17-3-R05

센서 네트워크 어플리케이션 개발을 위한 NesC (NesC Programming Language for Sensor Network Application Development)

이 민 구, 강 정 훈, 유 준 재
전자부품연구원 시스템연구본부
{emingoo, budge, yoojj}@keti.re.kr

요 약

최근 유비쿼터스(Ubiquitous)라는 단어가 다양한 과학 기술 분야에서 많이 사용되어지고 있다. 그리고 조만간 도래하게 될 유비쿼터스 컴퓨팅 & 네트워킹 시대의 핵심 기술로 떠오르고 있는 센서 네트워크 분야에서 TinyOS가 현재 적합한 OS로 촉망 받고 있으며, 전 세계 100여개 이상의 센서 네트워크 연구 그룹들에 의해서 채택되어 이용되고 있다. 이러한 TinyOS에서 구현하고자 하는 개념들을 지원하기 위해 고안되어진 NesC 프로그래밍 언어의 전반적인 특성에 대한 소개를 본 논문을 통해서 제공하고자한다. 본 논문은 이를 위해 Mote, TinyOS, NesC에 대한 각각의 기본적인 개념들에 대한 접근을 하였고, 예제로서 사람의 위치를 Tracking 할 수 있는 어플리케이션을 NesC를 이용해 직접 구현하였으며, Mote의 정상적인 추적이 가능함을 실험적으로 살펴보았다. 또한 앞으로 더욱 규모가 큰 시스템들에 적용하여 적합하도록 만들기 위해 NesC가 나아가야 할 발전 방향에 대해서 제시하였다.

1. 서론

NesC 라는 프로그래밍 언어는 [nes-si:]로 발음 되어지며, TinyOS의 구조적인 개념들과 실행 모델들을 구현하기 위해 사용되어 졌다. 이는 어플리케이션 개발자들에게 센서 네트워크와 같은 임베디드 네트워크 시스템들의 구현을 지원하기 위한 새로운 프로그래밍 영역을 제공해 주고자 개발 되어진 프로그래밍 언어이다. 이러한 센서 네트워크는 초소형, 초저전력의 특징을 갖는 Mote들로 구성이 되어지며, 이들 Mote 각각은 동시에 동작이 되어야하고, 작은 메모리와 적은 파워 소비라는 제한된 한계를 가지고 동작해야한다.

위에서 언급되어진 TinyOS는 Mote와 같은 작은 메모리들, 작은 외형 사이즈, 제한적인 파워 소비 등의 한정적인 자원을 갖고서 운용되는 센서 노드들을 위해 디자인 되어진 이벤트 기반의 운영 체제를 의미한다.

NesC와 TinyOS는 현재 전 세계의 100여개 이상의 네트워크 연구 그룹들에 의해서 채택되어져 활발히 사용되어지고 있다. 이와 같이 NesC 프로그래밍 언어가 센서 네트워크와 같은 여러 중요한 센서 어플리케이션 분야에 널리 사용되어 질 수 있는 이유는, 새로운 분야인 임베디드 네트워크 시스템들에 의해서 요구되어지는 복잡성과 동시성을 충분히 지원해 줄 수 있는 프로그래밍 언어이기 때문이다.

본 논문에서는 이상에서 언급되어진 Mote, TinyOS, NesC등에 대한 이해를 돕기 위해 각

각의 구체적인 설명을 제공하고, 실제로 NesC 프로그래밍 언어로 구현되어진 예제 어플리케이션에 대해 살펴봄으로 국·내외적으로 초미의 관심사로 떠오르고 있는 유비쿼터스 컴퓨팅 & 네트워킹의 핵심 기술인 센서 네트워크에 대해서 좀 더 폭 넓은 이해를 돕고자 한다. 추가로 마지막 장에서는 결론 및 향후 연구 방향에 대해 살펴봄으로 본 논문의 끝을 맺고자 한다.

2. Mote

우리가 살펴보고자 하는 센서 네트워크들은 수많은 Mote들로 구성된다. 이와 같은 센서 네트워크를 최초로 이용하기 시작한 분야는 지진과 같은 초자연적인 위협이 도사리고 있어 인간이 직접 접근하여 연구 활동을 하기에는 위험과 한계가 있는 곳에 Mote들을 배치해 환경 관련 정보 데이터를 계속해서 수집하고, 제공하도록 하는 것이었다. 이와 같은 기능을 수행하기 위해서 Mote들은 다음의 세 가지 특성을 가져야만했다.

① 센서를 통한 환경 정보 습득
② 무선 네트워크를 통한 통신
③ 초저전력의 긴 운용 시간

네트워킹 기술과 집적 기술에서의 발전은 소형의 Mote들이 센서, 무선통신을 통해 그들이 놓여있는 환경과의 상호작용이 가능하도록 하였다. 가격적인 측면에서도 Mote 한 개가 10센트 미만이 되어, 수 만개의 Motes 들을 갖는 네트워킹의 구성이 가능할 것이다. 또한, 전력 소비의 관점에서 살펴보면, Mote 들은 낮은 대역폭을 이용하여 1년 정도를 지속할 수 있으며, 소진된 배터리는 대기의 힘(대기의 열)에 의해서 충전이 가능하도록 고안 및 발전될 것이다.

Mote들은 위에서 언급하였듯이 활용 할 수 있는 자원들이 매우 제한적이기는 한계를 가지고 있지만, 복잡한 분산 알고리즘 등에서 효율적으로 동작해야만 하는 요구사항을 동시에 가지고 있다. 이와 같이 모순된 요구사항들을 만족하기 위해서 기존의 OS와 프로그래밍 모델들을 센서 네트워크 분야에 적용하였으나 결과적으로 부적합함을 알게 되었다. 따라서 이와 같은 제한 조건에서도 여러 복잡한 분산 알고리즘들을 처리하는데 효율적인 OS와 프로그래밍 언어를 위해 개발한 결과물이 TinyOS와 NesC이다.

3. TinyOS

TinyOS는 센서 네트워크와 같은 임베디드 네트워크 시스템들을 위해 특별히 고안 되어진 OS이며, 이는 이벤트 기반의 어플리케이션, 소형의 코어 OS(400 바이트 정도의 코드), 작은 데이터 메모리를 갖는 초소형 용량의 OS를 만들기 위해 고안되어졌다.

이와 같은 TinyOS는 다음의 세 가지 특성을 갖는다.

① 컴포넌트 기반의 구조
② 테스크, 이벤트 기반의 동시성
③ 구분된 동작

위의 특성들에 대해 각각 살펴보면, 먼저 TinyOS는 재사용이 가능한 시스템 컴포넌트들을 기반으로 하여 구조가 이루어져 있다. 즉, 어플리케이션들이 구현에 필요한 각각의 컴포넌트들을 Wiring Specification을 이용하여 연결하여 구성한다. 이와 같이 컴포넌트 기반으로 이루어진 구조에서는, 다른 OS 서비스들로 구분되어진 컴포넌트들을 다른 어플리케이션에서 사용하지 않아도 되는 장점을 제공한다.

TinyOS의 동시성을 확보하기 위해 사용되는 두 가지 것들이 테스크(Task)와 이벤트(Event)이다. 이 두 가지 요소들의 차이는 테스크나 이벤트들에 대한 선점가능 여부이다. 즉, 테스크들은 서로를 선점하지 않는 반면, 이벤트들은 테스크들이나 이벤트들의 실행에 대한 선점이 가능하다.

구분된 동작의 특성은 테스크들의 특성을 통해 그 의미를 찾을 수 있다. 즉, 테스크들은 위에서 언급했듯이 선점하여 실행되지 않기 때문에, TinyOS의 경우에는 Blocking 동작이 없다. 따라서 긴 시간동안 잠재적으로 동작하는 모든 작용들은 구분되어져 작동을 하게 되며, 이와 같은 동작에 대한 요구와 완성은 각각 구분되어진 함수들이다. 예를 들자면, 명령어들이 동작의 실행을 요청하게 되면, 즉시 명령들은 리턴(Return)하게 되고, 완성에 대한 통보는 이벤트를 통해 신호되는 구분되어진 동작을 한다.

4. NesC

NesC는 3장에서 언급되어진 TinyOS의 특성들을 지원하기 위해 고안된 프로그래밍 언어이다. 즉, NesC 어플리케이션들은 컴포넌트와 양방향의 인터페이스들로 구성되고, 테스크와 이벤트들에 기반한 동시성 모델을 제공한다는 점이 특징이다.

NesC는 커다란 관점에서 보면, C 언어의 확장 정도로 볼 수 있다. 이처럼 C 언어에 근거를 둔 이유는, C 언어가 마이크로 컨트롤러들을 위한 효율적인 코드 생성이 가능하고, 하드웨어에 액세스하기 위해 필요한 기본적인 특성들의 지원이 가능하며, 기존의 C 코드와의 상호작용이 단순하다는 장점 때문이다. 물론 기존의 많은 프로그래머들이 C 언어에 익숙해져 있다는 점도 NesC가 C 언어에 근거를 둔 큰 이유이다.

하지만, 위에서 언급한 C 언어의 장점만으로는 NesC의 초소형, 초저전력, 컴포넌트 기반의 구조라는 특성을 만족시키는 데에는 부족함이 있다. 즉, C 언어는 안전한 코드를 만든다는 점에서는 커다란 도움이 되지 못하며, 어플리케이션들을 구조화해야한다 라는 부분에서도 만족스럽지 못하다.

결국 NesC는 컴포넌트라는 개념에 기반 하여 TinyOS의 이벤트 기반의 동시성을 지원하고, 공유된 데이터의 동시 액세스가 가능하게 되었다.

NesC에서의 특징들을 살펴보면, NesC는 컴포넌트로 구성되는데 이러한 컴포넌트는 모듈과 Configuration이라는 두 개의 타입으로 존재한다. 모듈은 어플리케이션의 코드를 제공하거나, 한 개 이상의 인터페이스들을 수행하며, Configuration은 컴포넌트가 사용하는 인터페이스들을 연결하는데 사용된다.

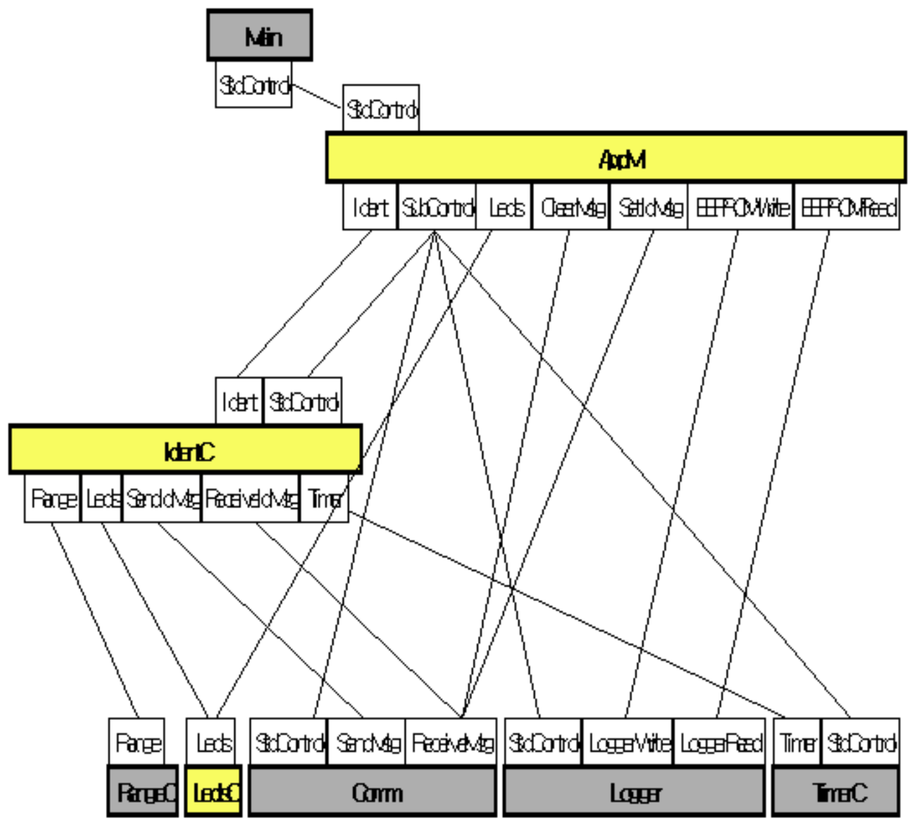
컴포넌트들은 인터페이스들을 제공하거나 사용한다. 즉, 이러한 인터페이스들이 컴포넌트에 액세스하는 유일한 통로를 제공하는 것이다. 특히, 인터페이스는 양방향이라는 특성을 갖는데, 이는 커맨드와 이벤트를 포함함으로 가능하다. 예를 들어, 제공자는 인터페이스가 커맨드를 구현하는 반면에 사용자는 이벤트를 구현하는 것이다.

NesC의 또 다른 큰 특징은 동시성이다. 이러한 동시성을 확보하기 위해서 두 가지의 방법을 사용한다. 즉, 테스크와 Atomic 구문이다. 테스크의 경우에는 동시성 위반 예상 코드 부분을 테스크로 따로 설정 한 후 Post를 이용하여 연기함으로 동시성을 보장 할 수 있고, Atomic의 경우에는 동시성 위반 예상 부분을 Atomic 구문내로 삽입하여 사용하면 NesC에서 동시성을 보장받을 수 있다.

이상에서 살펴보았듯이 NesC는 TinyOS에서 요구하는 기능인 환경반응성, 동시성, 통신 등을 포함한 기능들을 제공할 수 있으며, 전체 프로그램 최적화와 컴파일 타임 데이터레이스 검출을 수행하여 센서 네트워크의 어플리케이션 개발이 용이하고, 또한 코드 사이즈를 줄일 수 있으며, 발생 가능한 많은 버그들을 제거할 수 있다.

5. Application

5장에서는, NesC로 구현된 실제 예제 어플리케이션을 살펴봄으로 본 논문에서 설명하고자 하는 NesC에 대한 이해를 돕고자 한다.



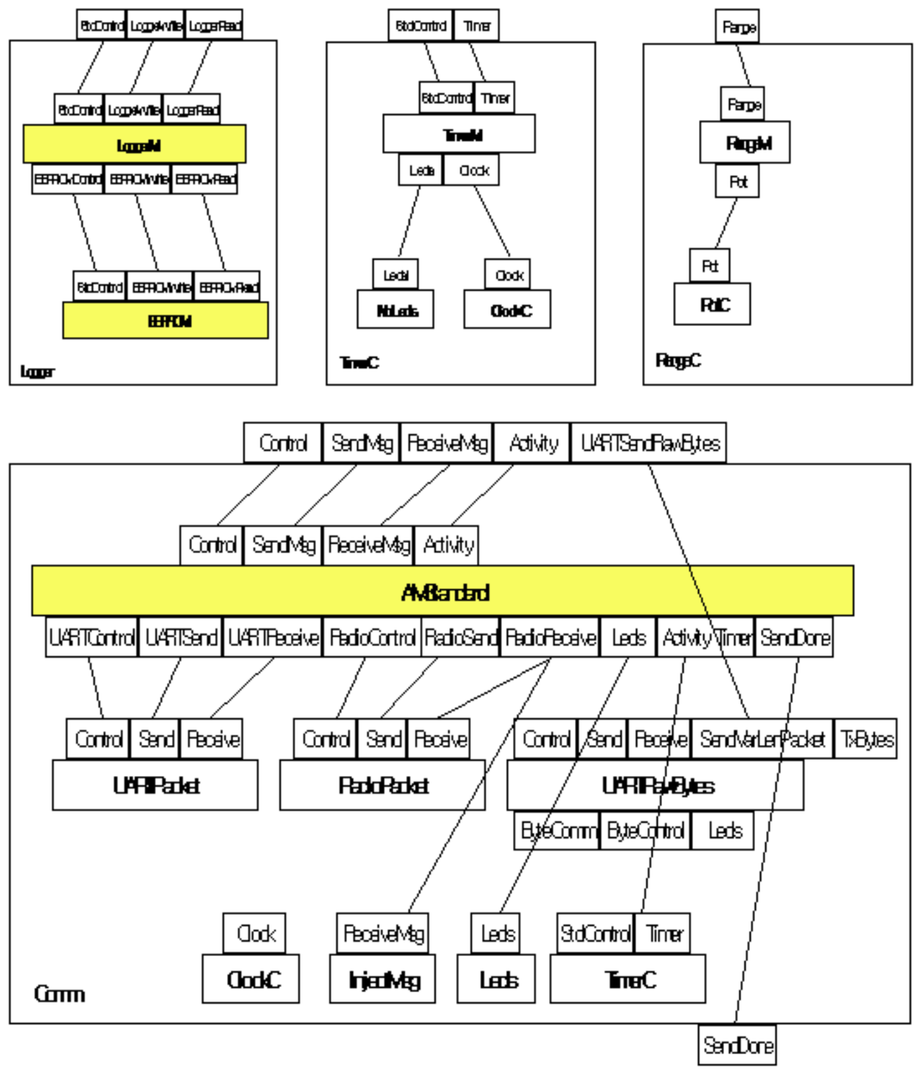
(그림 1) Ident의 최상위 Configuration

본 논문에서 다루고자 하는 어플리케이션은 Ident이다. Ident 어플리케이션은 아주 간단한 사람 추적 어플리케이션이라고 할 수 있다.

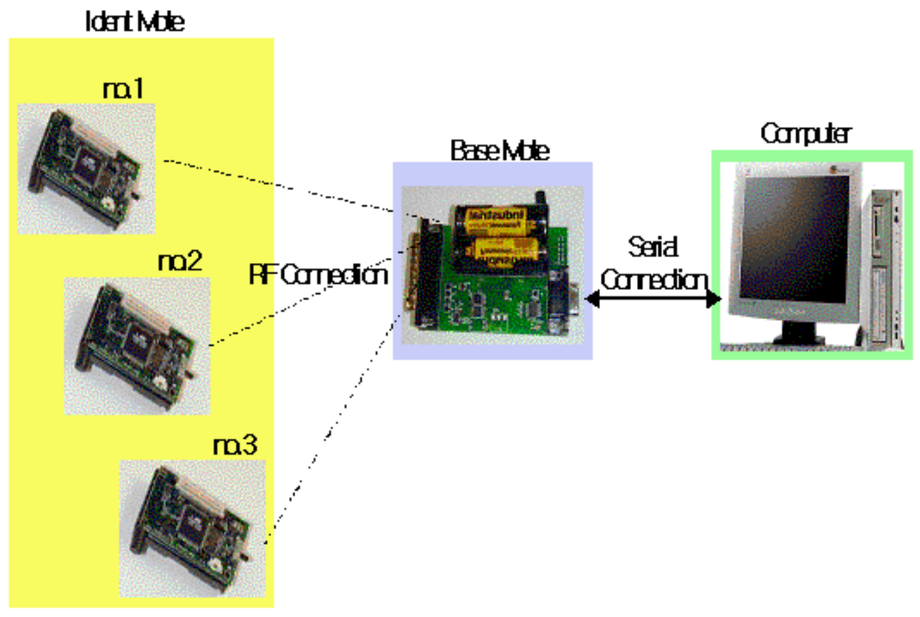
(그림 1)은 Ident 어플리케이션의 최상위 Configuration을 보여준다. (그림 1)에서의 회색으로 표시된 부분들은 컴포넌트의 구성요소 가운데 하나인 Configuration을 나타내고 있으며, 노란색으로 표시된 부분들은 컴포넌트의 구성요소 중의 다른 하나인 모듈을 나타낸다. 그리고 굵은 네모칸으로 표현된 부분은 모듈과 Configuration을 나타내고, 실선의 네모칸은 인터페이스를 표현한다.

(그림 1)의 인터페이스들 중에서 굵은 네모칸의 위에 위치하는 것들은 제공된 인터페이스를 나타내고, 아래에 위치하는 인터페이스들은 사용된 인터페이스들을 나타내고 있다. 제공된 인터페이스들은 컴포넌트가 컴포넌트의 사용자에게 제공하는 기능을 표현하기 위한 것이고, 사용된 인터페이스들은 컴포넌트가 컴포넌트의 사용자에게 작업을 수행하기위해 필요한 기능을 나타낸다.

본 논문에서는, NesC 프로그래밍 언어를 이용하여 만들어진 Ident 컴포넌트의 어플리케이션을 여러 개(실험에서는 3개를 사용)의 Mote들에 프로그래밍을 한 Ident Mote 들이 GenericBase 어플리케이션들이 프로그래밍 된 중앙 Mote와의 통신을 통하여 Ident Mote의 위치와 존재 여부를 인지하여 시리얼 통신을 통하여 컴퓨터를 통해서 확인 할 수 있음을 알아보고자 하였다.



(그림 2) 구체적인 Ident의 Configuration



(그림 3) Test 구성도

본 논문에서 실시한 실험에서는, 위의 (그림 3)에서와 같이 세 개의 Ident Mote들에게 각각의 아이디(no1, no2, no3)를 부여하면, Ident Mote의 EEPROM에 각각 부여받은 아이디(no1, no2, no3)가 저장된다. 이때 각각의 Ident Mote들이 아이디를 갖게 되면, 녹색 LED가 10초 마다 점멸을 하고, 정상적으로 아이디를 갖지 못했을 경우에 적색 LED가 점등되어진다.

이와 같은 Ident Mote들을 중앙 Mote의 일

정한 거리 내에 no1, no2, no3 각각을 하나씩 가져다 놓았을 때에 정상적으로 Ident Mote들의 개별식별이 가능하고, 3개의 Ident Mote들을 동시에 위치하였을 때에 3개 모두를 동시에 인식함을 실험적으로 확인 할 수 있었다.

Ident 어플리케이션의 구현을 통해, 우리는 본 논문에서 확인하고자 했던 NesC를 이용해 센서 네트워크의 어플리케이션을 구현하여 정상적으로 동작할 수 있음을 확인하였다.

6. 결론 및 향후 연구방향

NesC는 현재 버전 1.1이 2003년 5월에 배포되었다. 이는 임베디드 네트워크 시스템들을 프로그래밍 하는데 적합하며, TinyOS를 구현하기 위해 고안 되어졌다. 이와 같은 NesC 프로그래밍 언어는 미래 신기술의 한 부분을 담당하게 될 센서 네트워크 분야에서 더욱 그 가치가 인정될 것이다. 즉, 미래의 유비쿼터스 컴퓨팅 & 네트워킹 시대를 준비하는 새로운 개념의 프로그래밍 언어라 할 수 있다.

또한, 앞으로의 NesC는 임베디드 시스템들의 범주만으로 한정되어지지는 않을 것이다. NesC의 컴포넌트 기반의 구조, 동시성, 인터페이스의 양방향성의 특성들은 오히려 더욱 규모가 큰 시스템들에 대한 프로그래밍에 유용하고 적합하게 될 것이다.

이와 같이 NesC의 폭넓은 활용을 지원하기 위해 멀티프로세서, 블로킹, 쓰레드 등의 일반적인 개념들을 지원하도록 확장되어야 하며, 이러한 특성들은 대규모 시스템들의 컴포넌트 지향 프로그래밍을 지원하기 위해 더욱 구체적으로 적용되고, 활용되도록 연구개발 되어야 한다.

참고문헌

[1] F. Bachmann, L. Base, C. Buhrman, S. Cornella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau. Wallnau. Volume 2: *Technical Concepts of Component-Based Software Engineering, 2nd Edition*. Techinical Report CMU/SEI-2000-TR-008, Carnegie Mellon Software Engineering Institute, May 2000.

[2] M. Herlihy. A methodology for implementing highly concurrent data objects. *ACM Transactions on Programming Language and Systems,* 15(5):745-770, November 1993.

[3] P. Levis and D. Culler. Mate: a Virtual Machine for Tiny Networked Sensors. In *Proceedings of the ACM Conference on Architectual Support for Programming Languages and Operating Systems*, Oct. 2002.

[4] David Gay, Philip Levis, David Culler, Eric Brewer. *NesC 1.1 Language Reference Manual.* May 2003.

[5] J.Hill., R.Szewezyk, A.Woo, D.E.Culler, and K.S.J.Pister. System Architecture Directions for Networked Sensors. In *Architectural Support for Programming Languages and Operating Systems*, pages 93-104, 2000.

[6] B.W.Kernighan and D.M.Ritchie. *The C Programming Language, Second Edition.* Prentice Hall, 1988

[7] Edgar H.Callaway, Jr., *A Communication Protocol for Wireless Sensor Networks*, Ph.D.dissertation, Florida Atlantic University, Boca Raton, FL, August 2002.

[8] Jose A. Gutierrez et al. IEEE 802.15.4: *A Developing standard for low-power, low-cost wireless personal area networks*, ieee Network, v.15, n.5, pp.12-19, September/October 2001

이 민 구

1993년~2000년 서강대학교 전자공학과(학사), 전자공학과(석사과정)
2000년~2001년(주)한화/정보통신 네트워크 연구소 Access Network Group 연구원
2001년~현재 전자부품연구원, 시스템연구본부 디지털미디어연구센터 전임연구원
관심분야 : 데이터 방송, 센서 네트워크, 웨어러블 컴퓨팅

강 정 훈

1993년~1997년 단국대학교 전자공학과(학사)
1997년~1999년 단국대학교 전자공학과(석사)
1999년~현재 전자부품연구원 시스템연구본부 디지털미디어연구센터 전임연구원
관심분야 : 유비쿼터스 컴퓨팅, 센서 네트워크, 스케일러블 미디어 프로세싱, 서비스 디스커버리 미들웨어

유 준 재

1977년~1981년 경북대학교 전자공학과(학사)
1994년~1996년 아주대학교 컴퓨터공학과(석사)
1997년~1999년 충북대학교 정보통신학과(박사수료)
1983년~1992 고려시스템(주)
1992년~현재 전자부품연구원 시스템연구본부 유비쿼터스 컴퓨팅 센터 수석연구원
관심분야 : 유비쿼터스 네트워크 컴퓨팅, 센서 네트워크