유비쿼터스 센서 네트워크를 이용한 독거노인 지킴이 시스템 구현†

(Implementation of the Living Alone Elderly People Protection System using Ubiquitous Sensor Networks)

박 홍 진*
(Hong-Jin Park)

요 약 유비쿼터스 센서 네트워크는 초경량, 저전력 센서를 기반으로 하여 어플리케이션에 대한 수많은 연구가 진행 중에 있다. 사회가 발전하면 할수록 고령화가 되어가고 있는 요즘에 홀로 사는 노인에 대한 관심이 사회의 중요한 이슈 중 하나이다. 본 논문은 유비쿼터스 센서 네트워크를 이용하여 나 홀로 노인의 상황을 판단하여 위급시 신속한 조치를 수행할 수 있는 독거노인 지킴이 시스템을 구현한다. 센서 노드와 싱크 노드를 이용하여 독거노인의 정보를 습득하여 웹 환경에서 모니터링 함으로써 보다 통합적인 관리를 할 수 있다.

핵심주제어 : 유비쿼터스 센서 네트워크, 독거노인, 싱크 노드, 센서 노드

Abstract Numerous researches are being made on applications based on ubiquitous sensor networks and super light, low power sensors. With the development of society, the aged population is expanding and living alone elders are one of important social issues in today's society. This paper implements the living alone elderly people protection system using ubiquitous sensor networks. By collecting and monitoring information on living alone elders using sensor nodes and sink nodes in web environment, we can perform more integrated management. The implemented living alone elderly people protection system can monitor living alone elders' situation and take actions promptly in emergency.

Key Words : Ubiquitous sensor networks, Living alone elderly people, Sink node, Sensor node

1. 서 론

최근의 무선통신과 마이크로 전자공학 기술의 발전은 저 가격, 극소형의 센서들 간의 네트워크를 가능케 하고 있다. 센서 네트워크는 지능형 빌딩내의 환경 컨트롤, 생산 공정 자동제어, 창고 물류관리, 병원에서의 물품 정보 관리 및 환자상태 원격감지, 지능형 교통시스템, 텔레매틱스 등 그 응용범위가 광범위하다. 유비쿼터스 센서 네트워크(Ubiquitous Sensor Network: USN)는 유비쿼터스 컴퓨팅 구현을 위한 기반 네트워크로 초경량, 저전력의 많은 센서들로 구성된 무선 네트워크이다. 하나의 네트워크로 연결되어 있는 수많은 센서들이 필드(field)의 지리적, 환경적 변화를 감지하여 베이스스테이션으로 그 정보를 전달한 후 센서 네트워크 서버를 통해 사용자에게 전달되는 방식으로

† 이 논문은 2008년도 상지대학교 교내 연구비 지원에 의해 연구되었음.
* 상지대학교 컴퓨터정보공학부 교수

정보수집이 이루어진다.

유비쿼터스 센서 네트워크를 통해서 사물이 인간과 같은 다른 사물을 인식하고 주변 환경을 감지하게 하여, 네트워크를 통해서 언제, 어느 곳에서든 정보를 확인하고 활용할 수 있게 한다. 이런 방식은 생산, 유통, 물류 같은 경제 활동, 의료 서비스, 복지 서비스, 환경 감시 시스템 등에 적용되어 인류의 삶을 더욱 윤택하게 만들어 주는 기술이다.

유비쿼터스 센서 네트워크는 각 센서 노드들의 크기가 작기 때문에 전력과 컴퓨팅 능력 그리고 메모리에 제한이 있다. 또한 구성하는 센서의 수가 많고 필드에 센서들이 랜덤하게 배치되기 때문에 센서 간에 토폴로지를 예상하기 어려우며, 빈번한 센서 노드들의 추가와 제거에 의해 센서 네트워크의 토폴로지가 쉽게 변한다는 특성을 갖는다. 이러한 센서 네트워크는 센서를 통한 정보 감지 및 감지된 정보를 처리하는 기능을 수행함으로써 우리의 삶을 자동화시키고 편리함을 제공한다[1-5].

본 연구는 유비쿼터스 센서 네트워크를 이용하여 독거노인 지킴이 시스템을 개발한다. 급격한 사회변화(핵가족화, 산업화)로 인해 '나 홀로 가구'가 되면서 독거노인 문제가 심각한 사회문제화 되고 있다. 이를 해결하는 방안의 하나로 본 사업에서는 유비쿼터스 센서 네트워크를 이용하여 독거노인의 거동을 실시간으로 파악하여 외부기관이나 가족 및 친척에게 유무선으로 정보를 제공하여 위급 시 즉각 조치를 수행할 수 있게 한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문 연구의 필요성을 설명한다. 3장에서는 본 연구의 기반이 되는 연구로써 TinyOS와 NesC에 대해 설명을 한다. 4장에서는 독거노인 지킴이 시스템 구현을 단계별로 설명한다. 5장에서 결론을 맺는다.

2. 연구의 배경 및 필요성

통계청에 따르면 2008년 국내 출산율은 1.19명으로 세계 최저 수준이지만 노령 인구 비율은 상대적으로 급속하게 높아지고 있다. 2009년 9월 현재 국내 65세 이상의 노인 인구는 모두 519만명으로 전체 인구의 10.7%, 2000년 UN 인구 유형 기준으로 고령화 사회에 접어들었다[6]. 더구나 국내 평균 정년 연령이 55세~65세인 점을 감안하여 60세를 기준으로 보면 2020년의 노인 인구비는 20.1%로 전체 인구의 약 1/5이 노인 인구가 될 전망이다.

현재 국내 독거노인(가족없이 혼자 살아가는 노인)은 88만명으로 추산되고 있으며 5명 가운데 1명만 기초생활수급 대상자이다. 나머지 70만명의 독거노인들은 아무리 위급한 상황이 닥쳐도 돌봐줄 사람이 없는 심각한 상황에 처해 있다. 모 방송뉴스에 의하면(MBC TV 2007년 2월 27일자) 서울에 홀로 살던 60대 노인이 숨진 지 1년 만에 발견되었으며, 또 지병을 평소 앓던 60대 독거노인이 자신이 집에서 숨진 지 20여일 만에 발견(광주일보 2007년 3월 10일자), 안양에서는 자식을 그리워하며 우울증을 앓던 60대 독거노인이 숨진뒤 1개월 만에 발견(쿠키뉴스 2007년 2월 5일자)되는 등 인생의 마지막 황혼기에 가장 쓸쓸하고 외로운 죽음을 맞이하고 있어서 사회적으로 큰 충격이 아닐 수 없다. 혼자 사는 노인에 대한 사회적 보조와 관리가 절실히 필요함을 보여주는 사건이다. 앞으로 노령사회가 진전됨에 따라 독거노인 세대가 급속히 증가할 것이다. 이에 따라 적절한 사회적 대처방안이 시급히 요구된다. 만약 현재와 같은 비용 소모적인 노인 복지 정책이 계속된다면 위와 같은 사건은 앞으로 계속될 것이다. 이러한 외로운 죽음 마지하고 있는 독거노인은 주변의 조금만의 관심과 사랑이 있었다면, 그렇게 임종을 맞이하지 않아도 됐을 것이다. 노인 인구는 앞으로도 계속 늘어나기 때문에 이로 인해 발생하는 문제들에 대한 사회적인 대응방안을 강구하지 못한다면 노인문제는 치유하기 어려운 심각한 사태에 이를 수도 있을 것이다. 본 논문에서는 혼자 사는 노인의 일상 생활의 활동을 센서를 통해 분석하여 보다 안전한 생활이 지속될 수 있도록 독거노인 지킴이 시스템을 개발한다.

3. 기반 연구

3.1 TinyOS

TinyOS는 UC 버클리에서 진행해 온 스마트 더스트(smart dust) 프로젝트에 이용하기 위하여 개발된

이벤트 발생 중심의 상태 변화 방식을 채택한 센서 네트워크용 운영체제로써, 동시적인 프로세싱 및 제한된 하드웨어 메모리 공간에서의 효율적인 성능을 지원해주는 운영체제이다[7,8]. 특징은 다음과 같다. TinyOS는 상태 머신의 기반의 구조를 가지는 운영체제로, 각각의 상태는 컴포넌트가 해당된다. 응용 프로그램은 컴포넌트 구현에 독립적인 연결 방법을 사용하여 각각의 TinyOS의 컴포넌트를 연결한다. 그리고, 각 컴포넌트에 명령이 내려지고 이 명령을 처리하는 이벤트 처리기는 그 명령에 따른 상태변화를 신속하게 일으켜, 필요로 하는 일을 수행하는 특징을 가진다. 다음으로는 센서 네트워크의 특징인 저전력 파워 소비를 지원하기 위하여 센서 노드들의 일이 요구되지 않을 경우 저전력 모드인 슬립 모드로 전환함으로써 효율적인 CPU의 사용을 이룰 수가 있는 것이다

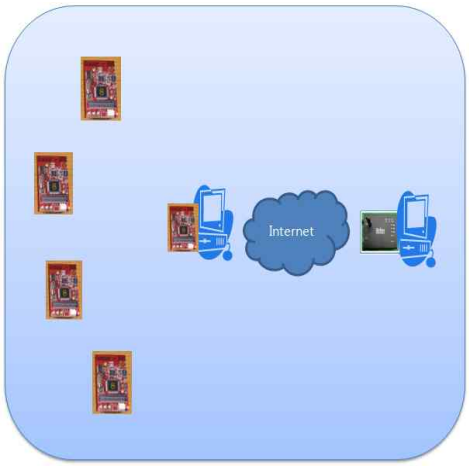
3.2 NesC

NesC(Network Embedded System C)는 컴포넌트 모델 언어로 여러 개의 컴포넌트 블록들을 컴파일 시 연결하여 하나의 어플리케이션 형태를 조합한다. 센서 노드에 로딩되어질 하나의 어플리케이션을 위해 필요한 라이브러리 및 커널 컴포넌트만을 선택하여 컴파일하기 때문에 매우 효과적으로 코드 크기를 줄일 수 있다. C언어와 비슷한 NesC는 응용프로그램의 정의 부분을 나타내는 구성 파일(configuration file)과 응용 프로그램의 실제 구현 부분을 담당하는 모듈 파일(module file)로 구성되어 있다. NesC에서 인터페이스(interface)는 두 컴포넌트를 연결하는 역할을 수행하며 양방향성을 가진다. 두 컴포넌트 사이에 연결 통로(인터페이스)를 연결하는 것을 와이어링(wiring)이라고 한다. TinyOS는 NesC로 구현되어 있다.

4. 독거노인 지킴이 시스템 설계 및 구현

4.1 시스템의 구성

본 논문에서 구현한 독거노인 지킴이 시스템 구현은 센서에 관한 하드웨어 플랫폼과 각 센서 노드에서 원하는 센서 정보를 수집할 수 있는 센서 관련 어플리케이션 프로그램, 센싱 정보를 취합하여 관리자에게 전송하여 실시간으로 독거노인 상황을 파악하는 어플리케이션 프로그램으로 구성하였다.



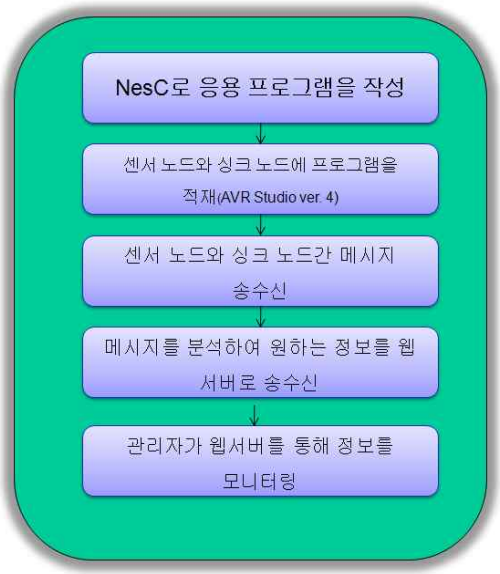
<그림 1> 독거노인 지킴이 시스템의 구성도

<그림 1>은 독거노인 지킴이 시스템의 전반적인 구성도를 나타낸다. 온도, 조도, 습도, 바이오센서 등을 센싱 할 수 있는 센서 노드(sensor node)와 센서 노드에서 감지된 센싱 데이터를 취합하여 시스템에게 정보를 전송시키는 싱크노드(sink node), 싱크노드에 정보를 획득하여 공중파로 전송시키는 시스템이 있으며, 공중파로 전송 되어지는 센서 정보를 관리자에게 제공할 수 있는 독거노인 지킴이 관리 서비스(어플리케이션 프로그램)를 제공하는 통합 서버가 있다.

센서 노드는 ZigbeX를 사용하였으며, 주요 사항은 다음과 같다. 센서 노드의 마이크로컨트롤러(MCU)는 Atmel사에서 나온 8-bit 마이크로 컨트롤로러 ATmega128L이 사용되었다[9]. ATmega128L은 하버드 구조를 가지는 고성능의 8비트 RISC형 마이크로 컨트롤러이며, 프로그램용 메모리로서 대용량의 플래시 메모리를 내장하고 있으며 이것을 사용자가 쉽게 여러 번 반복적으로 프로그래밍 할 수 있는 기능을 가지고 있어서 매우 편리하다. IEEE 802.15.4 표준을 지원하는 RF 송수신 칩으로 Chipcon사의 CC2420이 있으며, 2.4GHz용 PCB 안테나를 기본으로 장착하고 있다. 센서로는 온도 및 습도, 조도를 감지하는 센서가 있다.

센서 노드와 싱크 노드간의 센싱 정보를 송수신하고, 센싱 정보가 최종적으로 통합 서버로 전송되어 분

석하기 위해서는 <그림 2>와 같이 크게 다섯 단계의 작업을 통해 이루어진다.



<그림 2> 독거노인 시스템 구현 단계

먼저, 센서 정보를 얻기 위해 NesC로 센서를 위한 응용 프로그램을 작성한다. 작성한 응용 프로그램을 센서 노드와 싱크 노드에 적재시킨다. 그 이후 전원을 작동시키면 센서 노드와 싱크 노드간에 메시지가 송수신된다. 전송되어진 메시지에서 원하는 정보만을 추출하여 서버로 보낸다. 최종적으로 서버를 통해 관리자가 원격지에서 모니터링하게 된다.

4.2 시스템의 구현

본 논문에서는 독거노인 지킴이 시스템을 구현하기 위해 독거노인의 각 방과 거실에 온도, 습도, 조도 센서, 바이오 센서 노드를 부착하여 독거노인의 실시간적인 상황을 파악한다. <그림 3>은 온도, 습도, 조도 센서 구성 파일을 나타낸다. <그림 4>는 온도, 습도, 조도 센서 모듈 파일을 나타낸다.

센서 노드에서 온도와 습도를 측정하는 센서는 SHT11이고, 조도는 CDS 센서가 사용된다. SHT11 센서는 자체적으로 측정한 아날로그 신호를 디지털 신호로 바꾸어주는 ADC 기능이 있어 센싱한 데이터를 바로 CPU로 전송할 수 있다. TinyOS는 SHT11 온도와 습도 센서를 제어하기 위해 HumidityC 컴포턴트를 제공하고 있다 HumidityC 컴포넌트는 하위 컴포넌트인

```
implementation
{
  components Main, TestBeat_ThermoM, GenericComm as Comm,
Thermopile, LedsC, ADCC, TimerC;

  Main.StdControl -> TimerC;
  Main.StdControl -> TestBeat_ThermoM;
  TestBeat_ThermoM.CommControl-> Comm;

  TestBeat_ThermoM.Leds -> LedsC;

  TestBeat_ThermoM.ADCControl -> ADCC;
  TestBeat_ThermoM.BeatADC -> ADCC.ADC[TOS_ADC_BEAT_PORT];

  TestBeat_ThermoM.BioThermopileControl -> Thermopile.StdControl;
  TestBeat_ThermoM.BioThermopile -> Thermopile.BioThermopile;

  TestBeat_ThermoM.Timer -> TimerC.Timer[unique("Timer")];
  TestBeat_ThermoM.ResetCounterMsg ->
Comm.ReceiveMsg[AM_OSCOPERESETMSG];
  TestBeat_ThermoM.Send -> Comm.SendMsg[AM_OSCOPEMSG];
}
```

<그림 3> 온도, 습도, 조도 센서 구성 파일

```
module TestBeat_ThermoM {
    provides interface StdControl;
    uses {
        interface Leds;
                interface Timer;
        interface ADCControl;
                interface StdControl as CommControl;
        interface ADC as BeatADC;
        interface StdControl as BioThermopileControl;
        interface ADC as BioThermopile;
                interface SendMsg as Send;
                interface ReceiveMsg as ResetCounterMsg;
    }
}
implementation {
  // declare module static variables here
  ...........

} command result_t StdControl.init() {
.......
  }

command result_t StdControl.start(){
    ......
```

<그림 4> 온도, 습도, 조도 센서 모듈 파일

HumidityProtocolC의 도움으로 습도 측정값을 얻을 수 있는 Humidity.getData() 함수와 온도 측정값을 얻을 수 있는 Temperature.getData() 함수를 제공한다. 두 함수를 이용하여 SHT11 센서에 온도 및 습도 측정값을 요청하며, 이벤트 형태의 함수인 Humidity.dataReady(unit16_t data)와 Temperature.dataReady (unit16_t data)를 통해 습도와 온도 값을 추출한다 <표 1>.

조도를 측정할 수 있는 CDS 센서는 주변의 광량에 따라 자신의 저항값이 변하므로 INT0으로부터 들어오는 3V 전압은 광량에 따라 변화하는 CDS의 저항 성분에 의해 영향을 받는다. 따라서 출력 포트인 ADC0에서는 변화된 전압의 양에 따라서 광량을 감지할 수 있다. TinyOS에서는 조도 센서를 제어하기 위해

<표 1> 센서에서 습도와 온도 정보 추출 함수

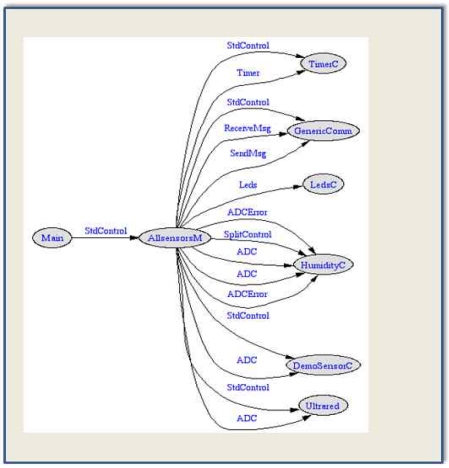
함수	내용
Humidity.getData()	ADC 포트로부터 측정된 습도값을 얻기 위해 호출
event Humidity.dataReady (unit16_t data)	ADC가 측정한 습도 값을 이벤트 형태로 반환
Temperature.getData()	ADC 포트로부터 측정된 온도값을 얻기 위해 호출
Temperature.dataReady (unit16_t data)	ADC가 측정한 온도 값을 이벤트 형태로 반환

Photo 컴포넌트를 제공하고 있다. Photo 컴포넌트에서는 StdControl.init() 함수에 의해 노드가 초기화를 시작할 때 ADCControl.bindPort(TOS_ADC_PHOTO_PORT, TOSH_ACTUAL_PHOTO_PORT) 함수를 호출하여 INT0의 전압을 활성화시켜 조도 센서에 의한 광량값을 측정한다. Photo 컴포넌트를 사용하는 상위 프로그램에서는 조도 데이터를 얻기 위해 ADC.getData() 함수를 호출한다. ADC.getData() 함수가 호출되면 CPU의 INT0와 ADC0 사이에 있는 CDS 센서의 의해 조도 정보를 얻게 되고, 그 값은 이벤트 함수로 반환되는 ADC.dataReady(unit16_t data) 함수를 통해 상위 컴포너트에게 전달된다.

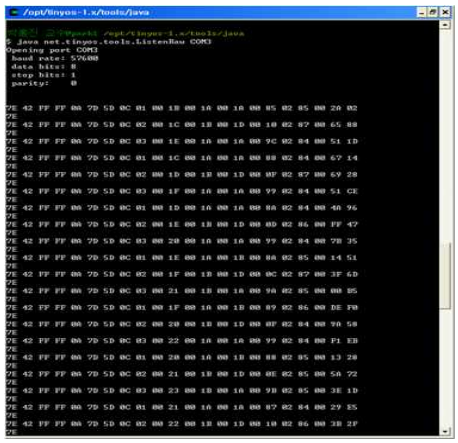
싱크 노드와 서버간의 무선 통신을 위해 GenericComm 컴포넌트와 조도 센서 값을 얻기 위해 DemoSensorC 컴포넌트, 프로그램은 주기적인 동작 여부를 위해 TimerC 컴포넌트, 동작여부를 확인하기 위한 LedsC 컴포넌트가 사용되었다.

<그림 5>는 본 논문에서 구현한 온도, 습도, 조도에 관련된 컴포넌트 연관 그래프를 나타낸다. 가장 상위 컴포넌트인 Main 컴포넌트는 AllsensorsM 컴포넌트를 호출하고 AllsensorsM 컴포넌트 아래 <그림 5>에 나타난 것 처럼 7개의 하위 컴포넌트로 구성되어 있다.

<그림 6>과 <그림 7>은 센서 노드에 전송되고 있는 실제 메시지들이다. 센서 노드는 TimerC 컴포넌트에 의해 125ms마다 Timer.fired()함수를 호출함으로써 온도, 조도, 습도 정보 값을 얻는다. 즉, 125ms 마다 센서 노드에서 싱크 노드로 메시지가 패킷 형태로 무선 통신으로 전송시키며, 싱크노드는 이 정보를 서버로 전송시킨다. 서버에서는 일정한 시간마다 전송되어 진 패킷 값을 분석하여 온도, 조도, 습도 값을 추출하는 것이다.



<그림 5> 온도, 습도, 조도 컴포넌트 연관 그래프



<그림 6> 센서 노드에서 전송되는 메시지(1)

7E 42 FF FF 0A 7D 5D 0C 01 00 0B 04 **1A 00 1D 00 61 02 87 00** 6B EA 7E
7E 42 FF FF 0A 7D 5D 0C 01 00 0C 04 **1A 00 1D 00 5C 02 88 00** C0 F3 7E
7E 42 FF FF 0A 7D 5D 0C 01 00 0D 04 **1A 00 1D 00 5C 02 86 00** 8A BF 7E
.....

<그림 7> 센서 노드에서 전송되는 메시지(2)

<표 2> 온도, 습도, 조도 정보 메시지 분석

바이트 수	예(16진수)	의미
1 byte	7E	Start Byte
1 byte	42	NonAck
2 bytes	FF FF	UARTADDR
1 byte	0A	MSGTYPE
1 byte	7D	GroupID
1 byte	0C	MsgLen
2 bytes	01 00	Src
2 bytes	0C 04	seq
2 bytes	**1A 00**	온도 정보
2 bytes	**1D 00**	습도 정보
2 bytes	**61 02**	조도(Photo) 정보
2 bytes	**87 00**	적외선 정보
2 bytes	6B EA	CRC
1 byte	7E	EndByte

<표 2>는 센서 노드에서 전송되는 메시지를 분석한 표이다. <표 2>에서 GroupID는 센서 노드의 그룹(group) 식별 번호이다. 센서 노드가 여러 개 있고, 이들과 무선 통신하는 싱크 노드가 하나가 있다. 여러 개 있는 센서 노드를 같은 그룹으로 지정해주어야 싱크 노드가 다수의 센서 노드를 인식할 수 있다. MsgLen 값(0C)은 사용자 정의 데이터 바이트 수이다. MsgLen 값 뒤에 CRC 전까지 바이트의 수는 13개(0C)임을 알 수 있다. 사용자 정의 데이터에서 온도 습도 조도 정보가 있다. CRC 값은 메시지 데이터의 정확성을 위해 CRC16 체크를 하는 부분이다.

본 논문에서는 온도, 조도, 습도 센서 노드 뿐만 아니라, 바이오 센서를 이용하여 맥박과 체온 정보를 수집한다. 바이오 센서 응용 프로그램은 <그림 8>과 <그림 9>에 있다. <그림 8>은 바이오 센서의 맥박과 체온 데이터를 수집하기 위한 구성 파일이다.

<그림 8>에서는 바이오 정보를 얻기 위해 여러 컴포넌트들을 선언하게 된다. 센서 노드에서 무선 통신을 위해 GenericCom 컴포넌트와 맥박과 체온 정보를 위한 TestBeat_ThemoM 컴포넌트 등이 있다. <그림 9>는 <그림 8>의 구성 파일이 실제 동작 부분을 기술한 모듈 파일이다. 센서 노드에서 수행되는 StdConrol.start() 함수에서 0.125초 마다 BeatADC.getData()함수와 BioThermopile.getData() 함수를 호출함으로써 맥박과 체온 센싱 정보를 획득하여 베이스 노드에 전송된다.

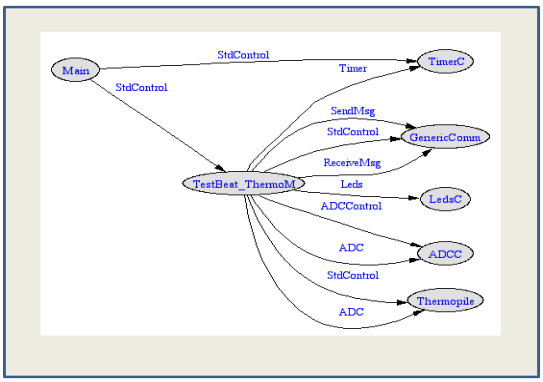
```
implementation
{
  components Main, TestBeat_ThermoM, GenericComm as Comm,
Thermopile, LedsC, ADCC, TimerC;

  Main.StdControl -> TimerC;
  Main.StdControl -> TestBeat_ThermoM;
  TestBeat_ThermoM.CommControl->Comm;

  TestBeat_ThermoM.Leds -> LedsC;

  TestBeat_ThermoM.ADCControl -> ADCC;
  TestBeat_ThermoM.BeatADC -> ADCC.ADC[TOS_ADC_BEAT_PORT];

  TestBeat_ThermoM.BioThermopileControl -> Thermopile.StdControl;
  TestBeat_ThermoM.BioThermopile -> Thermopile.BioThermopile;

  TestBeat_ThermoM.Timer -> TimerC.Timer[unique("Timer")];
  TestBeat_ThermoM.ResetCounterMsg ->
Comm.ReceiveMsg[AM_OSCOPERESETMSG];
  TestBeat_ThermoM.Send -> Comm.SendMsg[AM_OSCOPEMSG];
}
```

<그림 8> 바이오 센서의 구성 파일

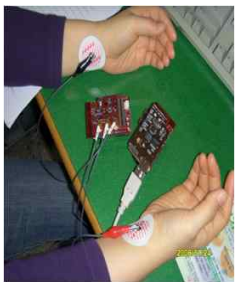
```
module TestBeat_ThermoM {
    provides interface StdControl;
    uses {
        interface Leds;
                interface Timer;
        interface ADCControl;
                interface StdControl as CommControl;
        interface ADC as BeatADC;
        interface StdControl as BioThermopileControl;
        interface ADC as BioThermopile;
                interface SendMsg as Send;
                interface ReceiveMsg as ResetCounterMsg;
    }
}
implementation {
  // declare module static variables here
  ..........

} command result_t StdControl.init() {
.....
  }

command result_t StdControl.start(){
    .....
```

<그림 9> 바이오 센서의 모듈 파일

<그림 10>은 본 논문에서 사용된 바이오 센싱에 관련된 컴포넌트 연결 관계도를 나타낸다. Main 컴포넌트 하위에 TestBeat_ThemoM 컴포넌트가 다른 컴포넌트보다 상위에 있다. 즉, 다른 컴포넌트는 TestBeat_ThemoM 컴포넌트를 상속받아서 수행하고 있다.

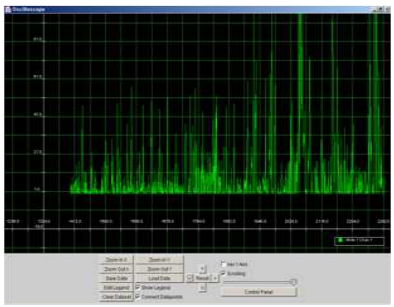


<그림 10> 바이오 센서에서 컴포넌트 연결 관계도

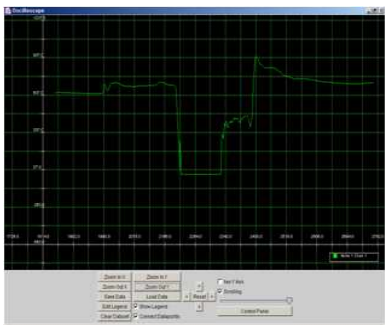


<그림 11> 맥박 센서 측정

<그림 12> 체온 센서 측정



<그림 13> 맥박 센싱 정보



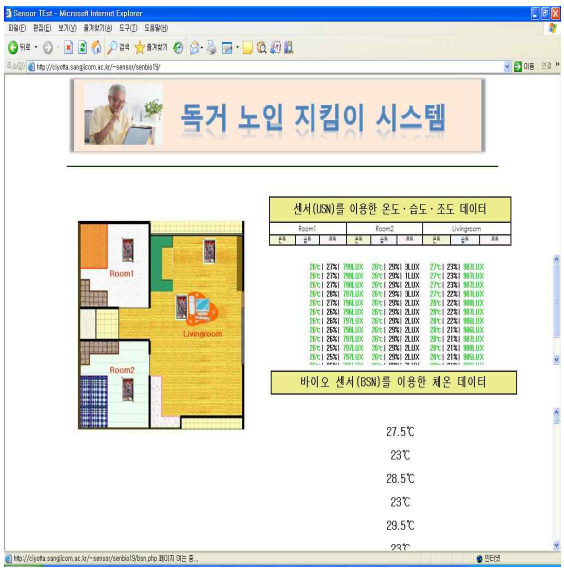
<그림 14> 체온 센싱 정보

<그림 11>은 맥박 센서로 측정을 나타내며, <그림 12>는 체온 센싱 정보를 측정하고 있다. <그림 11>에 나타난 것처럼 인체에 연결할 선은 3개로 구성되어 있으며 바이오의 MCX도 3개로 구성되어 있고 이를 두 팔과 오른쪽 발에 부착하여 맥박을 측정하게 된다. 체온 센싱 정보는 <그림 12>처럼 센서 노드를 인체에 가까이 위치시켜 체온을 감지할 수 있도록 한다.

<그림 13>은 맥박 센싱 정보에서 얻은 정보가 오실로스코프를 이용하여 측정된 결과를 나타내고 있다. <그림 13>에서 보여지 듯 그래프 모양이 맥박에 따라 주기적으로 고저가 있음을 알 수 있다. <그림 14>는 체온 센싱 정보에서 얻은 정보가 오실로스코프를 이용하여 측정된 결과를 나타내고 있다.

서버로 전송되어지는 메시지는 네트워크를 통해 통합 관리 시스템으로 전송되어진다. 통합 관리 시스템을 통해 관리자는 독거노인 지킴이 서비스의 전반적인 상황을 파악할 수 있다.

4.3 시스템 구현 결과



<그림 15> 독거노인 지킴이 시스템

<그림 15>는 통합 관리 시스템에서 독거노인 지킴이 시스템을 구현한 화면이다. 2개의 방에 센서 노드가 있고, 거실에도 센서 노드가 있다. 또한 거실에서는 싱크 노드와 서버가 있다. 각 방과 거실에 있는 센서 노드의 센싱 정보가 싱크 노드로 전송되고, 싱크 노드는 서버로, 서버에 전송되는 정보는 통합 관리 시스템에 전송되어 진다. <그림 15>를 보면 방 2개와 거실의 습도는 20% ~ 30%정도로 낮다. 또한 방 2는 조도가 10 lux 이하로 낮음을 알 수 있고, 이 는 방에 불이 꺼져 있음을 알 수 있다. 방 1의 조도는 700 lux ~ 800 lux 정도로 방에 불이 켜져 있는 조도 값이다. 거실에는 900 lux 이상으로 밖의 햇볕이 거실에 들어오는 조도 값이다. 온도는 20℃~ 30℃ 이하로 일반적인 방과 거실의 온도이다. 만약 장시간동안 온도가 낮고 조도 값이 낮으면 독거노인의 상태가 정상적이지 못함을 예상 할 수 있을 것이다.

6. 결 론

본 논문은 혼자 사는 노인의 일상 생활의 활동을 센서를 통해 분석하여 보다 안전한 생활이 지속될 수 있도록 독거노인 지킴이 시스템을 개발하였다. 개발된 시스템은 TinyOS 기반의 NesC로 구현하였다.

본 논문의 장점은 온도 센서, 습도 센서, 조도 센서, 바이오 센서(체온 센서)에서 센싱된 정보를 웹 환경에서 실시간으로 모니터링 할 수 있다는 것이다. 통합 관리자는 독거노인이 거주하는 방이나 거실에 설치된 센서의 센싱된 정보를 웹을 통해 모니터링 함으로써 언제 어디서나 손쉽게 접근함은 물론 독거노인 상황을 실시간으로 파악할 수 있다. 예를 들어 방안에 있는 온도 센서의 값이 너무 낮으면, 방 안이 매우 춥다는 것을 의미하고, 방 안의 조도 센서의 값이 낮으면, 불이 꺼져 있음을 나타냄으로써 정상적이거나 비정상적인 상태를 본 논문에서 개발한 시스템을 통해 파악할 수 있다.

향후 연구로는 본 논문에서 개발된 시스템에 위치 추적 센서 정보를 추가함으로써 독거노인의 비정상적인 활동이 나타날 경우 현재 위치를 실시간으로 파악함으로써 독거노인에게 보다 실질적인 도움을 줄 수 있을 것이다.

참 고 문 헌

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey", Computer Networks, vol. 38, issue 4, pp. 393-422, 2002.

[2] Yu-Chee Tseng, Wen-Chih Peng, Victor C.M. Leung, Wen-Tsuen Chen, and M. Cristina Pinotti, "Information processing and data management in wireless sensor networks", Signal Processing, vol. 87, issue 12, pp. 2859-2860, Dec. 2007.

[3] Aleksandar Milenković, Chris Otto, and Emil Jovanov,, "Wireless sensor networks for personal health monitoring: Issues and an implementation", Computer Communications, vol. 29, issue 13-14, pp. 2521-2533, Aug. 2006.

[4] Bert Gyselinckx, Julien Penders, and Ruud Vullers, "Potential and challenges of body area networks for cardiac monitoring", Journal of Electrocardiology, vol. 40, issue 6, pp. 165-168, Nov. 2007.

[5] Gershman, A., "Ubiquitous commerce - always on, always aware, always proactive", Proceedings of SAINT 2002. pp.37-38, 2002.

[6] http://www.nso.go.kr/

[7] 박승민, "센서 네트워크 노드 플랫폼 및 운영체제 기술 동향", 전자통신동향분석, 제21권, 제1호, 2006.

[8] http://www.tinyos.net

[9] http://www.atmel.com

박 홍 진 (Hong-Jin Park)

- 정회원
- 1993년 2월 : 원광대학교 컴퓨터공학과 (공학사)
- 1995년 8월 : 중앙대학교 컴퓨터공학과 (공학석사)
- 2001년 8월 : 중앙대학교 컴퓨터공학과 (공학박사)
- 2001년 9월 ~ 현재 : 상지대학교 이공과대학 컴퓨터정보공학부 부교수
- 관심분야 : 분산시스템, 센서 네트워크, 모바일 시스템

논 문 접 수 일 : 2010년 02월 02일
1차수정완료일 : 2010년 03월 02일
2차수정완료일 : 2010년 04월 12일
게 재 확 정 일 : 2010년 04월 13일