무선 센서 네트워크를 위한 운영체제들의 성능분석

민 홍°, 이상호, 구본철, 허준영, 김용태, 조유근
서울대학교 컴퓨터공학부
{hmin°, shyi, bcgu, jyheo, ytkim, cho}@ssrnet.snu.ac.kr

홍지만
광운대학교 컴퓨터공학부
gman@daisy.kw.ac.kr

Performance Analysis of Operating Systems for Wireless Sensor Networks

Hong Min°, Sangho Yi, Boncheol Gu, Junyoung Heo, Yongtae Kim, Yookun Cho
School of Computer Science and Engineering, Seoul National University

Jiman Hong
School of Computer Science and Engineering, Kwangwoon University

요 약

무선 센서 네트워크의 사용분야가 넓어짐에 따라 다양한 종류의 플랫폼을 지원하고 센서 노드의 제한된 자원을 효율적으로 사용할 수 있도록 관리하는 센서 운영체제들에 대한 연구가 활발하게 진행되었다. 본 논문에서는 기존의 운영체제들의 특징과 장단점을 분석해 보고, 태스크 관리와 메모리 관리 측면에서 성능 평가를 수행했다. 이러한 정보를 바탕으로 더욱 효율적인 시스템 설계를 위한 대안을 제시한다.

1. 서 론

오늘날 무선 센서 네트워크는 새롭고 중요한 연구 분야로 주목 받고 있다[1]. 센서 노드는 사무실, 공장 및 그보다 혹독한 환경에서 주변 상황을 모니터링하고 필요한 정보를 수집하는 용도로 사용되었다[2]. 최근 센서 노드 제조 기술의 발달로 저전력, 저비용으로 다양한 기능을 지원하는 노드들이 개발되고 있으며[3], 그로 인해 다양한 플랫폼을 갖는 센서 노드들이 개발되었고, 홈 네트워크, 자연 환경, 건강관리 및 다양한 종류의 임베디드 시스템[4] 등, 센서 노드가 사용되는 분야도 점점 넓어지고 있다. 센서 노드의 활용 분야가 다양해짐에 따라 자원을 효율적으로 관리하는 것과 응용 프로그램의 개발 속도를 향상 시키는 것이 중요한 이슈가 되고 있다. 이를 위해 다양한 플랫폼을 지원하고 자원을 효율적으로 관리하며, 응용 프로그램을 개발할 수 있는 환경을 제공하는 무선 센서 네트워크를 위한 운영체제들이 연구 되고 있다.

Berkeley 대학에서 개발한 TinyOS는 다양한 센서에 대한 드라이버와 응용 프로그램을 프로그래밍하고 테스트하는 환경을 제공하고 있다[6,8]. MANTIS는 Colorado 대학에서 개발한 운영체제로 멀티 쓰레드 기반으로 동작하며, 메모리 사용량이 적다는 특징을 가지고 있다. 그리고 SOS는 센서 네트워크 상에서 동적으로 프로그램을 업데이트 할 수 있다는 특징을 가지고 있다[5]. 우리나라에서도 2005년 전자통신연구원에서 Nano-Qplus라는 센서 노드용 운영체제를 개발하였다[3].

본 논문에서는 앞에서 언급했던 센서 노드를 위한 운영체제들의 기능을 전반적으로 살펴보고 태스크 스케줄러와 메모리 관리에 대한 실험을 통해서 각각의 운영체제들의 장단점을 분석해본다. 이러한 정보는 앞으로 더욱 효율적인 센서 운영체제를 설계하는데 도움을 줄 것이다.

본 논문의 구성은 다음과 같다. 2절에서는 센서 운영체제를 설계할 때 고려해야 할 기본적인 기준을 제시한다. 3절에서는 이전에 연구되었던 네가지 센서 운영체제에 대한 특징과 장단점을 살펴본다. 4절에서는 태스크 생성과 메모리 관리에 필요한 함수들의 지연시간을 측정하여 결과를 분석하고 5절에서는 결론을 맺는다.

2. 센서 운영체제의 요구사항

무선 센서 네트워크를 위한 운영체제에서 가장 먼저 고려해야할 사항은 메모리 공간과 에너지 사용량의 제약성이다. 센서 노드의 생산 비용 절감을 위해 작은 크기의 메모리를 사용하고, 전원 공급은 대부분 배터리를 통해 이루어지기 때문이다. 다음은 무선 센서 네트워크를 위한 센서 운영체제를 설계하는데 있어서 고려해야할 디자인 요소에 대해 나열한 것이다.

- 에너지 사용의 효율성
- 작은 메모리 공간
- 다양한 센서의 지원
- 동적 프로그램 재구성
- 실시간 작업 처리

먼저 에너지 사용의 효율성과 메모리 공간의 제약은 앞에서 언급했듯이 가장 중요한 이슈이다. 다음으로 센서 노드들은 온도, 조도, 습도 등과 같이 다양한 종류의 센서 장비를 지원해야 한다. 또한 이미 응용프로그램을 탑재하여 배치된 센서 노드들을 필요에 따라 동적으로 재컴파일 하거나 재구성하는 작업을 지원해야 한다. 마지막으로 센서 노드들의 작업의 특성상 실시간으로 작업 처리가 이루어 질 수 있도록 해야 한다.

3. 센서 운영체제

본 장에서는 지금까지 연구된 네가지 센서 운영체제들에 대한 특징을 정리하고 장단점을 살펴본다. 다음은 센서 운영체제들의 특징을 표로 정리한 것이다.

특징	센서 운영 체제			
	TinyOS	SOS	MANTIS	Nano-Qplus

저전력 모드 지원	Y	Y	N	Y
다중모드지원	N	Y	Y	Y
동적 재프로그래밍	Y	Y	N	N
우선순위 스케줄링	N	N	Y	Y
실시간 보장	N	N	Y	Y
실행 모델	컴포넌트 베이스	모듈 베이스	쓰레드 베이스	쓰레드 베이스

[표 1] 센서 운영체제들의 특징 비교

3.1 TinyOS

TinyOS는 가장 널리 사용되는 센서 운영체제로 컴포넌트 기반과 이벤트 드리븐 방식의 실행 모델을 사용한다[8]. 이러한 방식은 응용 프로그램을 구현할 때 코드의 사이즈를 최소화함으로써 개발 기간을 단축시키고 메모리 사용량을 줄일 수 있다는 장점이 있다. 정교한 전원관리 시스템은 에너지를 효율적으로 사용할 수 있게 해준다. 그러나, 다중모드를 지원하지 않기 때문에 한번에 한 작업만을 처리할 수 있으며, 실시간 스케줄링에 대한 고려가 없기 때문에 실시간 시스템으로는 적합하지 않다.

3.2 SOS

SOS 커널은 메시지 처리, 동적 메모리 할당, 모듈의 로드와 언로드, 그 밖에 다른 서비스를 제공하는 부분으로 구성되어 있다[7]. 모듈은 네트워크를 통해서 동적으로 추가, 수정 및 제거가 가능하다. 센서 네트워크에서 배치된 노드들을 네트워크를 통하여 응용 프로그램을 업데이트 하는 것은 중요하다. 상황과 연구 목적에 따라 사용하는 응용 프로그램이 다르며, 지리적으로 접근하기 어려운 곳에 배치되어 있는 노드들을 직접 수거하는 것이 어렵기 때문이다. SOS는 이러한 기능이 가장 효율적으로 이루어 질 수 있는 구조로 설계되었다. 그러나 모듈의 수행이 독립적으로 이루어지고, 이를 전체적으로 관리하는 실시간 스케줄러가 없기 때문에 실시간 시스템을 지원하지 않는다.

3.3 MANTIS

MANTIS는 쓰레드 기반의 센서 운영체제로 선점 가능한 다중 쓰레드 작업을 지원한다[1]. 또한 복잡한 작업을 수행하는 태스크와 시간 제약성을 갖는 태스크가 적절하게 상호 배치될 수 있도록 스케줄링한다. 다시 말해서 긴 수행시간을 필요로 하는 작업이 빠른 시간 내에 수행을 맞춰야하는 작업의 수행을 가로막지 않도록 해준다. 이벤트 드리븐 방식의 TinyOS는 이러한 문제를 해결하지 못하지만 MANTIS는 멀티 쓰레드라는 개념을 사용하여 이를 해결하고 있다.

3.4 Nano-Qplus

Nano-Qplus는 센서 네트워크의 두 가지 요구조건을 만족시키기 위해 설계되고 개발되었다. 첫 번째는 확장성과 시스템 재구성의 용이성이다. 앞에서 언급했던 센서 운영체제들은 다양한 환경 요인과 하드웨어 플랫폼을 지원하지 못하기 때문에 사용하기 어려운 문제점을 가지고 있다. Nano-Qplus에서는 다양한 하드웨어를 Nano -HAL (Hardware Abstraction Layer)를 통해 동일한 시스템 모델로 간주한다. 따라서 프로그래머가 하드웨어에 대한 고려 없이 의도하는 대로 프로그램을 쉽게 작성할 수 있다. 두 번째는 쓰레드 기반의 우선순위 스케줄러 이다. MANTIS와 같이 우선순위에 따라 선점 가능한 스케줄러를 사용하기 때문에 실시간 시스템을 지원한다.

4. 성능 분석

본 장에서는 센서 운영체제의 성능 평가를 위한 기준을 제시하고 이를 바탕으로 4가지 운영체제를 평가하여 성능을 분석할 것이다. 이러한 정보를 바탕으로 무선 센서 네트워크를 위한 운영체제들의 성능 향상을 위한 대안들을 제안하고자 한다.

4.1 실험 환경

실험을 위하여 옥타컴에서 제작한 Nano-24 무선 센서 네트워크 플랫폼을 사용하였고 상세한 스펙은 다음 표와 같다[7].

Component	Model	Description
CPU	ATmega128L	low-power 8bit microprocessor 8 Mhz clock
Memory	Flash SRAM EEPROM	128 KB 4 KB 4 KB
RF	CC2420	2.4 Ghz channel Zigbee support 250 Kbps rate
Power	2xAA Batteries	3.0 Volt

[표 2] Nano-24 플랫폼 스펙

옥타컴에서 TinyOS와 Nano-Qplus를 위한 커널을 제공하지만 SOS와 MANTIS의 커널은 제공하지 않기 때문에 Nano-24 보드에 맞게 포팅해서 실험했다.

4.2 평가 기준

센서 운영체제의 커널은 태스크 스케줄링과 메모리 관리를 담당하는 부분으로 나눌 수 있다. 이 부분의 처리 속도가 전체 운영체제의 성능을 좌우하기 때문에 성능 평가 기준을 다음과 같이 정했다.

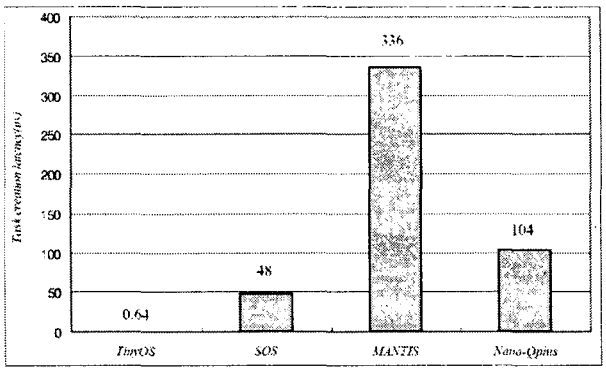
- 태스크 생성 지연시간
- 메모리 할당 지연시간
- 메모리 관리 함수 처리시간

태스크 생성 시간은 태스크 스케줄러의 부하를 측정하기 위해서 필요한 실험이고, 메모리 할당과 메모리 관리 함수 처리 시간은 메모리를 관리가 효율적으로 이루어지고 있는지 여부를 확인하기 위해 필요한 실험이다.

4.3 실험 결과 및 평가

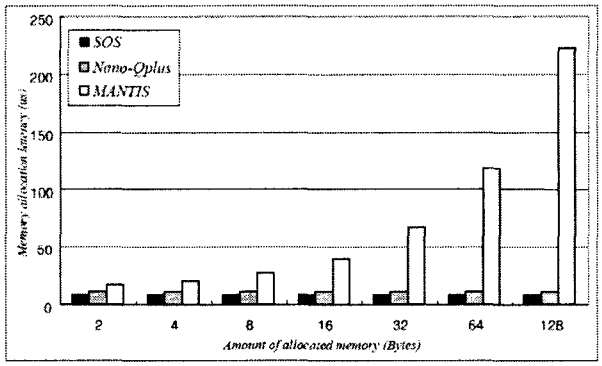
그림 1은 운영체제 별로 태스크 생성시간을 측정한 것이다[1]. 각 운영체제 별로 시스템 구조상의 특징이 다르기 때문에 태스크 생성 시간에 차이가 있었다. TinyOS의 경우 FIFO 스케줄러를 사용하며, 수행할 함수의 포인터를 레디큐에 삽입함으로써

1) 운영 체제별 버전 : TinyOS: 1.1.11-3, SOS: 05-july, MANTIS: 0.9.1b, Nano-Qplus: 1.6.0e



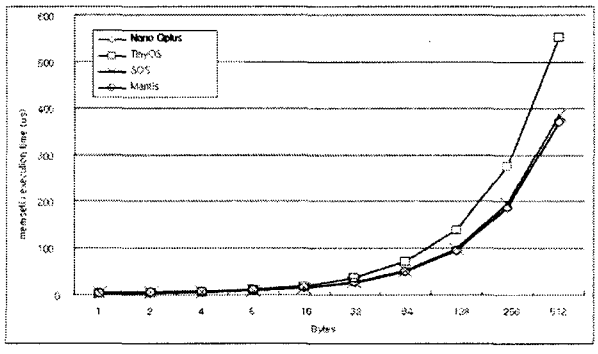
[그림 1] 운영체제별 태스크 생성 지연시간

새로운 태스크를 생성하기 때문에 수행시간이 짧다. 다른 운영체제들은 새로운 작업을 생성하기 위해서 필요한 메모리 공간을 확보해야 하기 때문에 생성 시간이 더 오래 걸린다.



[그림 2] 운영체제별 메모리 할당 지연시간

그림 2는 운영체제 별로 메모리 할당 지연 시간을 측정한 것이다. TinyOS의 경우는 메모리 할당 함수를 지원하지 않기 때문에 실험에서 제외 시켰다. SOS는 정적 메모리 할당을 사용하여 사전에 정의된 크기에 따라 메모리 주소를 리턴해주기 때문에 수행시간이 짧다. Nano -Qplus와 MANTIS의 경우 동적으로 메모리를 할당하는데 MANTIS의 경우 할당된 메모리를 초기화하는 작업을 추가로 해주기 때문에 사이즈가 증가함에 따라서 수행시간이 길어진다.



[그림 3] 메모리 초기화 속도

그림3은 메모리 관리 함수로 memset()의 수행 시간을 측정한 것이다. TinyOS의 경우 자체적으로 memset() 함수를 구현하였으나 기계어를 사용하여 구현한 다른 운영체제들 보다 성능이 떨어진다는 것을 알 수 있다. 다른 메모리 관리 함수인 memcy()에서도 비슷한 결과를 얻을 수 있었다.

4.4 성능 향상을 위한 대안 제시

TinyOS의 경우 우선순위에 따른 선점 스케줄러를 지원하지 않기 때문에 태스크 생성 시간을 빠르지만 실시간 시스템을 지원하지 못한다. Nano-Qplus와 같은 쓰레드 기반의 운영체제는 선점 가능한 스케줄러의 지원으로 실시간 작업을 보장할 수 있지만 태스크의 생성시간이 오래 걸린다. 따라서 센서에서 수행하는 작업의 특성에 맞게 빠르게 작업을 생성하면서도 우선순위에 따라 선점이 이루어지는 하이브리드 스케줄러에 대한 연구를 제안한다. 메모리 관리에 있어서도 동적인 메모리 할당을 통해서 단편화를 줄이고, 최소한의 공간을 사용하여 메모리를 관리하는 기법에 대한 연구가 필요하다.

5. 결 론

무선 센서 네트워크에서 센서 노드의 에너지 사용의 효율성과 메모리 공간의 절약은 중요한 요소이다. 본 논문에서는 이러한 기준을 바탕으로 현재까지 연구된 센서 운영체제들의 특성과 장단점을 살펴보았다. 또한 태스크 스케줄러와 메모리 관리 함수들에 대한 수행 시간 측정을 통해서 각 운영체제들의 성능을 비교해보고, 운영체제를 향상 시킬 대안을 제안했다.

향후 제안한 대안을 바탕으로 센서 노드의 자원을 효율적으로 관리할 수 있는 시스템 구조를 설계하고, 실시간 시스템을 지원하면서도 빠르게 태스크를 생성할 수 있는 태스크 스케줄러에 대한 연구를 지속할 것이다.

[참고문헌]

[1] Bhatti, S., Carlson, J., Dai, H., Deng, J., Rose, Mantis os: An embedded multithreaded operatingsystem for wireless micro sensor platforms. ACMKluwer Mobile Networks and Applications(MONET) Journal, Special Issue on Wireless Sensor Networks, 2005

[2] Shah, R., Rabaey, J., Energy aware routing for low energy ad hoc sensor networks.In, Proc. IEEE Wireless Communications and Networking Conference(WCNC), 2002

[3] Lee, K., Shin, Y., Choi, H., Park, S., A design of sensor network system based on scalable and reconfigurable nano-os platform, IT-Soc International Conference, 2004

[4] Srivastava, M., Muntz, R., Potkonjak, M., Sensor-based wireless networks for smart developmental problem-solving environments, The 7th Annual International Conference on Mobile Computing and Networking, 2001

[5] Han, C.C., Kumar, R., Shea, R., Kohler, E., Srivastava, A dynamic operating system for sensor nodes, 2005, 163-176

[6] Levis, P., Madden, S., Gay, D., Polastre, J., Szewczyk, R., The emergence of networking abstractions and techniques in tinyos, First USENIX/ACM Symposium on Networked Systems Design and Implementation, 2004

[7] Octacomm: http://www.octacomm.net/, 2005

[8] Martin, F., Mikhak, B., Silverman, B., A designer's kit for making computational devices, IBM System Journal 39, 2000