모바일 앱 프로파일 기반 센서 레지스트리 시스템의 구현 및 평가

최호진*, 유현석*, 정동원*††, 전근환**
*군산대학교 통계컴퓨터과학과
**군장대학교 작업치료과
e-mail:{shfwkd234, hyunseok88, djeong}@kunsan.ac.kr, khjeon@kunjang.ac.kr

Implementation and Evaluation of the Sensor Registry System based on Mobile App Profiles

Hojin Choi*, Hyunseok Yoo*, Dongwon Jeong*, Keunhwan Jeon**
*Dept. of Statistics and Computer Science, Kunsan National University
**Dept. Occupational Therapy, Kunjang University

요 약

이 논문에서는 기존 센서 레지스트리 시스템의 처리 성능을 개선하기 위해 제안된 모바일 앱 프로파일 기반 센서 필터링 기법을 실험하고 평가한다. 이기종 센서 네트워크 환경에서 센서 레지스트리 시스템은 센서 메타데이터를 제공함으로써 센서 정보의 의미 처리를 즉시적으로 가능하게 한다. 하지만 불필요한 센서 메타데이터까지 처리하여 전체적인 처리 속도를 저하시킨다. 이 문제점을 개선하기 위해 모바일 앱 프로파일 기반 센서 필터링 기법을 제안한다. 모바일 앱 프로파일을 활용하여 센서 데이터의 유효성을 식별 후 모바일 기기에 센서 메타데이터를 기록한다. 기록된 정보를 재사용하여 불필요한 센서 데이터를 필터링 한다. 불필요한 센서 데이터를 처리 하지 않고, 센서 메타 데이터의 요청 횟수를 줄여 전체적인 센서 데이터 처리 속도를 향상 시킨다. 기존 방법과 제안 방법을 구현 하고 실험하여 제안 방법의 전체적인 처리 속도가 향상됨을 확인한다.

1. 서론

최근 사물인터넷(IoT, Internet of Things)과 관련된 기술이 확산되면서 실생활에서 유용하게 사용할 수 있는 다양한 서비스들이 개발되고 있다. 예를 들어 스마트홈 IoT, IoT 보일러 등 사물인터넷 기술이 접목되어 실제 서비스 되고 있는 제품들은 사용자에게 편의성을 제공하고 있다[1].

이러한 센서 네트워크 환경에서 센서 데이터 처리는 매우 중요하며, 이를 위한 다양한 연구가 진행되어 왔다. 하지만 기존 연구들은 센서 데이터를 모바일 기기에서 바로 처리 하는 즉시성에 적합하지 않았다.

센서 레지스트리 시스템(SRS, Sensor Registery System)은 이기종 센서로 얻을 수 없는 다양한 센서의 정보를 등록 및 관리하여 센서 데이터의 정보, 즉 센서 메타데이터를 제공함으로써 모바일 기기에서 즉시적인 의미 처리 및 연산의 제공에 도움을 준다[2]. 하지만 불필요한 센서 데이터까지 처리하여 성능 측면에서 개선이 필요하다. 이러한 문제점을 해결하기 위해 모바일 앱 프로파일 기반 센서 필터링 기법이 제안되었다.

제안 방법은 프로파일 정보를 활용하여 센서 데이터의 유효성을 식별 후 센서 데이터를 활용한다. 센서 메타데이터와 유효속성 값을 모바일에 임시적으로 저장 후 정보를 활용하여 불필요한 센서 값을 필터링 함으로써 기존 SRS의 문제점을 개선하여 전체적인 처리 속도를 향상시킨다[3].

이 논문에서는 기존 연구의 구현 및 실험평가가 이루어 지지 않아 기존 SRS방식과 앱 프로파일 기반 SRS방식을 시뮬레이션 실험을 통하여 정량 평가한다. 평가 방법은 센서 처리 속도에 초점을 두어 평가한다.

이 논문의 구성은 다음과 같다. 제2장에서는 기존 연구에 대해 소개한다. 제3장에서는 시스템 구조와 유효성 식별 상세 알고리즘을 서술한다. 제4장에서는 실험과 평가 결과를 설명한다. 마지막으로, 제5장에서는 결론 및 향후 연구에 대해 기술한다.

2. 기존 연구

이기종 센서 네트워크 환경에서 원활한 센서 데이터 처리를 위해 다양한 연구가 진행되어 왔다. 기존 제안 방법들은 모바일 환경에서 센서 정보를 바로 처리하는 즉시성에 적합하지 않았다. SRS는 센서의 정보를 등록하고 관리하여 센서 메타데이터를 제공함으로써 모바일 기기에서 즉시적으로 센서 데이터 처리를 가능하게 만든다[2]. 하지만 센서 식별 과정에서 모바일 기기에 유효한 센서와 유효하지 않은 센서의 메타데이터를 모두 처리함으로써 처

† 이 논문은 2014년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2014R1A1A2058992)
†† 책임저자 : 군산대학교 정동원

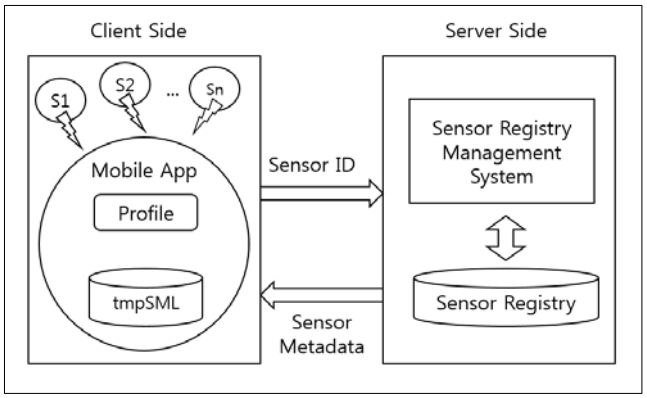
리 속도를 저하시킨다[3]. 이러한 문제점을 해결하기 위해 불필요한 센서를 필터링하고 유효 센서 데이터를 처리할 수 있는 기법인 앱 프로파일 기반 SRS가 제안 되었다.

앱 프로파일 기반 SRS는 모바일 기기에서 센서 데이터를 수신한 후, 수신된 센서 데이터의 유효성을 판단하기 위해 자신이 소유하고 있는 프로파일 정보를 이용한다. 프로파일 정보는 자신에게 유효한 센서 정보를 담고 있다[3]. 센서 메타데이터 즉, 현재 식별된 센서의 정보와 함께 프로파일을 활용하여 유효성을 판단 후, 유효속성 값을 임시적으로 모바일 기기에 저장한다. 그 후 센서 데이터가 모바일 기기에 수신될 경우 최근 식별된 센서 정보를 활용하여 유효성을 검사한다. 앱 프로파일 기반 SRS를 활용하여 센서 데이터를 처리할 경우, 초기 유효성 검사와 임시적인 센서 데이터 와 유효속성 값 저장을 위하여 센서 데이터 처리 시간이 더 소모된다. 하지만 같은 센서 데이터를 다시 수신 받을 경우 메타데이터를 요청하지 않고, 불필요한 센서의 정보를 필터링하여 전체적인 처리 속도가 향상된다.

3. 시스템 구조 및 유효성 식별 알고리즘

3.1 시스템 구조

그림 1은 앱 프로파일 기반 SRS 시스템 구조를 나타내고 있다. 서버영역(Server Side)은 센서의 메타데이터를 저장하고 관리하는 기존 SRS의 구조와 동일하다. 사용자 영역(Client Side)은 유효성을 식별하기 위한 프로파일 정보와 최근 식별된 센서 리스트(tmpSML)를 포함한다. tmpSML은 서버에서 수신 받은 센서 메타데이터와 유효속성을 저장한다. 프로파일 정보와 tmpSML은 다양한 형대로 저장이 가능하다. 이 논문에서 프로파일 정보는 리스트로 구성 하였고, tmpSML은 SQLite를 이용하여 구성 하였다.

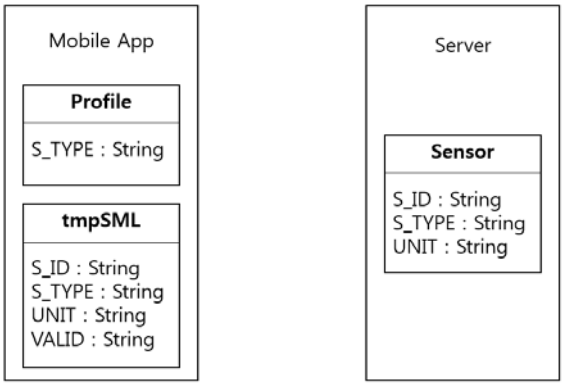


(그림 1) 시스템 구조

그림 2는 프로토타입을 구현하기 위한 클래스 속성을 표현한다. 우선 모바일 앱의 프로파일 정보는 해당 앱의 유효한 센서의 타입(S_TYPE)의 속성을 가지고 있다. 센서 타입은 해당 앱에 따라 여러 값을 포함 할 수 있다. tmpSML 클래스는 서버영역에서 전달받은 센서 메타데이터 정보와 유효성을 식별하기 위한 유효속성(VALID)을 포함한다. 유효속성의 값(유효함=1, 유효하지 않음=0)을 이용하여 필터링에 사용한다. 서버는 센서의 다양한 메타데이터를 제공한다. 센서 클래스는 센서 아이디(S_ID), 센서 타입(S_TYPE), 센서 단위(UNIT)을 각각 속성으로 정의한다.

3.2 유효성 식별 알고리즘 재정의

기존 연구의 유효 센서 식별을 위한 상세 알고리즘을 그림 3과 같이 재정의 하였다. 모바일 기기에서 센서 데이터를 수신 받은 경우 tmpSML에 정보가 있는지 확인한다.



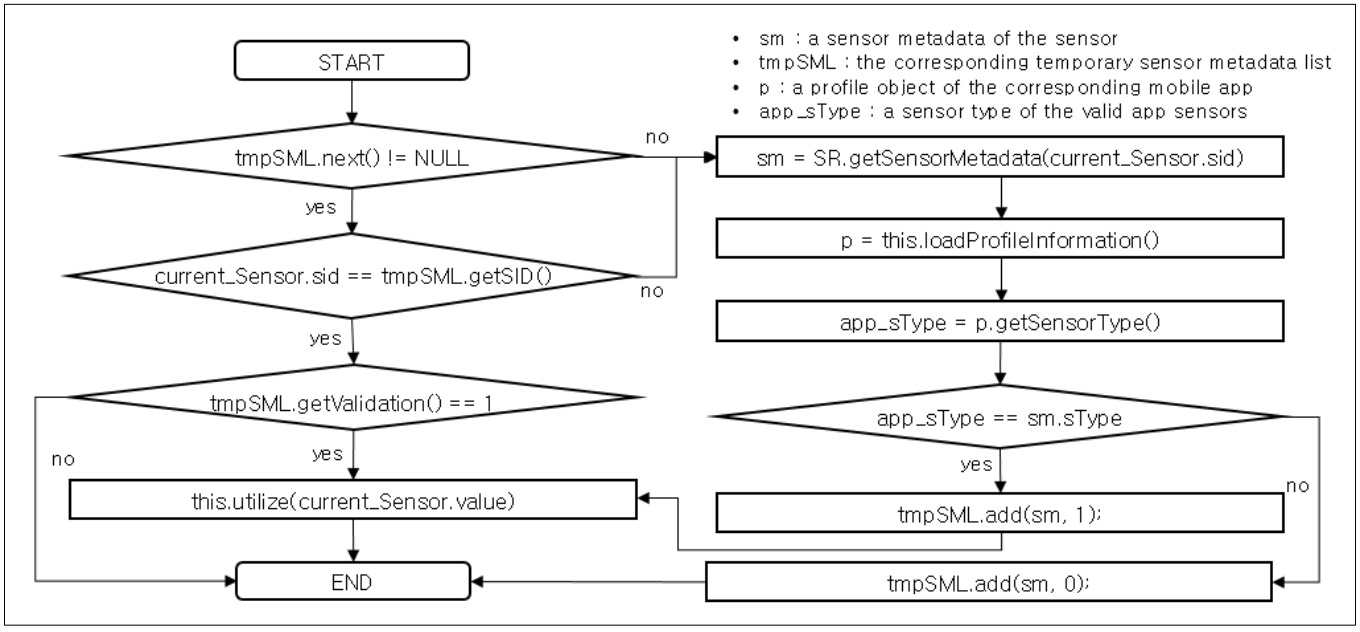
(그림 2) 프로토타입 구현을 위한 클래스 속성

정보가 있을 경우, 현재 센서 데이터와 비교 하고, 유효한 센서 값인지 확인한다. 유효한 값일 경우 센서 값를 사용한다. 유효하지 않은 값일 경우 센서 값을 사용하지 않는다. tmpSML에 정보가 없거나, 현재 센서 데이터의 정보가 없을 경우 센서 레지스트리 시스템을 통하여 현재 센서 정보의 메타데이터를 요청한다. 요청 후 받은 현재 센서의 메타데이터를 확인 후, 프로파일을 사용하여 유효한 센서 인지 확인한다. 유효한 센서 일 경우 최근 식별된 센서 리스트에 메타데이터와 함께 유효속성 값(VALID=1)을 저장한 후 센서 데이터를 사용한다. 유효한 값이 아닐 경우에도 유효하지 않은 속성 값(VALID=0)을 저장한다. 유효하지 않은 정보까지 저장하는 이유는 다시 유효하지 않은 센서 정보를 모바일 기기에서 수신 받을 경우 센서 메타데이터를 요청 하지 않고 모바일 기기에서 즉시 필터링하기 위함이다.

4. 실험 및 평가 결과

이 논문의 물리적인 구현 환경을 요약하면 다음과 같다.

- 스마트폰 운영체제 : Andrioid 4.1.2 (Jelly Bean)
- 앱 프로파일을 활용한 필터링 저장소를 위한 DBMS : Android SQLite
- 서버 시스템 운영체제 : Microsoft Window 7 Professional K 64bit
- 서버 시스템 사양 : Intel(R) Core(TM) i3 CPU 530 @2.93GHz
- 서버 레지스트리 저장소를 위한 RDBMS : Oracle Data Base 11g Enterprise Edition



(그림 3) 유효성 식별을 위한 재정의 알고리즘

이 논문에서 실제 활용하기 위해 Sensor 클래스의 데이터 Set을 그림 4와 같이 구성하였다.

	S_ID	S_TYPE	UNIT
1	s001	temp	celsius
2	s002	humi	percent
3	s003	illum	lx
4	s004	temp	fahrenheit

(그림 4) Sensor 클래스의 데이터 Set

(a) 기존 방식

```
use sensor: s_id: s001, s_type: temp, unit:
celsius, value: 20.7
use sensor: s_id: s001, s_type: temp, unit:
celsius, value: 21.0
use sensor: s_id: s001, s_type: temp, unit:
celsius, value: 20.6
use sensor: s_id: s001, s_type: temp, unit:
celsius, value: 20.6
use sensor: s_id: s001, s_type: temp, unit:
celsius, value: 20.7
```

(b) 제안 방식

```
insert record TmpSML: s_id: s001 valid:1
s_id: s001, s_type: temp, unit: celsius,
value: 20.3, valid: 1
s_id: s001, s_type: temp, unit: celsius,
value: 20.3, valid: 1
s_id: s001, s_type: temp, unit: celsius,
value: 20.4, valid: 1
s_id: s001, s_type: temp, unit: celsius,
value: 20.4, valid: 1
s_id: s001, s_type: temp, unit: celsius,
value: 20.9, valid: 1
```

(그림 5) 모바일 구현 화면

그림 5는 실제 모바일 기기 상에서 실행되는 화면을 보여주고 있다.

그림 5(a)는 기존 방식인 센서 데이터가 모바일로 수신될 경우 센서 메타데이터를 요청한 후 센서 데이터를 사용한다. 기존의 경우 센서 데이터가 들어올 때마다 메타데이터를 요청한다.

그림 5(b)는 제안 방식인 앱 프로파일을 활용하여 초기 센서 데이터가 들어올 경우 센서 메타데이터를 요청한 후 앱 프로파일을 활용하여 모바일 기기 상의 tmpSML 클래스에 임시적으로 메타데이터와 함께 유효속성 값을 저장한다. 저장한 후 자체적으로 유효속성 값을 구별하며 모바일 기기에서 즉시적으로 센서 데이터를 처리한다.

그림 6은 기존 방식의 모바일 기기 동작을 나타내고 있다. 모바일 기기는 센서로 전달받은 센서 데이터(센서 아이디 = S_id)를 서버로 보낸 후 센서 메타데이터를 수신 받고 분석하여 센서 데이터를 활용한다. 센서 데이터를 수신 받을 때마다 이 동작을 반복한다.

그림 7은 제안 방식의 모바일 기기의 동작을 나타내고 있다. 그림 7(a)는 초기 프로파일을 이용한 필터링과 최근 식별 센서 리스트의 기록을 나타낸다. 센서 필터링과 메타데이터 저장을 위해 기존 보다 더 많은 처리를 수행한다. 하지만 그림 7(b)와 같이 한번 기록이 된 센서 정보를 다시 재활용 하여 메타 데이터를 요청 하지 않고, 센서 데이터를 처리 한다.

이 실험에 활용된 앱 프로파일 정보는 온도(temp)와 습도(humi)센서 타입을 갖고 있다. s001, s002, s004를 수신 받게 되면 최근 식별된 센서 리스트에 유효한 속성 값(VALID=1)을 저장한다. s003을 수신 받게 되면 유효하지 않은 속성 값(VALID=0)을 저장한 후, 필터링 한다.

```
I/센서(s001,21.0)값: 센서 : output
I/센서(s001,21.0)값: 모바일 : input
I/서버로 S_id 데이터 전송: 모바일 -> 서버
I/서버에서 S_id 관련 SR 데이터 전송: 서버 -> 모바일
I/서버에서 받은 전체 내용 :: {"List":[{"s_id":"s001","unit":"celsius","s_type":"temp"}]}
I/JSON을 분석한 데이터 :: s001 temp celsius
I/SR 데이터: 바로 활용
I/센서(s001,20.4)값: 센서 : output
I/센서(s001,20.4)값: 모바일 : input
I/서버로 S_id 데이터 전송: 모바일 -> 서버
I/서버에서 S_id 관련 SR 데이터 전송: 서버 -> 모바일
I/서버에서 받은 전체 내용 :: {"List":[{"s_id":"s001","unit":"celsius","s_type":"temp"}]}
I/JSON을 분석한 데이터 :: s001 temp celsius
I/SR 데이터: 바로 활용
```

(그림 6) 기존 방식의 모바일 기기 처리 Log 기록

```
I/센서(s001,20.9)값: 센서 : output
I/센서(s001,20.9)값: 모바일 : input
I/TmpSML에 데이터가 없는 경우: 0개의 데이터가 있습니다.
I/서버로 S_id 데이터 전송: 모바일 -> 서버
I/서버에서 S_id 관련 SR 데이터 전송: 서버 -> 모바일
I/서버에서 받은 전체 내용 :: {"List":[{"s_id":"s001","unit":"celsius","s_type":"temp"}]}
I/JSON을 분석한 데이터 :: s001 temp celsius
I/Profile과 일치하는 타입: temp :TmpSML에 저장 = valid :1
```

(a) 초기 tmpSML 기록

```
I/센서(s001,20.1)값: 센서 : output
I/센서(s001,20.1)값: 모바일 : input
I/TempSML에 데이터가 있는경우 :: 1개의 데이터가 있습니다.
I/유용한 값: 센서 정보 활용: value=1
```

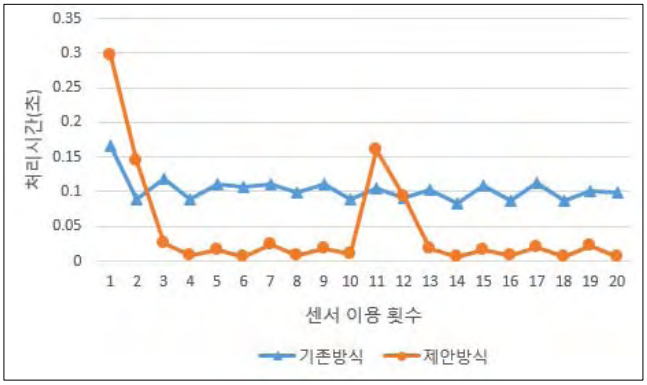
(b) tmpSML 정보 재사용

(그림 7) 제안 방식의 모바일 기기 처리 Log기록

그림 8은 기존 방식과 제안 방식의 센서 데이터 처리 속도를 그래프로 표현한다. 센서 이용 횟수는 시뮬레이션으로 생성한 센서 s001, s002, s003, s004를 각각 5번 활용한다. 총 10번 실험을 실행하여 평균 처리 시간을 기록한다. 센서 이용 횟수가 1, 2회 일 때 s001과 s002를 인식하여 총 10번의 데이터가 수신되고, 11, 12회 일 때 s003과 s004가 마찬가지로 10번의 센서 데이터가 수신된다. 1회 실험 당 데이터는 20회의 처리 시간 데이터가 생성된다. 총 10회 실험하여 총 200횟수의 시간 데이터가 생성된다. 센서 데이터 처리 시간은 수치상으로 기존 방식이 평균 0.1초, 제안 방식은 0.04초가 소요된다. 초기 센서 데이터 수신시 제안 방식이 기존 방식보다 처리 시간이 더 소요되었지만, 그 이후에는 기존 방식보다 낮은 처리 시간을 확인할 수 있다. 전체적으로 보았을 때에는 제안 방식이 데이터 처리 속도 측면에서 기존 방식보다 향상됨을 알 수 있다.

6. 결론 및 향후 연구

이 논문에서는 기존 앱 프로파일 기반 센서 필터링 기법을 모바일 기기에서 실험하고 평가하였다. 기존 연구는 SRS 이용의 단점을 보완하기 위해 앱 프로파일 기반 센서 필터링 기법을 제안하였다. 하지만 구현 및 실험이 이루어지지 않아 제안방법의 우수성이 잘 표현되지 않았다. 이 논문에서는 구현 및 실험에 앞서 기존 유효성 식별을 위한 알고리즘을 재정의 하였다. 앱 프로파일 기반의 SRS는 센서 유효성 식별을 위해 센서 메타데이터의 기록과 유효속성



(그림 8) 기존방식과 제안방식의 처리 속도

값을 처리해야 하므로 초기처리 시간과 모바일 기기 내부의 저장 공간을 소모한다. 하지만 그 이후 같은 센서 데이터를 수신 받기 되면 센서 메타데이터를 서버에 호출하지 않고, 모바일 기기 자체에서 메타데이터 정보를 처리하고 불필요한 센서를 모바일에서 필터링하여 전체적인 처리 속도가 향상되었다. 향후에는 실제 센서 환경을 바탕으로 한 구현 및 실험 및 평가 연구가 요구된다.

참고문헌

[1] W. Jeong, S. Kim, K. Min, "An Analysis of the Economic Effects for the IoT Industry," Journal of Korean Society for Internet Information 2013. Oct: 14(5): 119-128

[2] D. Jeong and J. Ji, "A Registration and Management System for Consistently Interpreting Semantics of Sensor Information in Heterogeneous Sensor Network Environments," journal of KIISE : Databases, Vol. 38, No. 5, pp. 289-302

[3] D. Jeong, H. Yoo, S. Lee, "Sensor Filtering based on Mobile App Profiles for Enhancing the Processing Performance of Sensor Registry System," Korea Information Processing Society (KIPS), Proceedings of the 2015 Spring Conference of the KIPS, Vol. 22, No. 1, pp. 273-276,