

센서 네트워크 시뮬레이션

한국전자통신연구원 | 박상준 · 문영백 · 박종준

1. 서 론

센서네트워크는 유비쿼터스 센서네트워크(Ubiquitous sensor network) 혹은 무선센서네트워크(Wireless sensor network)를 통칭하는 말로, 연산능력과 무선 통신능력을 가진 소형의 센서들을 이용하여, 주변의 환경정보 또는 물체를 탐지 및 식별하여 각종 데이터들을 수집할 수 있도록 구성된 무선 통신망이다. 센서 네트워크를 구성하고 있는 센서노드들은 자원이 제한된 하드웨어 및 저전력으로 동작해야 하기 때문에, 노드의 전력소모 최소화 및 프로그램의 크기등을 고려해야만 한다. 대부분의 센서 네트워크 응용은 다량의 센서노드를 설치하기 때문에 설치 후 다시 회수하기 위해서는 많은 시간과 비용이 필요로하게 된다. 만일 사람이 접근하기 어려운 지역에 설치한 경우에는 회수가 불가능하게 된다. 비록 센서 노드는 성능이나 가격 측면에서 제약사항이 있어도 H/W 및 S/W들은 높은 신뢰성이 보장 되도록 설계 및 검증이 되어야한다. 또한 다량의 센서노드가 분포된 상황에서는 네트워크 간의 간섭이 많이 발생되기 때문에 이에 따라 발생될 수 있는 문제점들을 사전에 검증하여야 한다.

다수의 센서노드들이 네트워크로 구성된 센서네트워크를 개발하기 위해서는 네트워크 시뮬레이터를 이용하는 방법 또는 실제 센서노드로 구성된 Testbed를 이용하는 방법으로 분류된다. 본 논문에서는 이러한 두 가지 방법들의 종류들과 각각의 특징들에 대해 설명을 한다. PC상에서 수행하는 네트워크 시뮬레이터로는 NS-2[1,2], OPNET[3,4], QualNet[5], TOSSIM[6] 등이 널리 알려 있다. NS-2는 사용자가 많고 NS-2 환경에서 개발된 소스코드들을 쉽게 구할 수 있을 뿐만 아니라 무료로 제공되고 있어 학교를 중심으로 가장 널리 사용되고 있다. OPNET과 QualNet은 상용으로 판매되고 있는 대표적인 네트워크 시뮬레이션 장비로서 NS-2에 비해 고가이지만 다양한 라이브러리를 제공하기 때문에 산업체에서 널리 사용되고 있다. NS-2, OPNET, QualNet과 같은 네트워크 시뮬레이터

들은 처음부터 센서네트워크용으로 개발된 것이 아니고 센서네트워크를 지원하기 위해 기능들을 추가하고 있다. OPNET이나 QualNet은 센서네트워크용 802.15.4 MAC[6]과 ZigBee[7]용 라이브러리를 개발하고 있다. UCLA에서는 QualNet에 센서네트워크 기능을 지원하기 위해 SenQ 또는 sQualNet[8] 불리는 센서네트워크 용 라이브러리를 개발하였다. 마지막으로 TOSSIM[6]은 TinyOS mote를 네트워크 수준에서 시뮬레이션하기 위해 개발초기부터 센서네트워크용 시뮬레이터로 개발되었다. 제2장에서는 네트워크용 시뮬레이터들의 특징들에 대해 보다 자세히 설명을 한다. 네트워크 시뮬레이터와는 달리 실제 센서노드로 구성된 Testbed를 이용하여 네트워크 수준에서 개발 및 검증을 할 수가 있다. 제 3장에서는 오하이오 대학의 Kansei[9,10], Harvard 대학의 MoteLab[11] 및 UC 캠퍼스(UCLA, UCSB, UCR, UCD, UCSD)들과 USC 및 멜라웨어 대학으로 구성된 여러 대학이 연합으로 구축한 WHYNET[12]들에 대해 소개하고 각각의 특징들에 대해 설명을 한다.

2. PC 기반 네트워크 시뮬레이터

네트워크 시뮬레이터들은 동일한 조건을 계속 시험할 수 있는 뛰어난 반복성, 컴퓨터의 성능에 따라 센서노드의 수를 수천, 수만 개로 확장 할 수 있는 뛰어난 확장성 그리고 다양한 노드 배치 및 토플로지 구성 등이 자유로운 특징들을 제공한다. 그러나 네트워크 시뮬레이션에서는 실 환경과 같은 전파환경이나 센서들의 센싱 특성을 정확히 모델링하기가 어려워 시뮬레이션 결과에 한계가 있다. 이 장에서는 앞에서 언급한 NS-2, OPNET, QualNet과 같은 일반적인 네트워크 시뮬레이터들과 TOSSIM과 같은 센서 네트워크 노드에 특화된 시뮬레이터들에 대해 설명을 한다.

2.1 NS-2

NS-2는 Network Simulator의 약자로 NS-1¹⁾의 확

1) 콜롬비아 대학에 의해 개발된 시뮬레이션 테스트베드인 NEST를

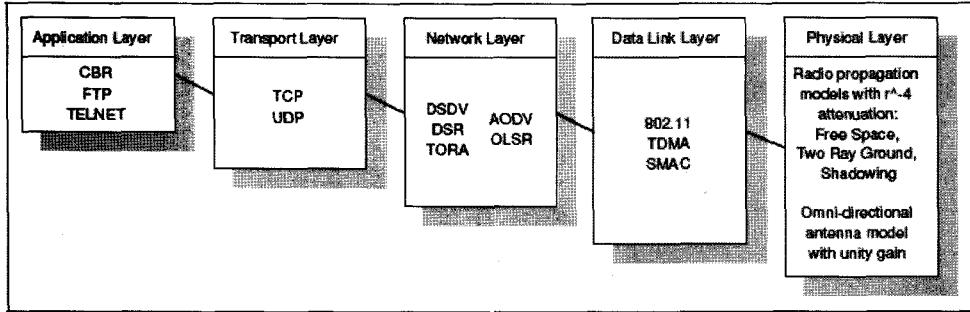


그림 1 현재 NS-2 상에서 가능한 protocol

장 버전이다. NS-2는 NS-1에서 확장 TCL(Tool Command Language) 대신 Otcl(Object TCL)을 사용하여 새로운 구조를 갖도록 개선한 것이다. NS-2는 일반적인 네트워크 시뮬레이터로 그림 1과 같이 TCP, 라우팅, Real Time Protocol 등 다양한 유/무선 인터넷 프로토콜에 대한 시뮬레이션을 수행하기에 적절한 환경을 제공하며, 사용에 대해 licence에 대한 제약을 받지 않기 때문에 학교를 비롯한 많은 연구실에서 많이 사용되고 있다.

앞서 언급한 바와 같이 센서 네트워크는 일반적인 네트워크와 그 구조를 다르게 하고 있기 때문에 일반적인 NS-2에 대한 시뮬레이션 결과를 센서 네트워크의 시뮬레이션 결과로 반영하는 것은 적당하지 않다. 특히 Physical layer 규격을 IEEE 802.15.4 LR-PAN (Low Rate Personal Areal Network) 수준이 아닌 IEEE 802.11 수준을 사용할 경우 패킷 성공률, throughput, 간섭 수준 등이 차이를 보이기 때문에 적당하지 않다. 따라서 최근에는 이러한 요구사항에 맞추어 센서 네트워크 및 MANET(Mobile Ad Hoc Network)을 위한 NS-2 확장 플랫폼[2]이 개발되고 있다. 이러한 확장 플랫폼은 센서 네트워크 시뮬레이션이 용이하도록 센서들에 대한 메시지 발생 및 컨트롤, 모바일 타겟 및 모바일 노드에 대한 시뮬레이션, 그리고 주기적 데이터 생성과 특정 지역에서 발생하는 다량의 데이터 생성에 대해 대처 가능한 기법이 포함되어 있다. Naval Research Lab에서는 센서네트워크 기능을 지원하기 위해 NRL's Sensor Network Extension[7]을 개발하였다. NS-2가 설치된 환경에 NRL's Sensor Network Extension을 추가로 설치하여 사용 할 수 있도록 개발되었다.

NS-2는 Object-oriented 기법(C++, Otcl)을 사용하기 때문에 모듈 기반의 프로그래밍이 가능하며 사용한 모듈 및 코드의 재사용을 쉽게 한다. 또한 control(Otcl)과 data(C++)에 대해 구분하여 사용함으로

기반으로 UC Berkeley에서는 REAL이라는 네트워크 시뮬레이터를 개발하였고 향후 LBNL(Lawrence Berkeley National Laboratory)에서 REAL을 기반으로 만든 네트워크 시뮬레이터가 NS-1이다.

써 사용자에게 편의성을 제공한다. 하지만, NS-2는 packet level에서의 시뮬레이션을 수행하기 때문에 센서 네트워크의 저전력 통신 기법 중 하나인 B-MAC이나 X-MAC과 같은 bit level의 시뮬레이션을 수행하는데 한계가 있다. 그리고, 다른 네트워크 시뮬레이터와 비교해 상대적으로 배우기가 어려우며, 대규모 센서네트워크에서의 구동에 수행 시간이 오래 걸리는 단점이 있다.

2.2 OPNET

대표적인 네트워크 시뮬레이터의 하나인 OPNET은 1984년 MIT의 LIDS(Laboratory for Information and Decision Systems)에서 미 국방성 연구 프로젝트를 수행하면서 만들어진 것으로 그 근본을 통신망에 두고 있다. 기본적인 네트워크 모델링, 시뮬레이션, packet level 시뮬레이션을 수행하며, 사양 및 특정 응용에 따른 다양한 확장 버전을 갖고 있다. OPNET은 finite state machine과 child process의 개념을 사용하는 state transition diagram editor를 제공하여 모델 자체 구조에 대한 이해를 쉽게 할 수 있도록 되어 있고, debug 시에도 보다 수월한 작업이 가능하다. 실제 많은 기업에서도 사용에 대한 편의성 및 state의 재사용성이 뛰어나기 때문에 네트워크 시뮬레이터로 OPNET을 많이 사용하고 있다.

OPNET의 경우 현재 대규모의 유/무선 네트워크 시뮬레이션에 사용되고 있기 때문에 아직 센서 네트워크를 위한 공식 라이브러리는 출시되고 있지 않는 상황이다. 그리고 OPNET은 정식 licence에 대한 비용이 상대적으로 비싼 편이며, 사용 및 교육에 대한 이해가 쉬운 대신에 다량의 노드가 존재할 경우 시뮬레이션 속도가 상대적으로 느리다는 단점이 있다.

정식 센서 네트워크 release는 없지만 OPNET을 센서 네트워크 시뮬레이션에 응용하기 위한 연구는 현재 진행 중에 있다. 그 대표적인 것들 중 하나가 TinyOS 와의 호환을 위해 개발된[4]이다. OPNET의 경우 state transition을 제공하기 때문에 TOSSIM과 달리 시나리

오 설정 및 통계자료 관리가 용이하다. 이러한 장점을 TinyOS 시뮬레이션에 적용한 것이다.

그림 2는 일반적인 OPNET model을 NesC compile 된 TinyOS 코드와 함께 compile하여 OPNET에서 TinyOS로 구성된 프로그램에 대한 네트워크 시뮬레이션을 구현한 예이다. 그림 3은 실제 TinyOS의 샘플응용인 CntToLedsAndRfm 프로그램을 구성한 state diagram이다.

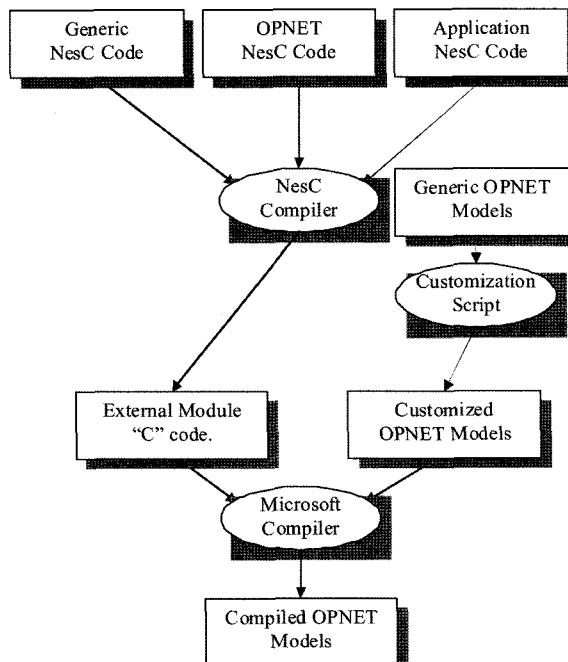


그림 2 OPNET TinyOS 시뮬레이터 구조[4]

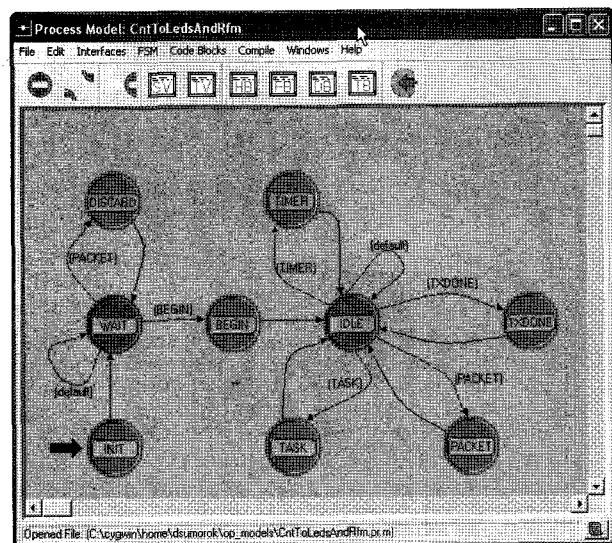


그림 3 CntToLedsAndRfm 응용의 State diagram

이러한 작업을 위해서는 TinyOS의 radio packet 및 radio 채널에 대한 정보를 모두 OPNET에 제공해야 한다. 이를 위해 [4]에서는 timer, radio, LED, TinyOS components와 radio/serial link에 대한 부분을 구현하였다. 이와 같은 연구 결과는 OPNET을 이용한 센서 네트워크 개발에 큰 도움을 줄 수 있을 것으로 기대된다. 하지만 [4]의 시뮬레이터는 개발에서 언급된 일부 응용에서만 동작 가능하며 전체적인 TinyOS 기능을 만족하기 위해서는 아직 그 개발 과정이 많이 남아있다. 그리고 추가적인 MAC, Routing과 같은 protocol 개발 시, 호환 및 안정성 문제가 남아있다.

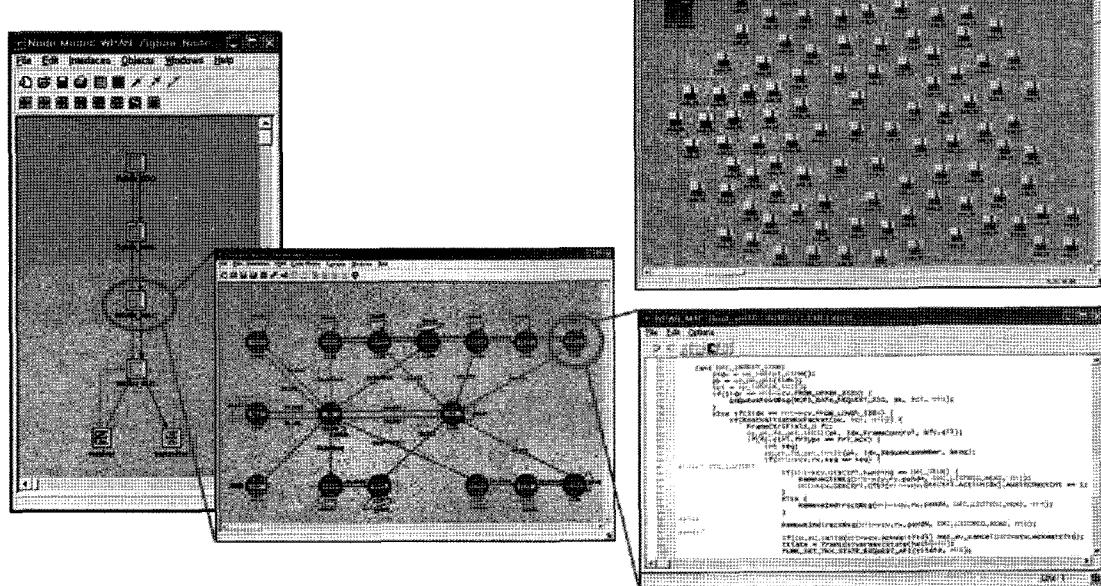


그림 4 OPNET에서 구현된 ZigBee 스택

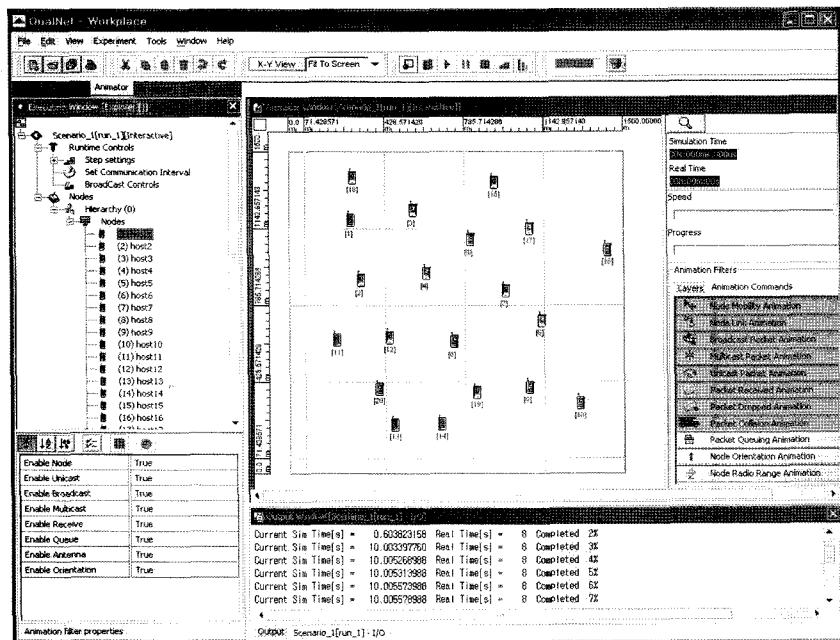


그림 5 QualNet 시뮬레이터

이 외에도 한국전자통신연구원 RFID/USN연구그룹에서는 OPNET 환경에서 IEEE 802.15.4 표준을 만족하는 physical 및 MAC layer, 그리고 Zigbee의 network layer를 library 형태로 구현하였다. 센서 네트워크의 표준이 이루어지지 않은 상황에서 가장 센서 네트워크의 응용 수준과 가까운 ZigBee의 통신 규격을 OPNET에서 지원함으로써 센서 네트워크 시뮬레이터로 활용할 수 있는 환경을 구축하였다.

2.3 QualNet

QualNet은 미국 SNT(Scalable Network Technologies)사에서 개발된 네트워크 시뮬레이터이다. QualNet 역시 NS-2, OPNET과 같이 기존의 유/무선 통신망의 네트워크 시뮬레이터로 활용되던 것으로 최근 센서 네트워크 지원을 위한 library(가칭 SenQ 또는 sQualNet)를 UCLA와 공동으로 개발 중에 있다.

SenQ의 경우 앞서 언급한 OPNET과 마찬가지로 QualNet에서 센서 네트워크 시뮬레이션을 수행하는데 가장 가까운 표준의 하나인 ZigBee에 대한 통신 프로토콜을 지원하며, 이 외에 기타 센서 네트워크에 특화된 센서 정보, 센서 데이터 포맷, 그리고 기타 특화된 통신 프로토콜을 제공하기 위한 library이다. 그림 6과 같이 SenQ는 일반 센서 노드가 가지는 radio, CPU, ADC(with Sensor), 배터리에 대한 모델 데이터를 갖고 있으며 이를 시뮬레이션에 반영하도록 구성되어 있다. 현재 SenQ는 UCLA와의 결과물로 존재하며 정식 release는 되어 있지 않은 상태이다.

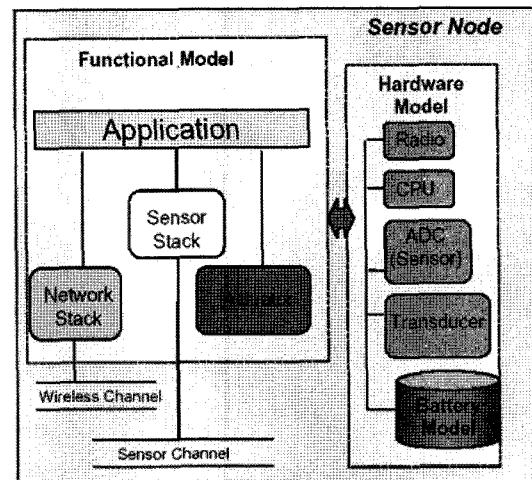


그림 6 SenQ 구조도

OPNET의 state 기반에서 동작하는 것과는 달리 QualNet은 NS-2와 유사하게 C function으로 구성된다. 따라서 코드에 대한 이해나 초기 배우는 과정에서 OPNET보다 어려움이 있는 대신 대규모 네트워크 시뮬레이션에서 OPNET보다 빠르게 동작한다. 차후 SenQ의 정식 release가 이루어진다면 향후 센서 네트워크 시뮬레이션에 보다 적극적으로 반영할 수 있을 것으로 예상된다. QualNet 역시 상용 네트워크 시뮬레이터이기 때문에 OPNET과 유사하게 licence 금액이 필요하다.

2.4 TOSSIM

TOSSIM은 앞서 언급한 NS-2, OPNET, QualNet과

는 달리 센서 네트워크에 특화된, 보다 정확히 말해 센서 네트워크 OS인 TinyOS에 특화된 시뮬레이터다. TinyOS는 UC Berkley의 SmartDust 프로젝트의 일환으로 개발된 센서 네트워크 운영체제로 nesC라는 programming language를 활용하는 component 기반의 운영체제이다. TinyOS는 센서 네트워크 개발에 있어 가장 활발한 움직임을 보이는 단체로 현재 2.0.2 version 을 release 했다. SmartDust 프로젝트의 일환으로 구현된 mica2, micaz, telosa, telosb 등 crossbow 사 및 기타 대부분의 open 플랫폼을 지원하기 때문에 국제적으로 가장 널리 이용되고 있다. TOSSIM은 이와 같은 TinyOS 정식 release²⁾의 모든 기능을 동일하게 공유한다. 그림 7은 TOSSIM의 간단한 구조를 나타낸다.

TinyOS가 component 기반으로 구성되어 있기 때문에 TOSSIM 역시 시뮬레이션에서 component에 대해 동일하게 구성한다. 그림 7과 같이 H/W 인터페이스인 clock, radio, ADC와 같은 부분에 TOSSIM이 동작 하며 radio model 및 생성 노드 간 통신, 메시지 충돌, 노드 관리 등을 구성한다. TOSSIM을 사용하기 위해서는 TinyOS환경에서 make [platform] 형태로 compile 하는 것과 유사하게 TinyOS-1.x version에서는 platform에 [pc], TinyOS-2.x version에서는 platform에 [micaz sim]을 입력하면 된다.³⁾ 이와 같이 execute 파

일을 구성한 후, 노드의 수를 정하여 simulation을 수행할 수 있다. TOSSIM은 break, function call, DBG를 이용한 debug를 수행할 수 있으며, java 기반의 TinyViz 를 이용한 User Interface를 지원한다. 그림 8은 TinyViz 화면이다.

TinyViz는 ADC reading, break point, debug message, sent message, radio model, radio link, neighbor map 등 다양한 plug-in을 지원한다. 하지만 TinyViz를 활용할 경우 기존 text 기반의 TOSSIM 보다 대규모 네트워크 동작 측면에서 시뮬레이션 속도 저하가 발생할 수 있다.

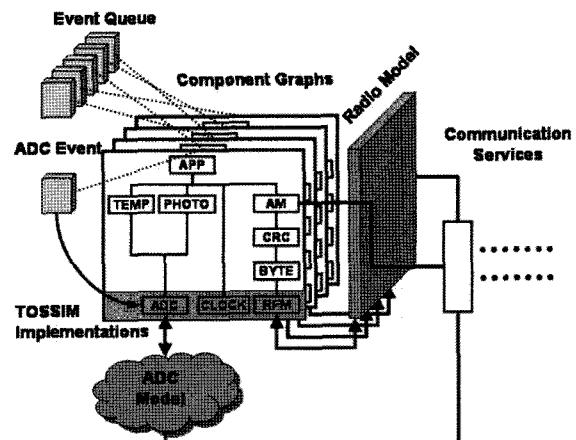


그림 7 TOSSIM 구조도

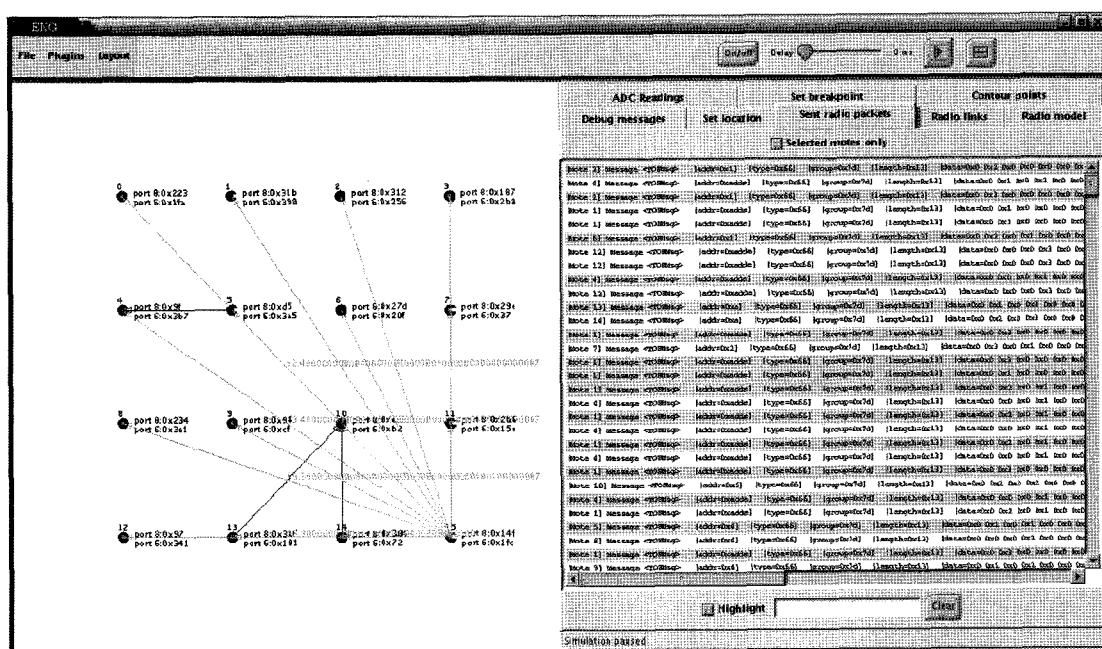


그림 8 TinyViz와 연결된 TOSSIM

2) TinyOS의 경우 자신만의 프로젝트를 위해 수정 및 보완된 프로그램에 대해 CVS의 source forge를 이용해 프로그램을 공개하고 있으며 이와 같은 프로그램은 contrib/ 폴더에 넣어둠으로써 특정 플랫폼의 프로그램을 open source 형태로 공개하고 있다.

3) TinyOS-1.x와는 달리 TinyOS-2.x에서는 micaz 플랫폼에 최적화 하였다. 이는 micaz의 radio인 CC2420의 radio 특성에 대한 구체적인 반영이 이루어졌기 때문이다. 현재 TinyOS-2.x에서는 따라서 TOSSIM에서 micaz만을 지원하고 있다.

기본적으로 TOSSIM은 TinyOS에 특화된 센서 네트워크 시뮬레이터로 TinyOS에서 개발중인 프로그램에 대한 source level debug 혹은 packet 기반의 시뮬레이터로 활용하기에 적당하다. 하지만, TOSSIM은 원천적으로 TinyOS에서만 활용 가능한 한계가 있다.

2.5 기타 센서 네트워크 시뮬레이터

앞서 언급한 시뮬레이터 외에도 각 프로젝트의 일환으로 개발된 많은 시뮬레이터, 예를 들면 SensorSim, EmStar 등이 존재한다. 하지만 대부분의 시뮬레이터들은 프로젝트에 특화된 경우가 많으며(ex. EmStar의 경우 linux 기반에서 동작하며, linux 구동이 가능한 노드(ex. iPAQ)에 대해서만 시뮬레이션을 할 수 있다.) 전체적인 시나리오를 구성하고, 범용적인 센서 네트워크 시뮬레이션을 수행하기에는 아직 무리가 있다.

3. 센서 네트워크 테스트베드

네트워크 시뮬레이터는 실 환경과 같은 전파환경이나 센서들의 센싱 특성을 정확히 모델링하기가 어려워 시뮬레이션 결과를 통하여 실제 환경을 판단하기에는 한계가 있다. 이러한 네트워크 시뮬레이터들의 문제점들을 보완하기 위하여 실제 센서노드들로 구성된 테스트베드를 개발 및 검증을 위하여 널리 사용한다.

이 장에서는 오하이오 대학의 Kansei, Harvard 대학의 MoteLab 그리고 UC 캠퍼스(UCLA, UCSB, UCR, UCD, UCSD)들과 USC 및 멜라웨어 대학으로 구성된 여러 대학이 연합으로 구축한 WHYNET들에 대해 소개하고 각각의 특징들에 대해 설명을 한다.

3.1 Kansei 테스트베드

오하이오 주립대에 있는 Kansei 테스트베드(그림 9)는 대규모 무선 네트워크 환경에서 센서 응용에 대



그림 9 Kansei Testbed에 설치된 센서 노드

한 연구를 용이하게 하기위해 설계되었다. Kansei 테스트베드는 다중 통신 네트워크, 다중 센서, 정적인 센서노드 이동 센서 노드를 가진 이기종 하드웨어로 구성되며 외부사용자들에게 웹서비스를 제공하여, 실행하는 어플리케이션에 대한 시각화, 디버깅, 모니터링 등이 가능하다. 실시간 데이터와 이벤트 발생이 가능하며 복합적이고 다계층적인 실험을 지원한다.

Kansei 테스트베드 하드웨어 구성은 Stationary array, Portable array, Mobile array등 3개의 array로 구성되어 있다.

- Stationary array : 그림 10과 같이 DARPA의 Extreme Scale(ExScal) 프로젝트에 사용한 Extreme Scale Mote (XSM)와 Stargate 노드등 총 210개, 3피트 간격으로 15×14 의 장방형 그리드로 구성되어 있다. XSM은 7.3 MHz 8-bit CPU, 128-Kbyte instruction memory, 4 Kbyte의 RAM을 가지고, 주파수는 433MHz를 사용하며 38.4 Kbps의 대역폭을 가진다. 센서로는 조도, Passive Infra Red(PIR), 자기, 음향 센서를 가지고 있으며 TinyOS 기반으로 동작한다. Stargate는 리눅스 기반에 Intel 400MHz PXA255 CPU를 가지고 동작한다. RS-232 시리얼, Ethernet, 그리고 USB 인터페이스를 가지고 802.11b 무선 네트워크 인터페이스 카드를 통해 WiFi 통신을 한다.

- Portable array : Stationary는 하나 또는 여러 개의 portable array와 연결될 수 있다. Kansei 테스트베드는 50개의 Trio mote로 구성된 portable array를 포함하고 있다. UC Berkeley에서 디자인한 Trio mote는 XSM sensor board(음향, PIR, 자기, 온도 센서 포함)와 TMote Sky 센서노드로 구성되며 태양전지로 충전하는 시스템이다.

- Mobile array : 그림 11과 같이 robotic mobile 노드로 구성된다. stationary array 위에 플라스틱 판넬을 설치하고 이 위에서 이동하며 동작한다. Mobile array

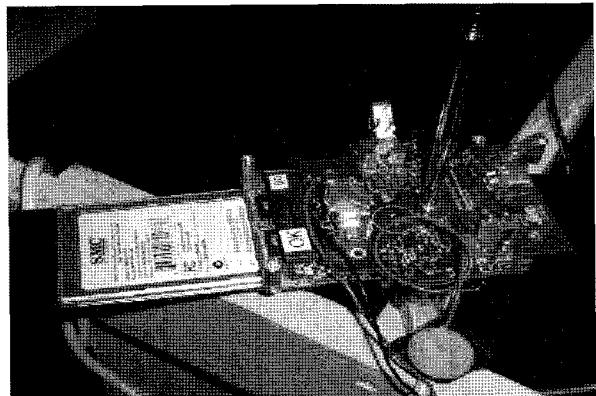


그림 10 Extreme Scale Motes(XSMs)와 Stargate

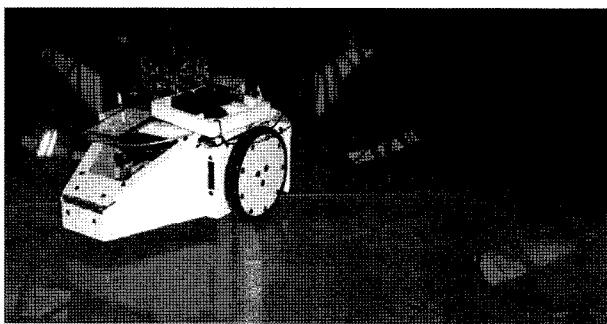


그림 11 Mobile Array

는 stationary array의 조도센서가 반응하도록 라이트 소스를 가진다. Mobile array는 stargate, XSM을 이 탑재되어 있다.

3.2 moteLab : A Wireless Sensor Network Testbed

MoteLab은 하버드 대학에 Maxwell Dworkin Laboratory, 전기 공학, 그리고 컴퓨터 과학 빌딩에서 활용된 실험적 무선 센서망(그림 12)이다. MoteLab은 웹 기반 인터페이스를 통하여 외부 사용자들에게 센서망 응용의 개발과 시험을 위한 테스트베드를 제공한다. 등록된 사용자는 실험하고자하는 프로그램을 실행 가능하도록 업로드 시키고, 프로그램을 수행하고자 하는 센서노드와 연계시킨다. 또한 사용자는 MoteLab에서 수행할 작업을 예약할 수 있다. 작업 동안 모든 메시지 및 데이터들은 사용자에게 할당 된 데이터베이스로 기록되고, 추후 시각화를 위해 이용 된다. 단순한 시각화 기능은 웹 인터페이스를 통해 제공된다. MoteLab은 센서망 프로그래밍, 통신 프로토콜 개발, 시스템 설계 및 응용에 대한 연구를 용이하게 도와 준다.

MoteLab의 구성은 그림 5와 같으며 다수의 소프트웨어 컴포넌트로 구성되어 있다. 주요 구성 컴포넌트 및 기능은 다음과 같다

- MySQL Database Backend : 실험 동안 수집된 데이터, 웹 콘텐츠를 생성하기 위해 사용된 정보와 시험장 작동 상태 등을 저장한다.
- Web Interface : 업무생성, 스케줄링과 데이터 수집을 위한 사용자 인터페이스를 제공하고 특정 테스트베드를 제어하는 기능등을 수행한다
- DBLogger : 작업에 의해 만들어진 데이터를 수집하고 분석하기 위한 자바 데이터 로거
- Job Daemon : 작업을 설정하고 분석하기 위한 perl 스크립트

처음 구성된 테스트베드는 26개의 Crossbow사 Mica2 센서노드로 구성되어 있고 각 노드들은 이더넷 인터페이스 보드인 MIB-600으로 연결되어 있으며 이 보드를 통해 데이터 로깅과 재 프로그래밍을 한다. Mica2 센서노드는 7.3MHz에 동작하는 ATmega128L 프로세서, 128KB 코드 메모리, 4KB 데이터 메모리와 433 MHz에서 약 34kbps 데이터율을 가지는 Chipcon사의 CC1000 라디오로 구성되어 있다. 최근에는 Moteiv사의 TMote Sky 센서노드도 추가 되었다. 이 센서노드는 8MHz에서 동작하는 MSP430 프로세서로 10KB RAM, 1Mbit 플래시 메모리 그리고 실내에서 최대 100m 전송이 가능한 2.4GHz CC2420 RF 칩으로 구성되어 있다. 또 다른 센서노드로는 CC2420 RF 칩을 사용하는 Crossbow사 MicaZ가 있다.

3.3 WHYNET

WHYNET은 차세대 무선 통신 기술과 응용 개발을

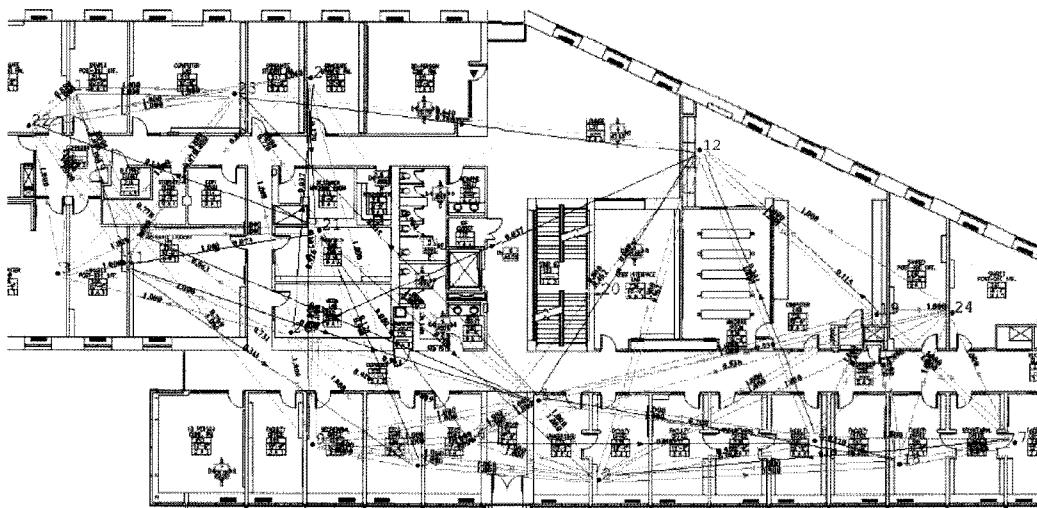


그림 12 MoteLab에서 배치된 센서노드 및 링크 연결 관계

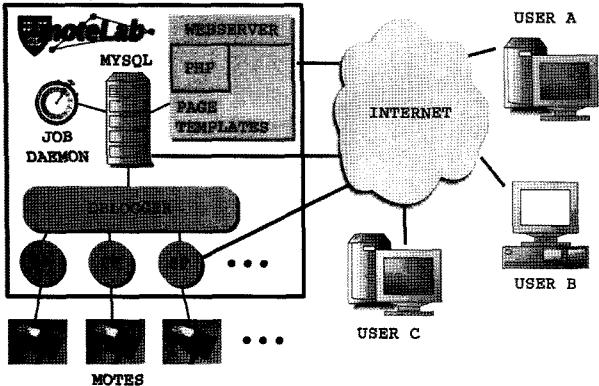


그림 13 MoteLab 구성도

위한 하이브리드 무선 테스트베드 환경을 제공한다. WHYNET은 다섯 UC 캠퍼스(UCLA, UCSB, UCR, UCD, UCSD)와 USC, 멜라웨어 대학으로 구성된 여러 대학의 협력적 노력에 의해 진행된 프로젝트이다. WHYNET 시험망 인프라의 요소는 그림 14와 같으며 구성하는 주요 요소들은 다음과 같다.

- TWINE: 다음 두가지 특성을 가진 이기종 에뮬레이션 프레임워크 : (i) 확장성과 리어리즘을 제공하기 위해 에뮬레이션, 시뮬레이션 및 물리적 테스트베드를 이음매 없게 연결 (ii) 실시간으로 높은 신뢰도를 가지는 라디오와 채널 에뮬레이션
- 802.11 기반의 네트워크(wireless LANs, mesh networks, MANETs, VANETs), 센서 네트워크, CDMA2000 셀룰러 시스템, SDR, MIMO 플랫폼, UWB 디바이스
- 상세하고 실시간 신호 수준의 채널 에뮬레이션 성능을 제공하는 하드웨어 기반의 채널 에뮬레이터

WHYNET 시험망 컴포넌트들은 이종의 대규모 무선망 시나리오에 있어서 어플리케이션-레벨 성능에 크로스 레이어 상호작용의 영향을 포함하여 다양한 평가 연구에 사용되고 있다.

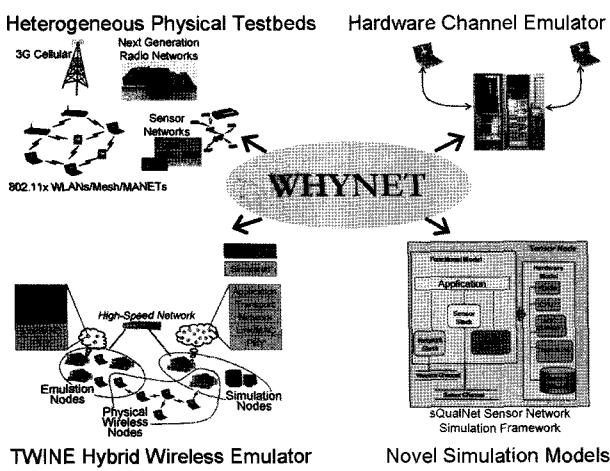


그림 14 WHYNET 시험망 인프라 요소

4. 결론

본 논문에서는 센서 노드들을 개발하고 개발된 센서노드들의 H/W 및 S/W를 검증하기 위한 네트워크 시뮬레이터 및 테스트베드에 대해서 소개를 하고 장단점 및 특징들을 소개하였다. NS-2, OPNET, QualNet, TOSSIM과 같은 네트워크 시뮬레이터들은 센서노드를 구성하는 각 Layer들의 H/W 및 S/W를 모델링하여 시뮬레이션을 수행한다. 네트워크 시뮬레이터들은 동일한 조건을 계속 시험하여 에러가 발생되는 부분에 대한 debugging이 용이하고 컴퓨터의 성능에 따라 센서노드의 수를 확장 할 수 있으며 또한 시뮬레이션 수행시 노드 배치등을 쉽게 할 수 있는 특징들을 가지고 있다. 그러나 네트워크 시뮬레이션에서는 실환경과 같은 전파환경이나 센서들의 센싱 특성을 정확히 모델링하기가 어려워 실 환경에서 작동시 발생될 수 있는 모든 요소들을 시뮬레이션 통하여 결정하기에는 한계가 있다. 이러한 네트워크 시뮬레이터들의 문제점을 보완하기 위하여 실 센서노드들로 구성된 테스트베드를 이용하여 개발 및 검증을 실시한다. 테스트베드는 실 환경을 반영 할 수 있지만 센서노드 개수를 크게 확장하기에는 한계가 있고 발생된 오류에 대해 디버깅하기가 어렵다는 단점을 가지고 있으므로 센서노드 H/W 및 탑재되는 S/W를 신뢰성이 보장되게 개발하기 위해서는 네트워크 시뮬레이터와 테스트베드를 서로 보완하여 사용하여야 한다.

참고문헌

- [1] NS-2 Official Site, <http://www.isi.edu/nsnam/ns/>
- [2] Ian Downard., “Simulating Sensor Networks in NS-2,”
- [3] OPNET homepage, <http://www.opnet.com/>
- [4] Daniel Sumorok, David Starobinski, Ari Trachtenberg, “Simulation of TinyOS Wireless Sensor Networks Using OPNET”
- [5] QualNet homepage, <http://www.scalable-networks.com/>
- [6] TOSSIM official site, <http://www.cs.berkeley.edu/~pal/research/tossim.htm>
- [7] NRL's Sensor Network Extension to NS-2 official site, <http://cs.itd.nrl.navy.mil/work/sensorsim/index.php>
- [8] Maneesh Varshney, Defeng Xu, Mani Srivastava, and Rajive Bagrodia, “SenQ: A Scalable Simulation and Emulation Environment for Sensor Networks”, ACM IPSN 2007, pp. 196–205, 2007.
- [9] Ertin, E., Arora, A, Ramnath, R., Nesterenko, M., Naik, V., Bapat, S., Kulathumani, V., Sridharan,

- M., Hongwei, Z., Hui, C., "Kansei: A testbed for sensing at scale", IPSN, pp.399–406, 2006
- [10] <http://ceti.cse.ohio-state.edu/kansei/>
- [11] <http://motelab.eecs.harvard.edu/>
- [12] Werner-Allen, G., Swieskowski, P., Welsh, M., "Mote-Lab: a wireless sensor network testbed", IPSN, pp.483–488, 2005
- [13] <http://chenyen.cs.ucla.edu/projects/whynet/>



박상준

1990 경북대학교 전자공학과 졸업(공학석사)
1990~2001 국방과학연구소(선임연구원)
2006 노스캐롤라이나 주립대 졸업(공학 박사)
2006~현재 한국전자통신연구원 팀장
관심분야: 무선센서네트워크, 데이터융합
E-mail : sangjoon@etri.re.kr



문영백

1996 경희대학교 전자공학과(공학사)
2001 경희대학교 전자공학 전공(공학석사)
2001~현재 한국전자통신연구원 선임연구원
관심분야: 센서네트워크 시스템, 테스트베드
E-mail : moonyb@etri.re.kr



박종준

2004 포항공과대학교 전자전기공학과(공학사)
2006 포항공과대학교 전자전기공학 전공(공학석사)
2006~현재 한국전자통신연구원 연구원
관심분야: 센서 네트워크 시스템, 위치인식
E-mail : juny@etri.re.kr