KIPS Tr. Comp. and Comm. Sys.
Vol.5, No.2 pp.27~32 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2016.5.2.27

A Sensor Identification Scheme for Dynamic Interworking Between Personal Sensor Devices and a Smartphone

Hong Min†

ABSTRACT

Several sensor devices have been developed for monitoring individual's health and status information and services which visualize customized information by associating applications running on smartphones with sensor devices also have been emerged. Though these applications provide similar information to a user, each sensor device has its own application caused by non-standardized packet formats. In this paper, we propose a sensor device identification for dynamic interworking between a smartphone and personal sensor devices. In the proposed scheme, we can use the same application which plays role of a client on the smartphone as changing sensor devices because server stores packet information of sensor devices.

Keywords : Personal Sensor Devices, Smartphone, Identification Scheme, Dynamic Interworking

개인용 센서 기기와 스마트폰의 동적 연동을 위한 센서 식별 기법

민 홍†

요 약

개인의 건강 및 상태 정보를 모니터링하기 위한 다양한 센서 기기들이 개발되고 있으며, 스마트폰 상에서 동작하는 응용과 연동을 통해 개인화된 정보를 시각화하여 보여주는 서비스들이 등장하고 있다. 이러한 응용들은 유사한 정보를 사용자에게 제공하면서도 표준화되지 않은 패킷 구조로 인해 센서 기기마다 별도의 응용을 설치해야 하는 문제가 있다. 본 논문에서는 비표준화된 개인용 센서 기기들이 스마트폰과 동적으로 연동될 수 있는 센서 기기 식별 기법을 제안한다. 제안 기법은 센서 기기에 대한 패킷 정보를 서버에 저장하기 때문에 사용하는 센서가 변동되더라도 스마트폰 상에서 동작하는 클라이언트의 변경 없이 응용을 동작할 수 있는 장점이 있다.

키워드 : 개인용 센서 기기, 스마트폰, 식별 기법, 동적 연동

1. 서 론

스마트폰은 통신기기로써의 역할과 함께 기기내의 다양한 종류의 센서들을 활용하여 개인 및 사회 조직의 활동을 모니터링 하는 기기로써의 역할도 수행할 수 있다[1]. 일례로 스마트폰에 탑재된 마이크와 가속도 센서의 패턴 정보를 저장하고 이를 통해 웹 페이지에 접속한 사용자를 식별할 수 있다[2]. 최근에는 스마트폰과 주변의 센서 및 사용자 몸에 부착할 수 있는 시계, 이어폰, 밴드와 같은 웨어러블 장비들과의 연동을 통해 사용자의 활동과 건강 상태 등을 관리 하는 유헬스 응용들도 많이 연구되고 있다[3]. 인구 구조가 고령화 되면서 지속적인 관리가 필요한 만성 질환들에 대한 관리의 필요성이 높아지면서 의료 시스템의 패러다임이 질병 치료에서 관리로 변화되고 있다. 이를 위해 예외 관리적 관점에서 재택환자의 이상의 건강상태를 실시간적으로 모니터링하고 스마트폰을 통해 자기관리 및 적절한 건강 개입이 가능하도록 하는 시스템도 개발되었다[4]. 스마트폰이 데이터를 전달하고 결과를 보여주는 단순한 용도가 아닌 컴퓨팅 능력을 활용하여 데이터를 가공하고 이를 처리하여 고급화

† 종신회원 : 호서대학교 컴퓨터정보공학부 조교수
Manuscript Received : September 14, 2015
First Revision : December 19, 2015
Second Revision : January 28, 2016
Accepted : January 28, 2016
* Corresponding Author : Hong Min(hmin@hoseo.edu)

된 정보를 사용자에게 제공할 수 있다. 또한 사물 인터넷 환경하에서 센서 기기들이 비동기적으로 데이터를 제공하고 이를 활용하여 서비스를 제공하는 응용을 지원하기 위한 공통의 플랫폼도 제안 되었다[5]. 이러한 다양한 웨어러블 디바이스와 응용 지원을 위한 플랫폼 개발은 개인 건강 관리 응용의 확대와 센싱 데이터의 정확도를 높이는 장점이 있지만 각 제조사에 특화된 통신 인터페이스 및 프로토콜, 패킷 형식을 사용함으로써 상호 호환성이 보장되지 못하는 단점이 있다.

본 논문에서는 표준화 되지 않은 센서 기기들과 안드로이드 기반 스마트폰의 연동을 위해서 클라이언트/서버 기반 센서 식별 시스템을 설계하고 이를 개발 보드 상에서 동작할 수 있도록 구현 하였다. 제안된 시스템은 센서 장치마다 스마트폰 상에서 연동하는 응용을 별도로 작성해야 하는 문제를 해결하기 위해서 센서 장치를 식별하고 표준화된 패킷으로 변환하기 위한 정보 제공 기능을 서버 상에 구현하였다. 스마트폰 상에 존재하는 클라이언트는 Wi-Fi, 블루투스, USB 등과 같은 다양한 인터페이스들로부터 전달되는 패킷을 모니터링하고 이를 서버에 전달한다. 전달된 패킷은 서버에서 패킷 식별 기법을 통해 연결된 센서의 종류를 확인하고 패킷 표준화에 필요한 정보를 클라이언트에게 전달한다. 이러한 과정을 통해서 응용은 표준화된 API(Application Programming Interface)와 패킷을 제공 받기 때문에 어떤 종류의 센서와도 동적으로 연동할 수 있다.

2. 클라이언트/서버 기반 센서 식별 기법

센서 장치에서 생성되는 패킷은 제조사가 같아도 Fig. 1에서 보는 것과 같이 서로 다른 길이와 구조의 바이트 스트림을 보인다. 따라서 센서를 활용하는 응용을 제작하기 위해서는 센서의 패킷 구조를 먼저 이해해야 하고, 다른 센서를 사용하기 위해서는 패킷을 해석하는 코드를 직접 수정해야 한다. 또한 패킷의 구조가 다른 센서 기기가 추가 될 때마다 패킷 정보를 저장하는 데이터베이스를 갱신해야 하는 문제도 발생한다. 제안 기법에서는 이러한 문제를 해결하기 위해 다수의 센서 패킷과 구조적 특징을 서버에 저장하고 클라이언트로부터 요청이 있을 경우 수집된 바이트 스트림을 전달받아 이를 비교해서 센서의 종류를 식별한다. 센서 식별이 완료되면 서버는 클라이언트에게 센서 종류, 메시지 길이, 통신 패킷에서 센싱된 데이터가 저장된 위치 등의 정보를 전송함으로써 스마트폰에서 동작하는 응용이 패킷에 대한 해석 없이 센싱 데이터를 추출할 수 있도록 해준다.

제안된 기법은 Fig. 2와 같은 구조로 동작한다. 클라이언트는 기존의 안드로이드 플랫폼 상에서 동작하기 위해 네이티브 라이브러리의 형태로 설계했으며 센서 장치의 연결 여부를 모니터링하고 새로운 장치 연결 시 일정 크기의 바이트 스트림을 샘플링하여 서버에 전송한다. 서버는 클라이언트로부터 전송된 바이트 스트림과 저장된 센서 식별 사전의 데이터를 비교하여 일치하는 데이터가 있는 경우 해당 센서의 식별 값과 패킷으로부터 실제 데이터를 추출하기 위한 정보를 클라이언트에게 전달한다. 클라이언트는 서버로부터 반환 받은 분석 결과를 바탕으로 장치의 식별정보와 센서로부터 추출한 데이터, 센싱 시간 등의 정보를 포함하는 표준화된 패킷을 생성한다. 이렇게 표준화된 패킷은 네이티브 API를 통해 안드로이드 응용에게 전달된다.

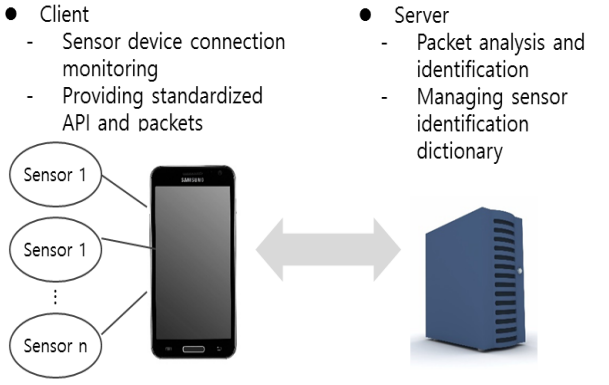

Fig. 2. The Proposed System Overview

2.1 클라이언트 동작 과정

센서 식별을 위한 클라이언트는 Fig. 3과 같은 구조로 설계되었다. 안드로이드 응용은 센서 식별 모듈의 클라이언트가 제공하는 네이티브 API를 통해 패킷의 분석을 요청하거나 표준화된 패킷을 전달 받을 수 있다. 장치 모니터링 모듈은 다양한 인터페이스를 가지고 통신하는 센서 장치들과 스마트폰 사이에 새로운 연결이 발생하는지를 주기적으로 확인한다. USB와 같은 문자 디바이스의 경우는 /dev 디렉터리의 디바이스 파일 목록을 주기적으로 검색하고 블루투스와 Wi-Fi 같은 네트워크 장치들은 응용으로부터 주기적으로 일정 길이의 바이트 스트림을 전달받는다.

0	1	2	3	4	5	6	7	8
Start		ID		Data		CSum	END	
0x76	0x00	0x13	0x02	CO2_H	CO2_L	CSum	0xF0	0x0F

0	1	2	3	4	5	6	7	8	9	10
Start		ID		Data				CSum	END	
0x76	0x00	0x15	0x0C	Obj_H	Obj_L	Amb_L	Amb_L	CSum	0xF0	0x0F

Fig. 1. The Packet Structure of Each Sensor Device (Up: CO2, Down: Temperature)

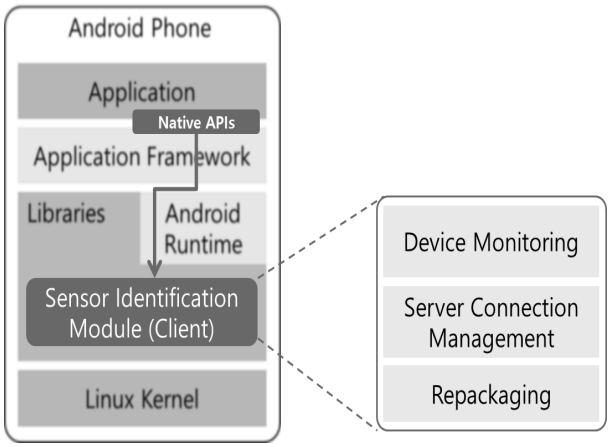

Fig. 3. The Proposed System Overview

새로운 장치의 연결이 확인되거나 응용으로부터 바이트 스트림에 대한 분석 요청이 있을 경우 서버 연결 관리 모듈은 소켓 통신을 통해 서버와의 통신할 준비를 하고 연결이 성공하면 센서로부터 수집된 바이트 스트림을 서버로 전달하고 분석 결과를 기다린다. 서버 연결 관리 모듈은 스레드(pthread) 기반으로 구현하여 동시에 여러 센서에 대한 요청이 이루어질 수 있도록 하였다.

리패키징 모듈은 서버로부터 분석된 결과를 바탕으로 바이트 스트림 중에서 유용한 센싱 정보를 추출하고 ISO/IEEE 11073 표준에 맞게 패키징하여 안드로이드 응용에 전달한다. ISO/IEEE 11073 표준은 개인용 건강 관리 장치 및 센서와 센싱된 데이터를 활용하는 관리 시스템 사이의 통신 프로토콜과 메시지 형식을 정의하고 있으며 의료 정보 분야의 표준으로 활용되고 있다[6].

2.2 서버 동작 과정

센서 식별을 위한 서버의 핵심적인 기능은 센서 패킷의 특징 정보를 저장하는 식별 사전 관리와 클라이언트로부터 전달 받은 바이트 스트림과 식별 사전의 유사도를 비교하여 센서 장치를 인식해 내는 것이다. 각 센서에서 전달된 패킷마다의 특징 정보를 저장하는 사전의 생성과 관리는 관리자에 의해서 수동으로 이루어지며 이 부분에 대한 자동화하는 향후 연구를 진행할 예정이다.

센서 식별을 위한 서버는 Fig. 4와 같은 구조로 설계되었으며 클라이언트와 연동하여 동작한다. 먼저 클라이언트는 서버와의 연결을 시도하고 연결이 완료되면 센서로부터 전송 받은 원형의 패킷 스트림을 서버에 전송한다. 서버는 클라이언트로부터의 연결 시도가 있으면 클라이언트 전용 소켓을 생성하고 전달 받은 패킷 스트림을 저장한다. 저장한 패킷 스트림은 다중 수준의 처리를 통해 사전과 비교 작업이 이루어지고 센서가 식별된 경우에는 해당 디바이스의 ID와 패킷에서 센싱 데이터를 추출하기 위한 정보들을 클라이언트에게 전송한다.

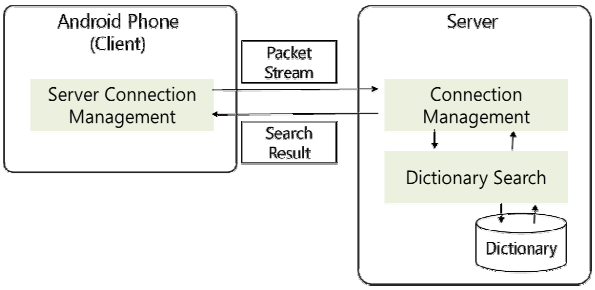

Fig. 4. Server Architecture

서버 상에서의 센서 식별을 위한 바이트 스트림과 사전과의 비교는 Fig. 5와 같은 다중 수준의 처리 과정을 통해 이루어진다. 이와 같은 계층 구조의 설계 목적은 식별 사전에 저장된 데이터량이 크기 때문에 가능한 비교 대상을 줄임으로써 검색에 필요한 시간을 최소화 하는 것이다[7].

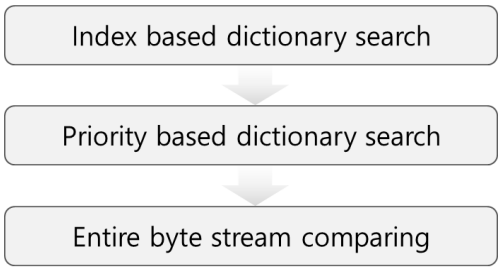

Fig. 5. Dictionary Search Steps

인덱스 기반 사전 검색에서는 클라이언트로부터 전달 받은 바이트 스트림을 전처리하여 센서 식별 사전과의 비교 속도를 높인다. 이를 위해서 바이트 스트림을 배열로 전환하여 각 바이트 값과 인덱스를 사상 정보를 저장하는 2차원 배열을 Fig. 6과 같이 생성한다. 각 바이트 스트림의 길이에 따라 1에서 k 바이트까지의 인덱스 테이블을 구성하여 시작 패턴 길이에 따라 다른 인덱스 테이블을 사용함으로써 검색 속도를 향상 시킨다[8]. 센서 식별 사전의 시작 패턴과 길이 정보

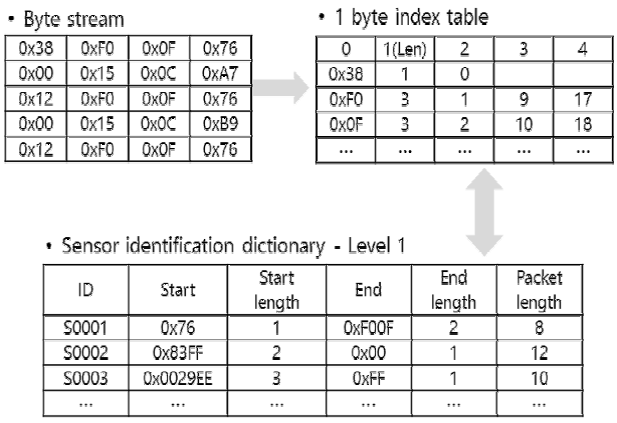

Fig. 6. Index based Dictionary Search

를 바탕으로 사용할 k 바이트 인덱스 테이블을 결정하고 시작 패턴이 일치하는 항목의 인덱스를 참조하여 종료 패턴의 일치 여부를 확인한다. 시작과 종료 패턴이 일치할 경우 바이트 스트림 내에서 다음 비교 대상 선정은 인덱스 테이블의 다음 열에 저장된 인덱스를 참조한다[9]. 바이트 스트림은 인덱스 기반의 사전 검색을 통해 일정한 크기의 패킷 단위로 정렬되고, 비교할 센서 식별 사전의 크기는 줄어든다.

1 단계에서는 클라이언트로부터 전달받은 바이트 스트림에 인덱스를 부여하고 순차적으로 1바이트씩 검색하면서 해당 위치의 값과 인덱스에 대한 사상 정보를 이차원 배열에 저장한다. 1바이트 인덱스 테이블이 생성되면 이를 참조하여k 바이트 인덱스 테이블을 생성하는데 이 과정에서는 사상된 인덱스 값은 변화되지 않고 복사되며 바이트 스트림의 길이만 늘어난다. 1바이트 인덱스 테이블을 참조하여 생성된 k바이트 인덱스 테이블의 전체 행의 길이는 k -1 만큼 줄어든다.

2단계에서는 센서 식별 사전의 시작 패턴 길이에 따라 1에서 k 바이트 인덱스 테이블 중에 해당하는 테이블을 선택하고 시작패턴의 위치를 검색한다. 검색된 시작 위치를 기준으로 패킷길이와 종료패턴을 비교하여 일치하는 경우 시작 위치부터 패킷 길이만큼 복사하여 정렬된 패킷과 센서 ID 정보를 저장한다. 다음 검색 시 새로운 시작 위치는 인덱스 테이블을 참조하여 점프함으로써 검색 시간을 줄일 수 있다.

다음으로 우선순위 기반 사전 검색을 통해 센서를 식별한다. 우선순위 기반 사전 검색에서는 Fig. 7의 센서 식별 사전(Level 2)과 같이 패킷에서 시작과 종료 패턴을 제외한 다른 데이터 중에서 각 센서를 구별할 수 있는 식별 값들의 위치를 비트 패턴(1일 경우 센서 식별에 사용, 0일 경우 무시)으로 정의하고 해당 비트가 1로 설정되어 있는 위치의 식별 값들을 순차적으로 저장한다. 이 때 식별 값 패턴에서 1의 값으로 설정된 비트가 가장 많은 위치부터 식별 값과 패킷 단위의 정렬 결과를 비교한다. Fig. 6에서는 식별 값 비트 패턴에서 3번째(패킷 인덱스는 3), 5번째(패킷 인덱스는 5), 2 번째(패킷 인덱스는 2) 순으로 값을 비교한다.

• Aligned Packet

0	1	2	3	4	5	6	7
0x76	0x00	0x15	0x0C	0xA7	0x12	0xF0	0x0F
0x76	0x00	0x15	0x0C	0xB9	0x12	0xF0	0x0F

• Sensor identification dictionary - Level 2

ID	Pattern	Pattern length	Value 1	Value 2	Value 3
S0001	01101	3	0x15	0x0C	0x12
S0382	00101	2	0x15	0x90	
S1863	00100	1	0xAA		

Fig. 7. Priority based Dictionary Search

마지막으로 인덱스와 우선순위 기반 사전 검색을 통해서 센서를 식별하지 못할 경우 패킷 단위 정렬 결과(X)와 센서 식별 사전에 저장된 패킷 샘플(Y)의 전체 문자열을 비교한다. 전체 문자열 비교는 Equation (1)과 같은 유클리디언 거리(Euclidean distance)를 측정하고 결과값이 최소인 패킷 샘플을 찾는다[10].

$$D[X,Y] = \frac{\sum_{i=1}^{n}\left(\sqrt{\sum_{j=1}^{m}(X_i[j]-Y[j])^2}\right)}{n}, \quad (1)$$

where n = # of aligned packets, m = packet length

3. 성능 평가

제안 기법의 성능 평가를 위해서 Table 1에서 보는 바와 같이 서버를 구축하고 개발 보드에 프로토타입을 구현하였다.

Fig. 8은 제안된 기법이 구현된 개발 보드상에서 동작하는 응용의 구동화면을 보여주고 있다. 응용에서는 센서의 종류와 패킷에 대한 사전 정보 없이 센서가 연결되면 동적으로 이를 인식하여 화면에 측정된 결과 값을 보여준다. 또한 동시에 여러 센서의 식별과 데이터 처리가 가능하다.

Table 1. Server and Client Specifications

Items		Specification
Server	CPU	Intel Xeon E3-1220v3 4-Core 3.10GHz
	Cache	8MB L3 Cache
	Memory	4GB
	Storage	1TB
	OS	Ubuntu Server 14.04 LTS
	DB	MySQL 5.6
Client	Model	HBE-SM7-S4412
	CPU	Samsung Exynos 4412 1.7GHz
	Memory	2GB LP-DDR2 880MHz
	GPU	ARM Mali 400MP 440MHz
	LCD	7 inch 800X1280
	Storage	eMMC 16GB
	Network	802.11b/g/n Wireless LAN
	Sensors	- Temperature: Z-TN9 - Air quality: SH-300ND, PS02C-PWM - Pulsation: SPO2, LAXTHA Ubpulse 360

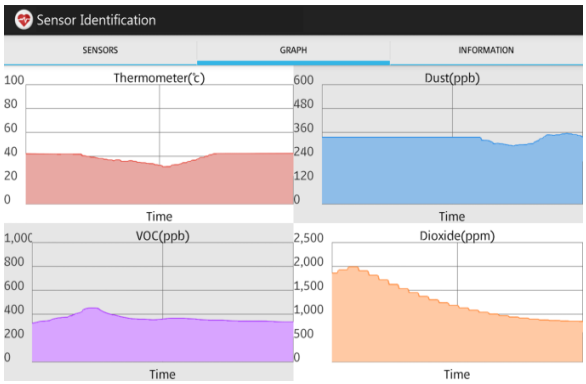

Fig. 8. The Screenshot of Demo Application

Fig. 9는 온도, 먼지, CO_2,심박 센서를 각각 안드로이드 기기와 연결하였을 때 이를 감지하고 서버에서 센서를 식별하는데 필요한 수행시간을 그래프로 표현한 것이다. 클라이언트로부터 전송받은 바이트 스트림의 크기는 100 바이트로 센서 식별 사전의 최대 크기는 500행과 1000행으로 각각 설정하였으며 문자열 전체를 비교하는 기법(Full matching)과 제안한 다중 수준의 사전 검색 기법(Multi-level matching) 사이의 식별 시간을 비교하였다.

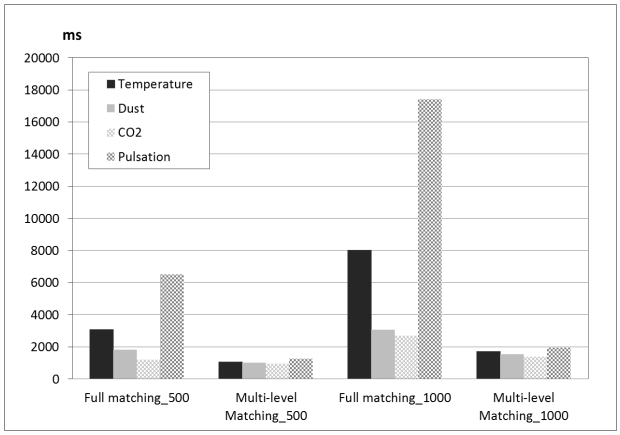

Fig. 9. Sensor Device Identification Time

기존의 문자열 전체를 비교하는 기법의 경우 데이터 패킷의 길이와 센서 식별 사전의 크기가 증가함에 따라 검색 시간이 급격하게 증가하는데 반해 제안 기법의 경우 각 검색 단계를 거치면서 Fig. 10에서와 같이 비교 대상의 크기가 줄어들기 때문에 검색 속도를 높일 수 있다.

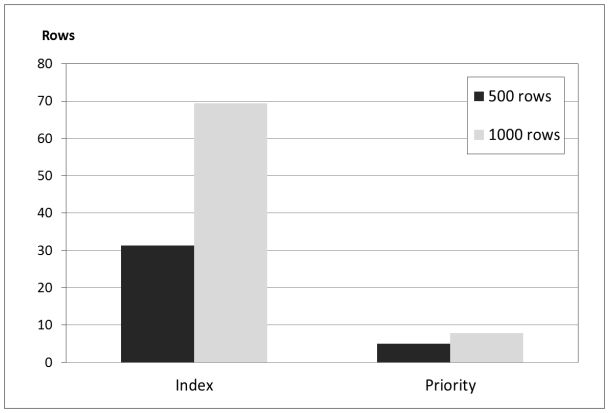

Fig. 10. The Number of Rows for Each Step

4. 결 론

최근 스마트폰은 다양한 센서들과 연동되면서 응용 영역이 점차 확대하고 있다. 기존 방식의 스마트폰과 센서와의 연동은 각 장치마다 표준화 되지 않은 패킷 형식과 통신 인터페이스 때문에 스마트폰 응용에서 동적으로 인식할 수 없는 문제가 있었다. 본 논문에서는 이러한 문제를 해결하기 위해 모든 응용에서 동일하게 센서를 접근하는 API를 제공하고 센서와의 연결을 모니터링하여 연결 시 서버에 바이트 스트림을 전달하여 식별하도록 하는 클라이언트/서버 기반 센서 식별 기법을 제안하였다. 제안 기법은 동적으로 센서를 인식할 수 있을 뿐만아니라 서버에서 센서 식별 시 다중 수준의 처리 과정을 통해 비교 대상의 크기를 효율적으로 줄여 센서 식별에 필요한 시간을 최소화 하였다. 하지만 센서 식별 사전에 정보를 등록하는 과정이 서버 관리자에 의해 수동으로 이루어지는데 이 부분은 향후 연구를 통해 자동화 할 필요가 있다.

References

[1] B. Guo, et al., “Toward a Group-Aware Smartphone Sensing System,” *IEEE Pervasive Computing*, Vol.13, No.4, pp.80-88, 2014.

[2] H. Bojinov, D. Boneh, and Y. Michalevsky, “Mobile Device Identification via Sensor Fingerprinting,” *Technical reports (Stanford University)*, pp.1-14, 2014.

[3] P. Castillejo, et al., “Integration of wearable devices in a wireless sensor network for an E-health application,” *IEEE Wireless Communications*, Vol.20, No.4, pp.38-49, 2013.

[4] N. Lee and J. Lee, “A Study on Mobile Personalized Healthcare Management System,” *The KIPS Transactions on Computer and Communication Systems*, Vol.4, No.6, pp.197-204, 2015.

[5] Y. Kang and W. Ko, “Asynchronous Sensing Data Aggregation and Processing Mechanism for Internet of Things Environment,” *The KIPS Transactions on Computer and Communication Systems*, Vol.3, No.11, pp.403-408, 2014.

[6] D. Lee, G. Bang, S. Han, and D. Choi, “A Design of U-Health System on Smart Phone Using ISO/IEEE 11073 PHD Standard,” *in Proceedings of the 2nd World Congress on Computing and Information Technology*, pp.133-138. 2014.

[7] V. Hulipalled, et al., “DRSM: Data Reduction and Similarity Matching for Time Series Data Streams,” *in Proceedings of International Conference on Advances in Communication, Network, and Computing*, pp.114-121, 2013.

[8] M. Najam, U. Younis, and R. Rasool, “Multi-byte Pattern Matching Using Stride-K DFA for High Speed Deep Packet Inspection,” *in Proceedings of IEEE 17th International Conference on Computational Science and Engineering*, pp.547-553, 2014.

[9] S. Fatehpuria, et al., “A very unique, fast and efficient approach for pattern matching (the Jumping Algorithm),” *in Proceedings of International Conference on Advanced Communication Control and Computing Technologies*, pp.1241-1245, 2014.

[10] R. H. Vishwanath, et al., “Alternate Data Clustering for Fast Pattern Matching in Stream Time Series Data,” *Advances in Communication, Network, and Computing*, Vol.108, No.1, pp.153-158, 2012.

민 홍

e-mail : hmin@hoseo.edu

2004년 한동대학교 전산과학(학사)

2011년 서울대학교 컴퓨터공학부 (석・박사통합)

2011년~2013년 Northwestern University (Postdoc)

2013년~현 재 호서대학교 컴퓨터정보공학부 조교수

관심분야 : Operating System, Embedded System, Smartphone Sensing, IoT