용 서비스에 따른 QoS 보장을 주요 목표로 개발된 MiLAN 미들웨어는 USN 응용 시스템의 QoS 요구 처

리 기능을 지원하면서 네트워크 리소스에 따른 절충된 응답을 통해 응용시스템의 QoS 요구를 최대로 만족시킬 수 있는 장점을 갖지만 이종의 센서노드들에 대한 추상화를 지원하지 못한다는 단점이 있다. 이는 USN을 구성하는 센서들이 동일 기종이며, 센서 메타데이터들이 동일한 포맷을 지원하지 않는다면 그에 따른 API를 추가적으로 지원해야 한다는 맹점을 갖는다.

마지막으로 센서 네트워크의 효율적인 활용을 목적으로 개발된 SensorWare 미들웨어 역시 여러 응용프로그램 사이에서 센서 노드끼리의 자원을 공유하는 방식을 제공할 뿐만 아니라 고급 스크립트 언어를 사용함으로 센서 네트워크 환경에서 편리한 개발 환경을 제공할 수 있다는 장점을 갖는다. 그러나 이 역시 메모리가 풍부한 환경에서 고급 스크립트 언어를 통해 운영되므로 제한된 메모리를 가진 센서 노드에는 적용이 부적합 하다는 단점이 존재한다. 이는 모든 센서 노드들이 스마트 노드 즉, 운영체제를 가진 디바이스여야 한다는 문제점이 있다.

지금까지 "USN 미들웨어의 특징 및 기술 개발 동향"[1]에 따른 미들웨어의 분류 중 각 분류를 대표할만한 미들웨어들의 장단점을 분석 해 보았다.

최근 USN 분야의 흐름을 읽어보면 u-Healthcare, u-Hospital, u-Transportation, u-City 등과 같은 USN 응용 서비스가 출현되고 있으며, USN 응용 서비스 모델들이 상호 유기적으로 통합되어 다양한 서비스를 제공하는 시점에서 센서 네트워크에 대한 메타정보의 공유 불가능, 센서 데이터 요청 방법이 서로 다른 문제들이 도출되어 왔다. 또한 USN 응용 서비스들이 다양한 사양의 센서 네트워크 및 센서 노드를 이용하고 있기 때문에 이종 센서 네트워크를 지원할 수 없으며 추상화 기능 등을 원활히 지원할 수 없다는 문제점들이 USN 응용서비스 발전의 저해 요소로 부각되고 있다[6]. 이러한 문제점을 해결하기 위해 USN 미들웨어는 특정 상황에 특화된 USN 미들웨어가 아닌 범용적인 USN 미들웨어가 필요하다.

2.3 범용 환경 USN 미들웨어의 요건

범용 환경 USN 미들웨어 요건으로는 센서 네트워크 추상화 기능, 센서 네트워크 지능화 계층 및 서비스 통합 기능 등을 들 수 있다.

첫째, 센서 네트워크 추상화 계층은 다양한 센서 네트워크와 미들웨어를 연동하는 부분으로서 센서 네트워크와 USN 미들웨어 플랫폼사이에 서로 주고받는 명령과 정보를 "공통 메시지"라는 이름으로 정의하고 각 센서 네트워크 별 어댑터를 통하여 이기종 통신 프로토콜을 지원할 수 있어야 한다[7].

둘째, 센서 네트워크 지능화는 센서 정보 통합관리 기능과 센서 데이터 마이닝 기능, 지능형 이벤트 관리 기능 및 상황정보 관리 기능 등을 통하여 센싱 정보를 효율적으로 관리하고 분석하여 새로운 상황정보를 생성하는 역할을 한다.

셋째, 서비스 통합 계층은 USN 응용 서비스의 개발을 효율적으로 지원하는 역할을 한다. 이를 위하여 Open-API를 제공하고 시스템이 요구하는 센서 노드와 센서 네트워크 관련의 메타데이터를 실시간으로 관리하는 기능을 제공한다. 또한 서비스 통합 계층 관리기능은 센서 네트워크내의 자원 및 데이터의 접근에 대한 인증(Authentication), 인가(Authorization), 기밀성, 암호화, 복호화와 같은 보안 기능 역시 선택적으로 제공할 수 있어야 한다.

III. 실시간 상황인식 QoS 보장을 위한 범용 미들웨어

본 장에서 제안하는 미들웨어 모델은 USN 환경에서 응용서비스 및 이기종 센서 네트워크의 원활한 통합을 위한 USN 미들웨어 모델로서 이후 GPMC(General Purpose Middleware for Context- Aware Service)라고 명명한다.

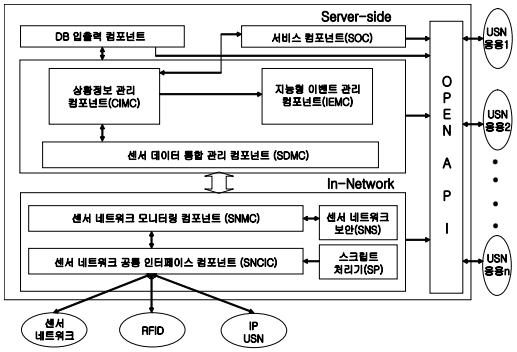

그림 4. GPMC 플랫폼의 컴포넌트 구성 및 연관관계
Fig 4. GPMC platform configuration and relationship of the components

GPMC는 Sever-side 미들웨어와 In-Network 미들웨어를 모두 포함한 시스템 모델이다. 또한 GPMC는 2.3절의 범용 미들웨어 플랫폼 기능요구사항을 바탕으로 설계되었으며, 네트워크 추상화, 지능화를 통해 USN 응용 서비스의 실시간 상황인식 QoS를 보장한다.

GPMC 플랫폼은 그림 4와같이 여러 개의 주요 컴포넌트들이 유기적으로 맞물려 있으며, Open-API를 통해 USN 응

용서비스들의 요구를 인식하고 처리하는 In-network의 구조와 상황인식 및 지능형 이벤트를 통합하여 관리하는 Server-side의 두 형태를 모두 포함하고 있다.

3.1 센서 네트워크 모니터링 동작

그림 5와 같이 GPMC는 스캔장치를 통하여 센서의 데이터를 주기적으로 입력받아 Aggregate Operator를 이용하여 센서 노드들로부터 얻은 데이터들을 병합시킨 후 헤더 노드에 전송한다[8]. 이 과정은 현실세계의 투표와 유사한 형태를 갖는다.

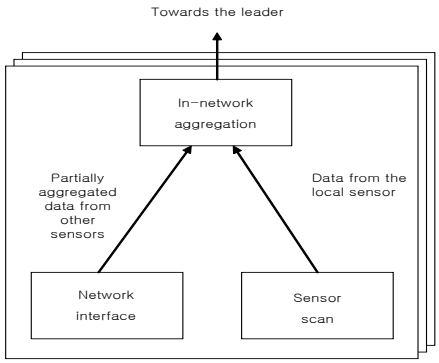

그림 5. 센서 노드들의 데이터 수집 및 병합
Fig 5. Data acquisition and Aggregation of sensor nodes

이때 Aggregation 과정은 그림 6과 같이 쿼리를 통하여 얻어진 센서 수치들의 병합결과 값 AVR(Aggregate Value Results)추출하고 SELECT 모듈을 이용하여 AVR 값이 threshold보다 큰지를 체크한 후 기준에 통과된 AVR 정보만을 전송하게 된다.

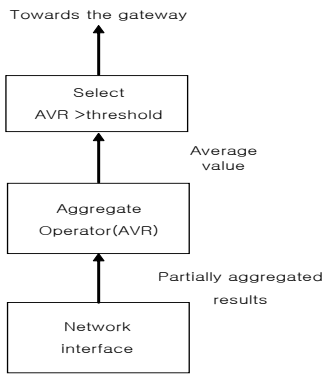

그림 6. GPMC AVR 추출 과정
Fig 6. AVR extraction process of GPMC

3.2 센서 데이터 품질측정

GPMC의 AVR 추출 과정에서는 시간적 상관관계에 대한 허용오차를 이용하여 데이터를 병합한 절차를 통해, 헤드(Head)노드에게 전송될 AVR을 평가할 수 있는 함수를 제시한다. 데이터 병합 동작은 시간상의 연속되는 데이터를 이전 시간의 데이터와의 차이가 설정한 허용 오차 이상인 경우만 데이터를 전송한다. 이러한 허용오차는 데이터의 품질(Quality)을 보장하기 위하여 병합된 데이터가 어플리케이션의 질의에 얼마나 근접한지를 나타내는 AVRQ(Quality of AVR)를 통해 측정되며 AVRQ는 수식 (3-1)과 같다.

$$AVRQ = \frac{1}{T}\sum_{t=1}^{T}(100 - \delta_t) \quad \cdots\cdots (3\text{-}1)$$

위 수식 (3-1)에서 δ_t는 시간 t 구간의 단위 시간 동안 여러 헤드 노드들을 통해 측정된 센서 데이터 중 어플리케이션 질의에 부합하지 않는 데이터의 평균값이다. 네트워크를 그룹으로 나누어 질의를 처리하는 경우 하나의 헤드 노드가 처리하는 그룹을 g라하고 USN 네트워크의 모든 그룹을 $\{g_1, \cdots\cdots, g_n\}$과 같이 나타낸다. 하나의 그룹 g_i에 대해 Local 영역의 센서 데이터 값과 이웃한 $g_{i \pm n}$ 영역의 센서 데이터 값은 각각 m_i와 m_k로 표시한다. 이때 하나의 헤드 노드 g_i에 대하여 어플리케이션의 질의에 부합하지 않는 데이터의 량을 ϵ_i라 하고 수식(3-2)와 같이 정리 된다.

$$\epsilon_i = \frac{(|m_i - m_k|)}{m_i} \times 100 \quad \cdots\cdots (3\text{-}2)$$

또한 단위 시간 n동안 여러 헤드 노드들을 통해 수집된 센서 데이터가 어플리케이션 질의에 부합하지 않는 값을 δ_t라 하여 식 (3-3)과 같이 정리 된다.

$$\delta_t = \frac{1}{n}\sum_{i=1}^{n}\epsilon_i \quad \cdots\cdots (3\text{-}3)$$

이 방법은 데이터의 변화량이 AVRQ의 높고 낮음에 따라 허용 오차 이상일 경우에만 전송하는 효과적인 방법으로 실시간 상황인식 서비스의 QoS를 보장할 뿐만 아니라 네트워크를 통해 전송될 정보의 량을 병합과 대표의사 추출이라는 절차를 통해 대폭 줄일 수 있다.

3.3 상황정보관리 컴포넌트의 동작

GPMC는 다단계 별 상황정보 우선순위 정의를 통해 각각

의 상황정보관리 QoS를 보장한다.

"다단계 우선순위 분류" 이하 MLQ(Multi Layered Queue)는 수많은 센서 노드들로부터 수집되는 다양한 센서 데이터 중 각 센서 데이터의 종류에 따른 우선순위 항목들로 구성되어 있는 기준 표에 입각하여 MLQ에 입력된다. MLQ는 각각의 대기 큐 중 우선순위가 높은 정보에 대하여 네트워크의 수용량에 따라 가변적으로 선택된 범위의 AVR을 추출하게 된다. 또한 우선순위가 낮은 MLQ 센서 데이터들의 무한 기아 상태를 회피할 수 있도록 다단계 피드백 큐(Multi Level Feedback Queue)로 구성 된다.

우선순위 별로 각 단계에 들어가게 되는 센서 데이터들은 정책적으로 작성된 상황정보관리 서비스의 Sensor Data 우선순위 기준 표에 입각하여 가장 효율적으로 피드백 큐에 입력된다. 그림 7은 MLQ의 동작을 나타내고 있다.

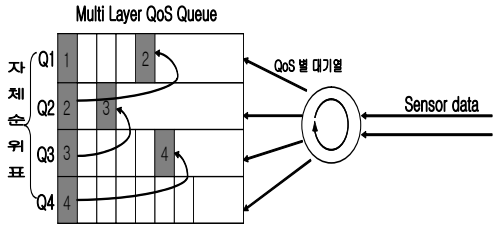

그림 7. MLQ 피드백 동작 과정
Fig 7. MLQ feedback action process

MLQ의 항목별 우선순위는 여러 USN 응용분야 및 네트워크상의 응용에 적용 할 수 있어야 함으로 Case by Case의 능동적인 구성이 가능해야 하며 표 1은 USN 분야의 한 파트인 USN 홈-네트워크에 적용 될 때의 Sensor Data들의 우선순위 기준 표를 작성한 예로서 홈-네트워크에서 발생할 수 있는 몇 가지 상황에 대한 항목별 우선순위를 기술한 것이다.

표 1. 상황인식을 위한 Level별 우선순위 예
Table 1. Priority example for Context-Aware

CASE 1	화재	Level 1
항 목	온도, 연기, 가스유출, 광량	
CASE 2	불법침입	Level 2
항 목	도어락, 유리창, 소리	
CASE 3	지진	Level 3
항 목	압력, 진동	
CASE 4	전기합선	Level 4
항 목	전류량, 전압	

IV. 실험

본 논문에서 설계한 GPMC 플랫폼의 성능을 평가하기 위하여 다음과 같은 환경과 방법을 이용해 실험해 보았다. 이 실험을 위해서 NS-2(Network Simulator 2)를 이용하였고 네트워크 구성은 헤드노드 6개와 각각의 헤드노드에 센서 노드 20개를 Sub 링크로 설정하였다. 센서는 이동이 없는 고정형태이고 각 센서의 버퍼 크기는 200패킷으로 설정하였다. 또한 패킷의 길이는 250byte로 가정하였다. 이러한 가정은 실제 센서 네트워크를 구성하는 미들웨어 동작 플랫폼의 대상인 Tiny-OS의 구동 버퍼와 수신 패킷의 크기에 기인한 것이다[9].

4.1 AVR 최적 추출 범위 실험

AVR 추출 시 몇 개의 헤드노드 영역까지 이웃 노드의 바운더리로 설정 할 것인지의 효율에 대한 실험을 진행한다.

AVR 최적 추출 범위 실험은 이벤트가 발생한 노드의 Local Sensor Data 이외에 이웃한 노드들로부터 수집된 센서 데이터를 병합하는 과정의 일환으로 이웃 노드의 범위를 몇 Hop 까지 병합해야 하는지의 바운더리를 찾는 실험으로 진행된다.

NS-2에서 AVR 최적 추출 범위 실험의 성능 분석을 위한 시나리오는 전송 데이터의 신뢰성 향상(AVRQ)에 역점을 둔 실험으로 각 헤드노드별 병합된 데이터의 AVRQ 측정을 통해 이상적인 병합 바운더리를 추출할 수 있는 시뮬레이션을 2회에 거쳐 수행한다. 그림 8은 AVR 최적 추출 범 위 실험에 대한 결과를 나타낸다.

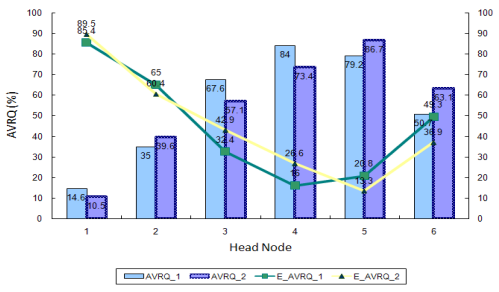

그림 8. GPMC의 AVR 최적 추출 범위 실험
Fig 8. AVR range of optimal extraction experiments of GPMC

실험 결과를 보면 센서 데이터를 병합하여 처리할 헤드노드의 증가에 따라 AVR 품질 측정 결과의 적합도(AVRQ)와 비적합도(E_AVRQ)의 변화를 확인할 수 있다.

실험 결과를 통해 AVR을 추출 하는 헤드노드의 병합 바운더리의 개수가 비교적 적은 3 헤드노드 까지는 평균 65%의 적합도를 보이지만 4~5개의 헤드노드 데이터를 병합하였을 때에는 90%에 가까운 최적의 AVRQ를 보장함을 보았고 이후 병합 바운더리가 증가하면 더 이상의 성능 향상을 기대할 수 없음을 확인하였다. 이 같은 결과로 그림 5에 도식화 돼 있는 모듈 중 Network Interface 모듈을 통해 각 미들웨어 노드들이 서로가 가진 센싱 정보를 수합하여 AVR을 추출 할 때 이웃한 노드의 수가 5노드 범위를 벗어나게 되면 AVR 품질이 저하된다는 것을 말하며 실례로 화재가 발생되었을 경우 발화점에서 원거리에 있는 센서들의 값은 의미가 없음을 알 수 있다.

4.2 MLQ 항목별 최적 큐 크기 추출 실험

두 번째 실험은 실시간 상황인식 QoS 보장을 위한 "MLQ 각각의 항목별 최적 큐 크기 추출"을 위한 실험을 진행한다.

상기의 실험 시나리오 역시 NS-2 환경에서 진행되며, AVR 최적 추출 범위 실험의 시나리오와 동일한 환경 하에 진행된다. 이때 그림 8의 실험 결과 데이터를 기반으로 AVR 추출 바운더리는 4로 설정하였다.

실험 절차는 센서 노드로부터의 응답 시간과 전체 센서 네트워크에 유포된 메시지 비율을 측정하고 표1의 우선순위 Case 종류별로 구성되어있는 단계별 큐의 메시지 수용력을 변경해가며 각각의 상황에 맞는 전송 지연을 추출하여 다단계 큐의 최적 크기를 분석한다. 그림 9는 MLQ 각각의 Queue 사이즈별 지연 시간의 상관관계를 실험한 결과를 나타낸다.

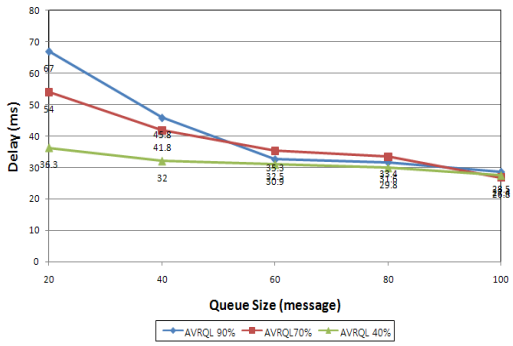

그림 9. MLQ 항목별 최적 Queue 크기 추출 범위
Fig 9. Queue size of the optimal extraction MLQ

그림 9는 MLQ의 큐 크기에 따라 Overflow가 발생하는 경우 패킷 오류율이 증가하게 되고 헤드노드들 간 지연 및 USN 응용 서비스까지 전달되는 지연이 증가하는 측정 데이터에 관한 그래프이다. 이때 4개의 헤드노드의 "병합된 측정 결과 품질 단계"(AVRQL : Quality Level of AVR) 값을 임의로 40%, 70%, 90%로 설정하여 동일 환경에서 시뮬레이션 해보았다.

실험결과를 분석해보면 각각 AVRQL 90%의 경우 큐 사이즈에 따라 급한 지연 변동량을 확인할 수 있지만 AVRQL 70%, AVRQL 40%로 AVR의 품질이 점점 낮아질 때는 MLQ의 큐 대기열의 크기 60을 기점으로 변동되는 지연량이 거의 미미한 것으로 나타나고 있다. 이는 AVRQL의 값에 따라 특정 단계의 큐에 부하가 집중 되거나 여라 다단계 큐로 고르게 분포되느냐의 상태에 따라서 지연 발생의 차가 나타나기 때문이다. 결과적으로 AVRQ 90%의 경우 큐의 사이즈가 20으로 큰 지연을 갖고 전송을 시작하지만 점차 큐의 사이즈를 크게 설정하여 메시지 Overflow 발생이 줄면서 일정 수준의 전송 지연 값을 유지하게 된다.

본 실험을 통해 제안하는 GPMC의 시뮬레이션 환경에서 MLQ를 이용한 전송 알고리즘 효율을 보장하기 위해서는 다단계 큐 사이즈를 60 전후로 설정하는 것이 가장 효율적임을 볼 수 있었다.

4.3 노드 수에 따른 에너지 소비량 측정 실험

그림 10은 GPMC와 Cougar, MiLAN의 노드 수에 따른 에너지 소비량 측정 결과이다.

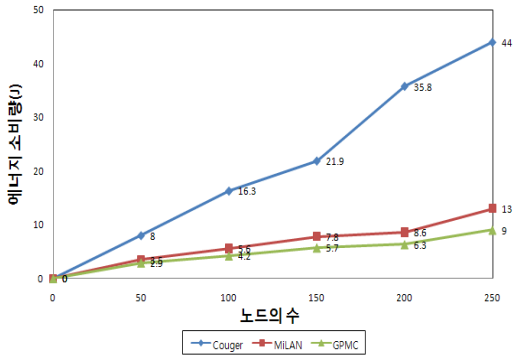

그림 10. GPMC와 Cougar, MiLAN의 노드 수에 따른 에너지 소비량
Fig 10. GPMC, Cougar, MiLAN energy consumption according to the number of nodes

Cougar는 전체 노드들에서 발생된 센싱 데이터를 응용서비스가 요구한 상황정보 인식을 위해 서버로 통합 되도록 일괄 전송한다. 이처럼 전체 노드들의 모든 센싱 데이터 병합과정에서 병합 데이터 전송 및 수신에 따른 에너지 소비량은 노드의 수에 비례한 에너지 소비가 발생한다.

MiLAN은 프로토콜의 추가 옵션(Add-Option) 및 명령(Open-API)을 통하여 데이터를 전달하기 때문에 센서 노드의 수에 따른 에너지 소비량에는 거의 변화가 없지만 USN 응용서비스의 질의와 일치하는 데이터는 모두 수신하고 어떤 최적화 및 병합 절차 없이 이웃 노드에 전송한다. 이러한 데이터 전송 기법은 동일한 값의 데이터를 다중 경로를 통해서 전송될 수 있는 가능성을 내포하기 때문에 데이터 중복 전송 문제가 발생할 수 있다. 데이터 중복 전송은 네트워크 전체의 에너지 낭비를 초래한다. 반면, GPMC는 데이터 병합 과정을 통해 중복 전송을 제거하여 에너지 효율 면에서 비교군보다 좋은 성능을 보였다.

4.4 응답 시간 측정 실험

두 번째 실험은 제안된 데이터 병합 및 전송 프로토콜과 기존의 프로토콜을 비교하여 센서 필드에서 획득한 데이터의 응답시간을 측정하였다. 그림 11은 제안된 데이터 병합 프로토콜이 Cougar보다는 노드 수의 증가에 따라 응답 시간이 빠르고 MiLAN과는 유사한 형태의 측정 데이터 그래프이다.

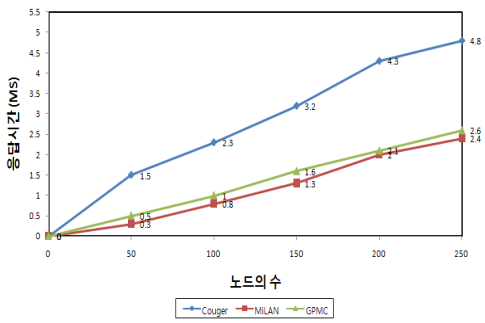

그림 11. GPMC와 Cougar, MiLAN의 센서 필드에서 획득한 데이터 응답시간 성능 비교
Fig 11. Comparison of acquired data response time between GPMC, Cougar and MiLAN

제안한 GPMC는 병합 과정을 수행하기 때문에 응답시간이 MiLAN보다 조금 느리게 나타난다. 이는 MiLAN이 데이터 병합 과정 없이 획득한 데이터를 바로 전달하므로 응답 시간이 비교적 빠르기 때문이다. 그러나 제안된 데이터 병합 프로토콜이 이벤트 주위의 노드들만 신속한 병합을 통하여 에너지 낭비와 중복 데이터 전송 횟수를 최소화할 수 있음을 앞절의 그림 10에서 확인 하였듯, 정확한 데이터의 전송과 에너지 효율적인 효과 면에서는 비교 대상에 비하여 성능이 우수함을 반증한다.

4.5 미들웨어 성능측정 및 평가 실험

제안하는 GPMC 미들웨어의 범용 지원 여부를 평가하기 위해 두 가지의 평가 요소들을 사용한다.

첫째, 센서 네트워크상의 데이터 전송 흐름제어 성능 평가를 진행한다. 전송 흐름제어 성능 평가는 USN 네트워크의 전체 데이터 전송량과 헤드노드의 수신율을 기준으로 네트워크 지연을 해소하는 성능에 대하여 평가한다. 둘째, 헤드노드 수신 데이터 수렴도에 대한 성능을 평가한다.

4.5.1 데이터 전송 흐름제어 성능 평가

그림 12는 첫 번째 실험인 데이터 전송 흐름제어 성능 평가 결과를 나타낸다. 전송 흐름제어의 성능 평가를 위해 시뮬레이션 환경에서 초기 네트워크 리소스 대역폭을 30%이하로 설정하고 실험을 진행한다.

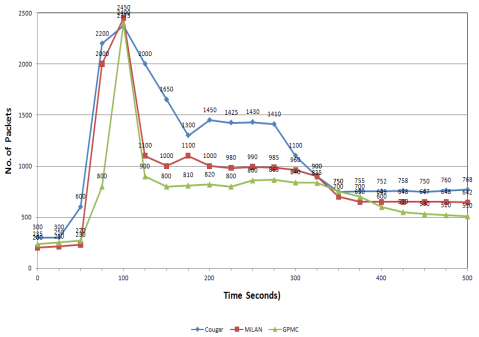

그림 12. 데이터 전송 흐름제어 성능 평가
Fig 12. Performance evaluation of data flow control

단위시간 100을 경유하여 네트워크의 리소스를 전체 가용량의 80% 이상으로 급격히 상승시켰다. 이때 네트워크 리소스 대비 패킷 생성량 증가로 인해 네트워크 혼잡이 탐지되고 이에 따른 조치로 각각의 미들웨어 기법들은 전송 흐름제어 알고리즘을 수행한다.

MiLAN의 경우 혼잡이 탐지되어 혼잡 발생을 하위로 통보하면 하위 노드들이 AIMD(Additive Increase Multiplicative Decrease)기법을 적용하여 패킷 생성량을 줄이기 때문에 패킷 생성량이 급격히 줄어드는 것을 볼 수 있다. 또한

Cougar의 경우 TCP-Reno 방식으로 패킷 전송량을 조절하여 패킷 손실 점을 기준으로 전송량을 절반으로 조율한 후 Linear Increment 방식으로 패킷 전송량을 조절하는 것을 확인 할 수 있다. 이에 반해 GPMC에서는 혼잡 발생 시 적응적 흐름제어를 통하여 우선순위에 입각한 긴급 데이터를 선출하고 전송간격을 조절하기 때문에 혼잡 발생에 따른 전송 지연이 비교 대상들에 비해 천천히 전파되어, 생성되는 패킷 량이 서서히 감소하는 것을 확인할 수 있다.

4.5.2 수신 데이터 수렴도 평가 실험

혼잡발생시 패킷이 손실되는 것을 줄이기 위해서는 근본적으로 센서 노드들이 생성하는 패킷의 양을 줄여야 한다. MiLAN과 Cougar의 경우와 같은 기존의 혼잡제어 기법들에서는 혼잡 발생 시 데이터 샘플링 주기를 늘리는 방식을 주로 사용하고 있다.

데이터 샘플링 간격이 늘어나면 시간당 생성되는 데이터의 양이 줄고 따라서 생성되는 패킷양도 감소하는 결과가 있을 수 있다. 하지만 샘플링 주기를 조절하면 시간당 생성되는 데이터의 양이 줄기 때문에 시간당 고정적인 샘플링 주기를 필요로 하는 응용에 적용하기 어려운 문제점이 있다. GPMC는 압축 기법을 적용하여 데이터의 시간적 손실을 제거한다.

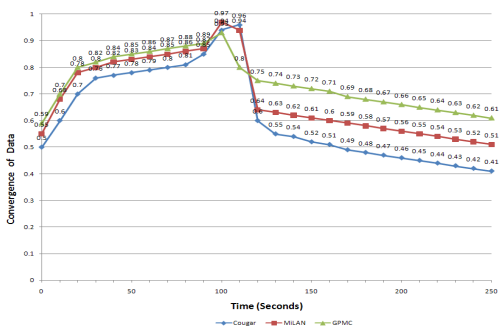

그림 13. 수신 데이터 수렴도 평가
Fig 13. Evaluation of the incoming data convergence

그림 13은 각각의 미들웨어를 통해 수신한 패킷에서 추출한 이용 가능한 데이터의 양을 보여준다. MiLAN과 Cougar, GPMC 모두 혼잡이 발생하면서 패킷이 손실되기 때문에 이용 가능한 데이터의 양이 급격히 감소하는 것을 볼 수 있다. 하지만 시간이 지나면서 지속적으로 데이터의 양이 감소하는 Cougar에 비해 ADMI 기법을 사용하는 MiLAN은 비교적 그 손실이 적음을 볼 수 있다. 이러한 결과는 Cougar에 비해 MiLAN이 보다 점차적으로 네트워크 적응적인 흐름제어를 수행하는 결과이다. 반면 MiLAN과 GPMC의 데이터 수렴도를 분석해보면 동일한 네트워크 리소스를 사용하는 환경에서 GPMC가 약 10% 이상 일정한 수준으로 수렴하는 것으로 볼 수 있으며 최종적으로 Cougar에 비해 거의 2배에 가까운 데이터를 보장함을 볼 수 있다.

V. 결론

범용 USN 미들웨어를 효과적으로 구현하기 위해서는 UNS 응용서비스의 상황 인식 효율을 높이기 위해 센서 노드로부터 수집된 긴급데이터를 실시간으로 전송할 수 있는 기법이 필요하다. 이는 센서 네트워크 자원과 환경이 동적으로 변화하는 상황에서 긴급 상황 발생 시 그 긴급 상황 정보에 적절한 긴급 데이터를 USN 응용서비스에 실시간으로 제공하여 상황인식 성능과 정확성을 보장해주는 USN 미들웨어의 기본 기능이다.

본 논문에서는 센서 네트워크 환경에서 수집된 데이터의 효율성을 향상 시켜 효과적인 병합 데이터 전송을 통해 USN 응용서비스의 실시간 상황인식 QoS를 보장 할 수 있는 범용적인 미들웨어 모델을 제안하였다.

GPMC 미들웨어의 구현을 위하여 본 논문에서는 데이터 전송 과정에서 에너지를 효율적으로 사용하고, USN 응용 서비스에 정확한 데이터를 빠르게 전달 할 수 있는 데이터 병합 및 전송 알고리즘에 대한 연구와 GPMC 성능을 최적화할 수 있는 방안에 대한 연구를 수행하였다.

AVR 최적 추출 범위 실험과 MLQ 각각의 Queue 항목별 최적 크기 추출 실험을 통하여 GPMC 자체의 최적 성능을 위한 임계값 및 처리 영역에 대한 결과 값을 산출하였다.

이후 그 성능을 기존의 Cougar와 MiLAN의 데이터 전송 알고리즘과 비교하였다.

이러한 실험 결과를 토대로 본 논문에서 제안한 GPMC 미들웨어 모델이 다른 여타의 USN 미들웨어보다 센서 노드별 데이터 수집 효율성이 높아 USN 응용서비스의 실시간 상황인식 성능에 조력도가 높음을 확인할 수 있으며, Open-API 지향 구조 및 in-network 구조를 포함함으로 범용적인 USN 환경에 적합하다고 할 수 있다.

향후 본 논문에서 제안 하는 GPMC 미들웨어 모델을 범용 센서 네트워크 응용에 원활히 활용하기 위해서 이웃 노드들 간의 연결성 위주의 병합을 고려한 다양한 센서 네트워크 응용에 활용 가능한 방법에 대하여 추가적인 연구가 필요하다.

참고문헌

〔1〕 김민수, 김광수, 이용준, "USN 미들웨어의 특징 및 기술 개발 동향," IITA, 주간기술동향. 통권 1284호, 2007.2.

〔2〕 최재원, 이광휘, "무선 센서 네트워크에서 신뢰성 있는 데이터 전달 기법," 한국인터넷정보학회, 제 7권, 제 1호, 25-28쪽, 2006년 4월.

〔3〕 Y. Yao and J. Gehrke, "The Cougar Approach to In Network Query Processing in Sensor Networks," SIGMOD Record, Vol. 31, No. 3, 2002.

〔4〕 A. Murphy and W. Heinzelman, "MiLAN: Middleware Linking Applications and Networks," TR-795, University of Rochester, Computer Science, 2002.

〔5〕 정현준, 김주일, 길아라, 정기원, 이우진, "속성 기반의 USN 센서 노드 동적 재구성 기법," 한국정보과학회 2008 종합학술대회 논문집 제35권 제1호(D), 2008.

〔6〕 이상학, 조위덕, 정태충, "무선 센서네트워크의 네트워킹 기술," 정보통신 기술 제 18권, 제 1호, 2004년 5월.

〔7〕 A. Boulis, C. C. Han, and M. B Srivastava, "Design and Implementation of a Framework for Efficient and Programmable Sensor Networks," In MobiSys, San Francisco, USA, 2003.

〔8〕 M. Sharaf, J. Beaver, A. Labrinidis, and P. K Chrysanthis, "TiNA: A Schme for Temporal Coherency-Aware in-Network Aggregation," Proc. of the 3 ACM international workshop on Data engineering for wireless and mobile access, Sep. 2003, pp. 69-76

〔9〕 김승천, "유비쿼터스 센서 네트워크의 신뢰성 확보를 위한 데이터 전송 방법의 성능 비교," 전자공학논문지, 제 45권, 제 6호, 45-51쪽, 2008년 11월.

저 자 소 개

김 광 훈

2009 : 충북대학교 공학 석사
2009 ~ 현재 : 충북대학교 컴퓨터공학 박사 과정
관심분야 : 유비쿼터스, 멀티미디어통신 기술, USN 관련 기술

최 운 수

2007 : 충북대학교 공학 석사
2009 : 충북대학교 컴퓨터공학전공 공학 박사 수료
2010 ~ 현재 : 공주대학교 강사
관심분야 : 유비쿼터스컴퓨팅, 정보보안, 멀티미디어통신

이 태 우

2004 ~ 2008 : (주)LG전자 단말연구소 선임연구원
2008 ~ 현재 : 충북대학교 강사
관심분야 : 멀티미디어 통신, 유비쿼터스 센서 네트워킹, 임베디드 시스템

이 준 석

2007 ~ 2009 : 영동대학교 강사
2008 ~ 현재 : 충북대학교 강사
관심분야 : 멀티미디어 통신, HCI, 엔터테인먼트 기반기술

구 경 옥

충북대학교 대학원 컴퓨터공학과 컴퓨터공학전공 (공학박사)
한양대학교 산업대학원 산업공학과 전자계산전공 (공학석사)
현재 : 강릉영동대학 사무정보과 부교수
관심분야 : 유비쿼터스 센서 네트워킹, 멀티미디어 통신, 네트워크 보안

조 용 환

1982년 ~ 현재 : 충북대학교 전자정보대학 교수
관심분야 : USN, 멀티미디어 통신, 정보통신정책, 멀티미디어 콘텐츠, 엔터테인먼트 기반 기술