센서네트워크 응용 개발을 위한 네트워킹 지원 구조

최석원° 차호정
연세대학교 컴퓨터과학과
{sukwon°, hjcha}@cs.yonsei.ac.kr

Networking Support Framework for Wireless Sensor Network Applications

Sukwon Choi[0], Hojung Cha
Department of Computer Science, Yonsei University

요 약

본 논문에서는 WSN의 개발자를 커널 개발자, 네트워크 응용 개발자로 분리하고 이들이 상호 배타적으로 프로그래밍 할 수 있는 네트워크 구조를 제시 한다. 이를 위해 커널을 정적 부분과 동적 부분으로 분류하고 응용에 따라 변경되는 부분을 최소화하면서도, 네트워크 모듈의 재사용성을 높일 수 있는 효율적인 센서네트워크 프로그래밍 환경을 제공한다. 이를 위해 MAC 프로토콜과 독립적으로 네트워크 알고리즘을 개발 할 수 있는 새로운 계층 구조를 제안한다. 제안하는 네트워크 구조는 상하 독립적 프로그래밍 구조를 제공함으로써 hop-to-hop통신과 end-to-end통신의 개발을 별도의 알고리즘으로 할 수 있도록 한다. 유효성을 검증하기 위해 Dynamic Loadable Kernel Module을 지원하는 RETOS 운영체제를 이용하여 다양한 네트워크 알고리즘 및 응용을 구현한 후, 그 성능을 평가하였다.

1. 서론

WSN(Wireless Sensor Network)은 응용 및 적용 분야에 따라 최적화된 네트워크 프로토콜을 요구하기 때문에 응용의 다양화에 맞추어 각종 네트워크 프로토콜이 개발되었다. 이에 따라 이를 운영체제 관점에서 다양한 프로토콜을 효율적으로 유지하기 위한 방법이 필요하게 되었으며, 시스템 추상화 및 Layering 그리고 Modularity 대한 연구가 활발하게 진행되고 있다. 그 결과 WSN최적화된 H/W 추상화 구조[1]가 제안되거나, RETOS[2] 등과 같이 동적으로 커널의 일부분을 변경할 수 있는 운영체제가 등장하였다. RETOS는 기존의 General Purpose(GP)응용체제와 같이 커널과 응용을 구조적으로 분리할 뿐 아니라, Dynamic Loadable Kernel Module(DLKM)을 지원함으로써 기존 센서 운영체제보다 진보된 응용개발환경을 제시하고 있다. 현재 센서네트워크의 개발환경은 MAC에서 네트워크 알고리즘, 그리고 응용까지 한 명의 개발자가 전체를 디자인해서 개발 및 배포하고 있다. 때문에 DLKM을 지원하는 운영체제를 이용함에도 불구하고 개발자 마다 다른 형태로 모듈구조를 유지하게 되고, 이로 인해 시스템의 효율성 및 모듈의 재사용성이 저하된다. 적절한 시스템 Layer구조, 특히 네트워크 프로토콜 개발을 위한 Layer구조가 정의되면 모듈구조가 가지는 장점을 최대한 보장할 수 있다.

WSN는 한정된 리소스를 가진 하드웨어를 이용하고, Ad-hoc multi-hop 을 기반으로 통신하고, 응용 종속적이라는 특성이 있다. 이러한 특징으로 기존 네트워크와 같이 분리된 프로토콜 구조를 가지지 못하고 cross-layer형태의 응용종속적인 단일구조를 가지게 되었다. 분리된 네트워크 구조를 제시하기 위하여 SP(Sensornet Protocol)[3] 와 같은translucent link abstraction 가 제안되거나, [4]와 같이 data-centric응용을 위한 data fusion 구조의 네트워크 계층화도 제안되었다. 이러한 연구들은 네트워크 프로토콜의 부분적인 독립은 가능하게 하였으나, 다양한 응용 및 네트워크 프로토콜의 개발이 가능한 독립적 구조의 네트워크 구조화에는 미흡하였다.

새로운 네트워크 구조는 WSN을 위한 네트워크 구조는 각 계층의 개발주체의 관점에서 계층의 역할을 결정하고 이에 맞는 특징적인 구조를 가져야 한다. 이는 그림 1에서와 같이WSN

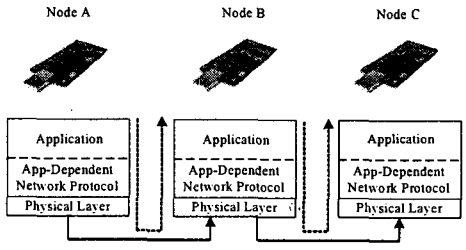


그림 1. 무선 센서노드의 프로토콜 스택 구조

환경 에서는 각각의 노드가 Router역할을 하기 때문에 데이터 전송에 네트워크 프로그램 전체가 관여하고 데이터 전송을 위한 multi-hop 통신뿐 아니라 In-Network Processing을 위한 주변 node와의 hop-to-hop통신을 요구한다. 따라서 기본적인 hop-to-hop 통신은 하위 계층에서 처리하도록 하고, 상위계층은 사용자는 목적지 지정을 위한 end-to-end통신을 처리하기 위한 알고리즘을 구현 할 수 있어야 한다.

이런 관점에서 본 논문에서는 WSN의 개발자를 커널 개발자, 네트워크 응용 개발자로 구분 하고, 커널 개발자는 H/W를 최대한의 성능으로 동작할 수 있도록 hop-to-hop통신을 위한 주요 부분을 운영체제 입장에서 프로그래밍하는 역할을 담당하고, 네트워크 응용개발자는 다양한 end-to-end통신 알고리즘을 커널 및 응용 독립적으로 개발하여, 원하는 기능의 응용을 개발을 보장하는 네트워크 구조를 제안한다. 논문에서 제안하는 구조는 커널과 응용이 분리된 구조의 센서 운영체제위에서 다양한 네트워크 프로그램을 작성 할 수 있는 네트워킹 지원 구조를 제공하고, 적절한 구조의 독립적 패킷과 커널 API를 통해서 개발자가 원하는 네트워크 알고리즘 및 응용을 구현 할 수 있도록 도와준다.

2. 네트워크 지원 구조

네트워크 구조의 계층화는 유연한 구조의 개발을 가능하게 한다는 장점도 있지만 지나친 계층의 증가로 인해 시스템 오버헤드가 증가하고, 개발자 부담을 가중시킬 수 있다는 단점이 있다. 센서네트워크의 경우 시스템 리소스의 제약으로 인해 이러한 단점이 더욱 치명적이다. 논문에서는 개발자의 관점에서 네트워크 계층 을 성능이 중요한 부분과 유연성이 중요한 부분으로 구분하였다. 성능이 중요한 부분은 하드웨어와 밀접한 부분으

• 본 연구는 한국과학재단에서 지원하는 국가지정연구실사업으로 수행하였음 (과제번호: 2005-01352)

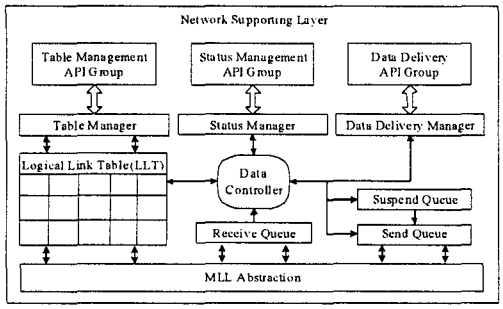
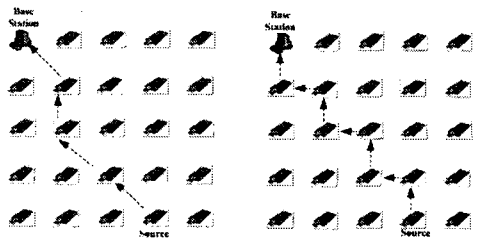


그림 2. Networking Support Layer

로써 커널 개발자가 디바이스 드라이버 레벨에서 최적화된 커널 형태로 개발하여야 한다. 유연성이 중요한 부분은 네트워크 응용 부분으로 세분화 할 수 있다. 네트워크 응용 부분은 전문적인 네트워크 지식을 통해서 주어진 환경에 최적화된 프로그래밍이 가능해야 하며, 센서노드를 배치하고 사용하기 위해서 수집하고자 하는 데이터 중심으로 프로그램 가능해야 한다. 이는 각 하위 네트워킹 개발자와 응용개발자가 자신의 전문기술에 맞추어 이루어 져야 한다. 이를 위해 논문에서는 정적 커널의 역할과 동적 커널의 역할을 분리하였다. 정적 커널에서의 네트워크 기능은 이웃노드의 연결 유지 및 데이터 전송이며, 동적 커널 부분은 네트워킹 알고리즘 구현 및 응용개발을 도와주기 위한 부분으로 기능적 분리를 하였다. 특히 정적부분을 MLL(MAC and Data Link Layer), NSL(Networking Support Layer)의 2단으로 계층을 분리하였다. MLL은 네트워크 구조의 가장 아래층으로써 네트워크 장치를 제어하고 물리적인 연결 및 데이터 전송하는 역할을 담당하고, NSL은 이웃노드를 관리 및 데이터 송수신을 위한 매커니즘을 제공하여 논리적인 연결을 유지한다. 이러한 구조를 이용하면 네트워크 응용 개발자가 자신이 할 수 있는 최적의 알고리즘을 커널 독립적으로 개발 할 수있다. 제안하는 네트워크 구조 는 모듈화된 센서 운영체제가 가지는 커널 독립적 응용 배포의 기능을 최대한 이용할 수 있는 구조이다. 다음은 NSL의 구체적인 설계를 기술한다.

NSL의 구체적 역할은 MLL 위에서 MLL에서 이루어지는 물리적 데이터 교환과 관련된 정보와 이웃노드 정보를 유지하여 응용에서 구현된 네트워크 알고리즘에 따라 결정된 목적지 node에 데이터를 전송하는 것이다. NSL은 그림 2과 같이 Table Manager, Status Manager, Data Delivery Manager, 그리고 NSL API 4가지 요소로 구성되어 있다. Table Manager 는 이웃노드에 대한 정보를 가진 Logical Link Table(LLT)를 정해진 알고리즘에 맞추어 유지시켜준다. 유지되는 테이블 정보는 real-time, energy efficiency, reliability를 보장할 수 있는 노드간의 정보이다. LLT를 유지하기 SPIN[5] 프로 토콜을 변형하여 ADV-REQ 2way handshake 프로토콜을 사용하였으며, Update 주기는 네트워크 프로토콜 개발자가 네트워크 프로토콜 알고리즘 정책에 의해 정해 질 수 있도록 주기적/비주기적 Update API를 제공한다. Data Delivery Manager 는 데이터 송수신을 위한 Send/ Suspend/Receive Queue를 관리한다. 제안된 구조에서는 개별 node가 라우터 역할을 할 수 있어야 하기 때문에 Send/Receive Queue외에 Suspend Queue가 추가되었다. Suspend Queue는 수신된 패킷의 최종 목적지 주소를 판별하여 현재 node가 routing을 위한 node로 판명되면 다음 목적지가 결정될 때까지 대기하는 공간이다. 이를 이용하면 데이터 송수신 시간을 적절히 조절 할 수 있기 때문에 혼잡상황 회피 정책 및 모니터링이 가능 하다. Status Manager는 MLL에서 들어오는 정보와 지정된 이웃노드 정보, 그리고 네트워크 Queue에



(a) Reliable Network Protocol (b) Real-time Network Protocol

그림 3. One-hop data transmission

서 결정된 상황을 NSL에 반영하는 역할을 하는 부분이다. 또한 DNL정책에 맞추어 수신된 패킷을 Suspend Queue로 전달하거나 곧바로 전송해주는 역할도 한다.

NSL API는 응용 프로그램과 NSL의 인터페이스 역할을 해주는 부분으로 3개의 매니저를 컨트롤 할 수 있도록 Table Manage- ment API Group, Status Monitoring API Group, Data Delivery API Group으로 구성되어 있다. 각각의API는 커널 Function으로 지정하여 Dynamic Kernel Module에서 불러 쓸 수 있도록 한다.

3. 실험

제안하는 네트워크 구조를 Moteiv사의Tmote Sky[6] 센서노드를 사용하는 RETOS 운영체제 커널에 구현 하였다. 커널에 MLL 및 NSL을 구현하고, 몇가지 네트워크 알고리즘을 사용한 응을 구현하여 제안된 네트워크 구조의 유효성을 판단하였다.

Tmote Sky[6]는 Chipcon사의 CC2420 트랜시버를 사용한다. CC2420는 IEEE 802.15.4의 호환 frame은 데이터 처리를 패킷단위로 하기 때문에 S/W로 제어할 수 있는 부분이 적기 때문에 다양한 MAC의 구현이 힘들다. 하지만 중요 부분에 대한 인터페이스를 제공 할 뿐 아니라, H/W에서 많은 부분을 처리해 주기 때문에 MLL을 최소화 할 수 있다. 또한 CCA, LQI등 NSL 구현을 위한 요소들을 기본적으로 가지기 때문에 S-MAC이나 B-MAC, Z-MAC 같은 S/W로 구현한 센서 MAC에 비해 더 효율적인 MLL을 구현 할 수 있다. 본 논문에서 이러한 이유와 더불어 향후 센서네트워크의 기본적인 MAC프로토콜이 H/W로 구현 될 것임을 예측하고 MLL을 CC2420기반의 Radio transiver를 기초로 하여 구현하였다. MLL의 가장 중요한 역할은 NSL과의 연결이기 때문에 RETOS의 Device Driver Architecture(R-DDA) 구조를 이용하여 MLL을 최적의 성능으로 구현하였다.

MLL과 NSL의 모두 정적커널에서 개발되지만 각 layer의 독립성을 보장할 수 있어야 한다. MLL은 하위 H/W와 밀접한 관계를 가지고 있기 때문에 Device 종속적인 형태로 개발되어야 하고, NSL은 이웃노드 및 송수신 데이터 관리를 위한 시스템 구조 이기 때문에 Device 독립적인 형태로 개발되어야 한다. 본 논문에서는 이러한 구조적 원칙하에 제안된 NSL구조를 RETOS 운영체제 커널 안에 Neighbor Table, Network Queue, API를 위한 자료 구조를 Device 드라이버와는 독립적인 구조를 가지도록 개발하였다.

제안된 네트워크 구조의 성능평가를 위하여 2가지 요소를 측정하였다. 첫째는 네트워크 알고리즘에 따른 적응성 둘째는 성능 이다. 이를 위해 개발된 MLL과 NSL위에 환경 모니터링 어플리케이션인 Surge를 구현하고, 응용에 사용되는 네트워크 알고리즘을 변환하여 2가지 버전의 RETOS-Surge구현 하였다. 그림 3는 네트워크 알고리즘에 따른 적응성을 판별하기 위하여 RETOS-Surge에 서로 다른 네트워크 프로그램을 적용하여 5X5로 배열된 센서 필드의 데이터 전송경로를 추적한 결과이다. 사

용한 네트워크 알고리즘은 Reliable Network Protocol과 Real-time Network Protocol 두 가지 이다. 그림에서 볼 수 있듯이 NSL의 수정 없이 네트워크 알고리즘만 교체하여 다른 경로로 데이터가 전송 될 수 있음을 확인하였다.

성능저하 및 코드 효율성을 확인하기 위하여 RETOS-Surge 응용을 NSL과 DNL을 사용한 네트워크 구조와 사용하지 않은 구조를 이용하여 One-hop 및 Multi-hop 토폴로지에서 데이터 전송시간을 측정함으로써 오버헤드를 체크하였다. One-hop의 데이터 전송은 제안된 구조가 가지는 Overhead 및 안정성을 확인하기 위한 실험이다. 그림 4의 결과에서 볼 수 있듯이 NSL을 사용하지 않은 응용의 경우 데이터 전송시간의 변화폭이 큰 것을볼 수 있다. 그에 비해 제안된 네트워크 구조를 사용한 응용의 경우 안정된 전송시간을 유지할 뿐 아니라 좀더 빠른 전송을 하는 것을 알 수 있다. 이는 제안된 네트워크 구조를 사용하지 않은 데이터 전송은 네트워크 알고리즘과 응용이 하나로 동작하여 패킷 단위로 체크하고, 적절한 queue 제어 매커니즘이 없어 수신 노드의 현재 상황에 따라 불규칙한 데이터 전송을 하기 때문이다. 하지만 제안된 구조를 사용하는 경우 적절한 queue 제어 매커니즘과 node간 전송은 네트워크 알고리즘과 별도로 커널내 NSL에서만 이루어지기 때문에 빠르고 안정된 전송을 할 수 있다. 제안된 구조의 경우 Multi-hop 데이터를 전달하는 중간 node의 source 와 destination과 상관 없이 순수 relay만 해주기 때문에 네트워크 구조를 사용하지 않는 경우보다 데이터 전송이 빠르게 이루어 져야 한다. 이를 확인하기 위하여 Multi-hop 전송을 측정한 결과가 그림 5 이다. 제안된 구조를 사용한 응용의 데이터 전송의 속도가 제안된 구조를 사용하지 않는 데이터 전송보다 hop수를 증가할수록 더 빠르게 이루어졌다. 네트워크 구조를 사용하지 않는 경우 응용 전체가 다음 전송 node를 결정하기 위하여 패킷 전체를 확인한 후 전송하지만 제안된 구조를 사용한 경우 data 전달 후 바로 다음node가 결정된 상태에서 NSL에서 직접적으로 데이터가 전달되기 때문에 스위칭 등의 오버헤드가 없어 더 빠른 전송이 가능하게 되었다.

4. 결론

WSN 기술은 점점 발전하고, 다각화 되고 있다. 이로 인해 하드웨어도 점점 강력해지고 있어 기존의 센서네트워크 개발환경으로는 한계를 느끼게 되었다. 본 논문은 WSN 개발자 관점에서 개발의 효율성과 성능을 보장할 수 있는 네트워크 구조를 제시함으로써 새로운 센서네트워크 개발환경을 제시한다. 제시된 구조는 기존 범용운영체제에서 사용하는 Static Network Layer의 장점과 Sensor에서 요구하는 다양성을 모두 만족하기 위한 구조를 제시한다. 특히 각 계층간의 독립성을 보장함으로써 네트워크 프로토콜의 재사용 성을 높였으며, 빠르고 손쉬운 응용 개발 환경을 제공 할 수 있다. Dynamic Loadable Kernel Module(DLKM)을 지원하는 RETOS 커널을 tmote sky 센서노드에 이식하여 MLL 및 NSL 구현하고, 그 위에 네트워크 응용을 개발하여 제시된 구조의 타당성을 평가하였다. 그 결과 제시된 네트워크 구조를 이용하면 효율적인 네트워킹 프로그램이 가능하고 성능저하 없는 네트워킹이 가능함을 보였다. 이러한 결과를 통해 우리가 제안하는 네트워크 프로토콜 구조는 센서 네트워크에 적합하게 사용될 수 있으며, 특히 RETOS와 같이 모듈화 된 센서 운영체제에 적절한 프로그래밍 가이드라인을 제시할 수 있다.

향후 NSL을 위한 최적화된 이웃노드 관리 기법 및 Queue관리 기법을 개발하여, 제시된 네트워크 구조의 성능 향상을 꾀하며, 더불어 네트워크 알고리즘을 독립적으로 구현 할 수 있는 DLKM을 응용 및 NSL과 독립적으로 구현할 수 있는 상위계층의 네트워크 구조를 개발 하려 한다.

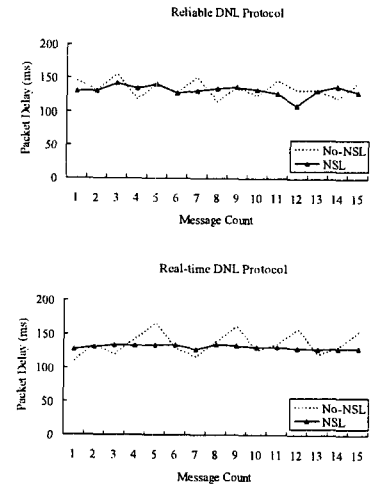


그림 4. One-hop data transmission

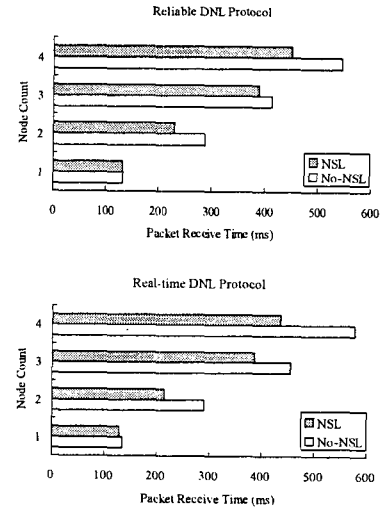


그림 5. Multi-hop data transmissions

Reference

[1] V. Handziski, J.Polastre, J.H.Hauer, C.Sharp, A.Wolisz and D.Culler, "Flexible Hardware Abstraction for Wireless Sensor Networks," *in Proceedings of the 2nd European Workshop on Wireless Sensor Networks (EWSN 2005)*, Istanbul, Turkey, January 2005.

[2] H. Kim and H. Cha, "Towards a Reliable Operating System for Wireless Sensor Networks,"To appear in the USENIX Annual Technical Conference: Systems Practice & Experience Track, Massachusetts, USA, May 2006.

[3] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, I. Stoica, "A Unifying Link Abstraction for Wireless Sensor Networks," *In Proceedings of the Third ACM Conference on Embedded Networked Sensor Systems (SenSys 2005)*, San Diego, USA November 2005.

[4] R. Kumar, Santashil PalChaudhuri , David Johnson , Umakishore Ramachandran' Network Stack Architecture for Future Sensors," Rice University, Computer Science, Technical Report, TR04-447

[5] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," *In* Proceedings of the 5th ACM/IEEE Mobicom Conference, Seattle, USA, August 1999.

[6] MoteIV Co. Ltd., http://www.moteiv.com/