SenOS: 동적 센서 노드 재구성을 위한 상태 기반 운영 체제 구조

요 약
무선 센서 네트워크의 동작 환경과 구조가 매우 특이하여 개발자들은 센서 네트워크 노드를 디자인할 때 많은 제약 조건들과 요구 조건들을 고려해야 한다. 먼저 무선 센서 네트워크의 각 센서 노드에는 극도로 제한된 하드웨어 자원 조건 하에서도 무선 통신 가능한 마지막까지 동작할 수 있는 기능이 포함되어야 한다. 또한 환경과 응용 프로그램의 변화에 잘 대처하기 위해 한타임(run-time)에 각 센서 노드들로 동적으로 재구성할 수 있는 기능이 제공되어야 한다. 이러한 디자인 요구 조건들과 제약 조건들은 앞서 서로 상관없는 것처럼 보이더라도, 무선 센서 노드들을 위한 실행 환경을 다자인할 때는 이러한 조건들을 모두 만족할 수 있는 운영 체제가 반드시 필요하다. 본 논문에서 우리는 무선 센서 노드들을 위한 매우 효율적이고 효과적인 유한 상태 머신(finite state machine) 기반의 운영 체제, SenOS를 제안한다. 또한 새로운 운영 체제인 SenOS가 국도의 제한에서도 동작성과 반응성, 재구성성의 요구 조건들을 모두 만족시키면서 동작할 수 있다는 것을 보인 것이다.

1. 서론
무선 센서 네트워크는 유비쿼터스 컴퓨팅에서 하나의 핵심 기술로 따로고 있는 분야이다. 작은 범위에서 동작하는 RF 통신 매체를 통해 연결된 무선 장치는 유비쿼터스 컴퓨팅 환경에서 사람과 컴퓨팅 사이에 중개자로서의 역할을 수행한다. 기존의 컴퓨팅 플랫폼은 대용량 데이터와Ⰾ라 무선 센서 노드들은 다음과 같은 세 가지 특징을 가진다. (1) 컴퓨팅 환경의 메모리, 배터리 등의 모든 자원이 극도로 제한적이다. (2) 센서 네트워크를 수 만대의 작은 자료화된 노드들로 구성된 분산 컴퓨팅 플랫폼으로 보는 데이터 중심형(data-centric) 프로그래밍 스타일이 일반적이다. (3) 재사용을 고려하지 않는 일회용 컴퓨팅 플랫폼이다. 네트워크 센서 노드들의 이러한 특징 때문에 국도의 자원화 경영 장치에서 동작할 수 있으며 동시에 환경과 응용 프로그램의 변화에 대처하기 위해 동적인 재구성을 지원할 수 있는 특특한 구조의 운영 체제가 필요하다. 또한 이런 운영 체제는 동시에 여러 개의 빌드기반의 이벤트를 대응할 수 있어야 하며 이들 빌드용 통신 범위 중심형 프로그래밍 모델을 지원해야 한다. 사실 이렇게 상상된 것처럼 보이는 요구 조건들을 모두 만족시키는 운영 체제는 매우 어려운 일이다.

하지만 여타 센서 네트워크의 응용 프로그램들은 안전한 일련의 동작들을 수행하거나 프로그램의 모드에 따라 입력 이벤트를 다르게 처리하는 경향이 있기 때문에 상태 머신(State Machine) 기반의 컴퓨터 모델에 적합하다. 상태 머신 기반 컴퓨터 모델은 다음과 같은 장점을 가지고 있다: (1) 프로그래밍이 쉽게 디자인 모델을 개발할 수 있으며 일반적으로 널리 쓰이는 코드 생성 템플릿을 이용하여 디자인 모델을 자동으로 실행 코드로 변환할 수 있다. (2) 동시에 여러 입력 이벤트들을 다룰 수 있으며 반응이 매우 빠르다. (3) 프로그래밍이 결정되는 과정에서 저장되고 복구되는 상태 정보가
2. 유한 상태 머신의 프로그램 실행 환경
유한 상태 머신은 불연속적인 입력과 출력을 가진 시스템의 수학적 모델이다. 상태 머신은 유한 개의 내부 상태를 중 하나의 상태를 가질 수 있으며 이는 초기 상태로부터 입력된 이벤트들에 의해 결정된다. 보다 자세히 살펴보면 유한 상태 머신은 (1) 유한 개의 상태 집합, (2) 유한 개의 입력 집합, (3) 유한 개의 출력 집합, (4) 현재 상태와 입력으로부터 다음 상태를 구해내는 상태 전이 함수, (5) 현재 상태와 입력으로부터 출력을 구해내는 출력 함수, (6) 초기 상태로 구성된다. 이러한 구성은 출력이 단지 현재 상태의 함수가 되는 모레 마이어(Moore machine)이 아닌, 출력이 현재 상태와 입력의 함수가 되는 미레 마이어(Mealy machine)의 경우를 묘사한 것이다.

유한 상태 머신에서 유한한 입력 이벤트는 머신을 현재 상태에서 다음 상태로 옮기는 상태 전이와 출력을 발생시킨다. 이러한 상태 전이는 즉각적으로 일어나며 전이된 상태에서 입력 출력이 발생한다. 이를 실행 액세스를 이용하여, 유한 상태 머신은 연속된 입력의 동작들을 수행하거나 머신의 상태에 따라 입력 이벤트를 다르게 처리한다.

유한 상태 머신을 구현하기 위해서는 다음과 같은 네 가지 구성 요소가 필요하다. (1) 이벤트 큐로부터 입력을 받아들이는 상태 템플릿(state sequence)과 (2) FIFO 방식으로 동작하여 입력을 저장하는 이벤트 큐, (3) 출력 함수들을 담고 있는 콜백(callback) 함수 라이브러리, (4) 각 유한한 상태 전이와 연결된 콜백 함수들을 정의하고 있는 상태 전이 테이블이다. 각 콜백 함수는 순간적인 상태 전이 체제를 유지하기 위해 “run-to-completion” 조건을 만족시켜야 한다.

3. 상태 기반 운영 체제 구조
Senos 커널의 구조는 유한 상태 머신 모델에 기반을 두고 있다. Senos 커널은 다음과 같이 세 가지 부분으로 구성된다. (1) 상태 전이기와 이벤트 큐로 구성된 핵심 커널부, (2) 상태 전이 이벤트, (3) 콜백 라이브러리. 다음 그림 1은 Senos 커널의 구성을 잘 보여주고 있다.

![그림 1. 상태 기반 운영 체제 구조](https://example.com/series/350.png)
항수가 선정되지 않고 실행을 마칠 수 있도록 유닉스(mutex)로 항수를 보호해야 한다.

클랙 라이브러리는 머리 편의로 생성된 항수들의 집합으로, 응용 프로그램의 센서 노드의 기능을 결정할 때 사용한다. 상태 전이 테이블의 엔티티에 리드택이나 수정될 수 있는 빈번에 커널과 클랙 항수 라이브러리는 정적으로 필요되어 센서 노드의 클래식 likewise에 저장된다. SenOS는 여러 개의 상태 전이 테이블을 동시에 유지함으로써 여러 개의 응용 프로그램을 지원할 수 있으며 상태 전이 테이블을 교체하는 방법으로 여러 개의 응용 프로그램을 동시에 수행 시킬 수 있다. 이때 하나의 상태 전이 테이블은 한 개의 응용 프로그램을 의미한다. 응용 프로그램이 실행될 때 커널은 실행되는 응용 프로그램의 현재 상태 정보를 저장하고 전환하는 응용 프로그램의 상태 정보를 복구시킨다. 그리고 상태 전이 테이블을 변경함으로써 프로그램 교체를 완료한다.

SenOS에는 또한 동적으로 응용 프로그램을 로딩하는 런타임 모니터(monitor)가 포함되어 있다. 롱 1에서도 볼 수 있지만, 사실상 모든 외부 인터럽트에 대해 훌륭한 일반적인 인터럽트 서비스 선태리한 인터럽트 필터가 SenOS의 런타임 모니터에 내장되어 있다.SenOS가 인터럽트를 통해 런타임로부터 응용 프로그램 리코드 메시지를 받았을 때, 모니터는 커널이 안전한 상태에 두고 가능한 한 새로운 상태 전이 테이블을 리로드한다. 이 경우 모니타는 커널이 상태 전이 상태에 있지 않는 한 언제라도 커널은 인터럽트 시 BCHP 수 있다. 상태 전이가 유닉스에도 보호되기 때문에 유한 상태 머신은 이러한 인터럽트에도 불구하고 안전성을 유지할 수 있다.

이상은 국내 우리는 SenOS가 동작성, 반응성, 런타임 재구성성 등의 요구조건을 충족하며 안전사실 수 있다는 것을 알 수 있다.

4. 결론


참고문헌