The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 16, No. 1, pp.147-153, Feb. 29, 2016. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2016.16.1.147
JIIBC 2016-1-20

모바일 인터넷 환경에서 CoAP 프로토콜 기반의 RD를 이용한 IoT 임베디드 노드 등록 서비스 설계 및 구현

A Design and Implementation for Registration Service of IoT Embedded Node using CoAP Protocol-based Resource Directory in Mobile Internet Environments

항뢰*, 김문권*, 김도현**

Lei Hang, Wenquan Jin, Do-Hyeun Kim

요 약 최근 IETF(Internet Engineering Task Force) CoRE(Constrained RESTful Environment) 워킹그룹에서 IoT 프로토콜로 CoAP(Constrained Application Protocol)을 표준으로 채택하고 있다. CoAP 프로토콜은 작은 용량의 메모리와 저전력 등 제한된 환경에서 IoT 임베디드 노드 간의 통신을 지원하고 있다. 본 논문에서는 모바일 환경에서 CoAP 프로토콜을 이용하여 IoT 임베디드 노드와 이동 단말을 연결성을 지원하기 위해 RD(Resourse Directory) 기반의 등록 서비스를 설계하고 구현한다. 이동 단말과 IoT 노드 사이에 RD를 두고, 이를 통해 이동 단말은 IoT 노드를 검색하고 상황 정보를 습득할 수 있다. 이때 이동 단말은 CoAP 클라이언트를 갖고 있으며, IoT 임베디드 노드는 CoAP 서버를 갖고 있어 이동 환경에서 제한된 IoT 노드들을 연결하여 편리하게 상황 정보를 습득하고 사물을 제어할 수 있다.

Abstract Recently, IETF (Internet Engineering Task) working group has adopted CoAP (Constrained Application Protocol) as a standard IoT proctocol. CoAP is a specialized web transfer protocol for use with constrained nodes and constrained environment such as small memory and low power networks. In this paper, we design and implement a registration service with CoAP protocol based on RD(Resource Directory) to connect IoT nodes in mobile Internet environments. The resource directory between the mobile terminal and IoT nodes provides to discover the IoT nodes and get the context data. The mobile terminal has as the CoAP client and embedded IoT nodes includes as the CoAP server so that it can conveniently manage the constrained IoT nodes to get the context data and control devices in a mobile environments.

Key Words : CoAP, Resource Directory, IoT, Mobile

Ⅰ. 서 론

최근 전 세계적으로, IoT(Internet of Things) 기술을 이용하여 모든 사물을 인터넷 표준 통신 프로토콜로 상호 연결하고자 한다. IoT는 사물을 인터넷에 연결하여 현실과 가상세계의 모든 정보를 상호작용하는 환경을 제

*준회원, 제주대학교 컴퓨터공학과
**종신회원, 제주대학교 컴퓨터공학과
접수일자: 2016년 1월 3일, 수정완료: 2016년 2월 5일
게재확정일자: 2016년 2월 5일

Received: 3 January, 2015 / Revised: 5 February, 2016 /
Accepted: 5 February, 2016
**Corresponding Author: kimdh@jejunu.ac.kr
Dept. of Computer Engineering, Jeju National University, Korea

공할 수 있으며, 우리 주변의 사물로부터 상황 정보를 획득할 수 있다[1,2]. 현재 IoT 응용 통신 프로토콜로 HTTP, CoAP(Constrained Application Protocol), MQTT등에 대한 표준화가 진행되고 있다. 이들 중 IETF CoRE(Constrained RESTful Environments) 워크그룹에서 작은 용량의 메모리와 저전력 등 제한된 환경에서 센서나 구동체 노드 간의 통신을 지원하는 CoAP 프로토콜을 표준화하고 있다. CoAP프로토콜은 UDP 기반의 응용 통신 프로토콜이며, 작은 자원을 가진 IoT 노드에서 인터넷을 통해 데이터 교환을 최적화하고 있다. 이를 통해 소형, 저전력 센서 노드로부터 온도, 습도, 광합성도, 기울기, 오염 등과 같은 상황 데이터를 전달할 수 있다[3].

또한 이동 단말에서 안드로이드와 iOS 를 이용하여 자동차, 가전제품 등의 사물 상황을 파악하고 제어하는 연구도 진행되고 있다. 향후 모바일 환경에서 이동 단말을 이용하여 상황을 인지하고 사물을 제어하는 프로토콜로 IETF의 표준 CoAP 프로토콜이 될 것으로 예상된다.

본 논문에서는 모바일 인터넷 환경에서의 CoAP 프로토콜을 이용한 RD(Resourse Directory) 기반의 IoT 노드와 이동 단말 연결 구조를 제시하고, RD 기반의 이동 단말 등록 서비스를 설계하고 구현한다. 이를 통해 작은 용량의 메모리와 저전력 등 제한된 환경에서 센서나 구동체 노드와 이동 단말 간의 통신을 지원하고, 상황 데이터를 획득할 수 있을 것으로 예상한다. 서론에 이어 2장에서는 CoAP 프로토콜과 RD 등 관련 연구를 서술한다. 3장에서는 모바일 환경에 CoAP 기반의 RD를 이용하여 IoT노드 등록 서비스를 설계한다. 그리고 4장에서는 구현한 내용과 실험 결과를 보여주고, 마지막으로 5장에서는 결론을 맺는다.

II. 관련연구

IETF CoRE(Constrained RESTful Environment) 워킹 그룹에서 2010년 초부터 본격적으로 개발되기 시작한 CoAP 프로토콜은 여러 차례의 변화를 거쳐 2014년 RFC 7252를 제정했다. CoAP 프로토콜은 저성능 CPU, 작은 저장장치, 저전력 등의 노드 제약 조건과, 높은 데이터 손실과 느린 데이터 전송 등의 네트워크 제약 환경에서 데이터 전송하기 위한 응용 프로토콜이다. 현재 제한된 리소스를 갖는 센서 노드 등에 적용하기 위해 CoAP 프로토콜을 개발하고 있다.

그리고 CoAP서버는 기존의 인터넷 상의 컴퓨터 서버가 아니라 주위 상황 정보를 획득하는 노드를 말한다. 그리고 클라이언트는 일반 인터넷 상에서 상황 정보를 요청하는 단말을 의미한다. CoAP 서버와 클라이언트 사이에서 존재할 수 있는 CoAP 프락시는 데이터를 수신하여 메시지를 전달하는 중개 역할을 수행한다. CoAP 프락시는 HTTP 메시지를 CoAP 메시지로 상호 전환하여 사용자의 웹 브라우저에 전송하여 상황 정보를 보여주고 있다. 모바일 환경에서는 프락시를 거치지 않고 직접 클라이언트가 IoT노드에 접근할 수 있다.

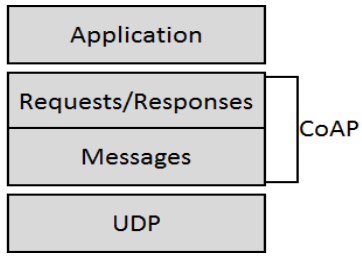


그림 1. IETF CoAP 프로토콜 스택
Fig. 1. IETF CoAP Protocol Stack

CoAP 프로토콜은 UDP 상에서 동작하고, 요청과 응답 (request/response) 방식으로 데이터를 전달한다. CoAP 프로토콜은 그림 1과 같이 CoAP은 응용층과 UDP 층 사이에 Request/Response층과 Message층으로 이루어져 있다. CoAP의 Message층은 UDP 기반으로 두 노드간의 상호연동을 지원한다[4].

최근 IoT 네트워크 환경에서 CoAP 프로토콜을 이용하여 인터넷을 통해 IoT 노드를 관리하고 검색하는 RD에 대한 연구가 진행되고 있다. IETF CoRE 워킹 그룹에서는 RD를 제안하고 있다[5].

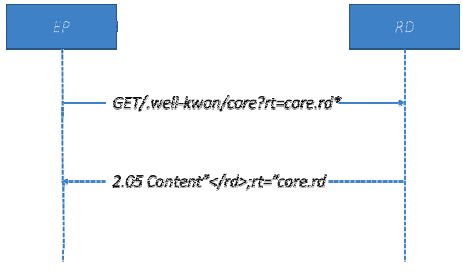


그림 2. EP의 검색 과정
Fig. 2. Discovery Process of Endpoint

RD는 웹 링크의 저장소와 웹 서버 역할을 수행할 수 있으며, RD는 여러 개의 REST 인터페이스를 제공하여 기타 EP(End Point)들이 등록 및 조회를 할 수 있게 한다. 여기서 EP를 IoT 노드가 될 수 있다. CoAP 환경에서는 노드는 하나 이상의 EP가 존재할 수 있다. 하나의 EP는 IP와 포트를 가고 있다. IoT 노드는 RD를 찾아서 자신의 노드 정보를 등록, 갱신, 삭제 및 조회할 수 있다

RD의 주요 기능은 EP의 RD 검색과 노드 등록, 갱신, 조회 및 삭제 등이 있다. 이 기능들을 제공하기 위하여 RD는 REST인터페이스를 제공하며, 이때 각각의 EP가 RD에 등록할 수 있고, EP 그룹으로 등록할 수 있다.

그림 3. EP의 등록 과정
Fig. 3. Registration Process of Endpoint

EP가 RD에 등록하기 전에 RD의 IP 주소와 포트를 알고 있어야 하며, 이를 위해 EP는 등록 인터페이스를 이용하여 RD의 주소를 비롯한 자원 정보를 검색하여 얻어올 수 있다. 이때 EP는 GET메소드를 이용하여 RD를 조회하여 주소를 가져 올 수 있다. REST 인터페이스를 통하여 EP가 RD를 조회하여 주소를 가져 올 수 있다. 그림 2은 REST인터페이스를 통하여 EP가 RD를 검색하는 예이다.

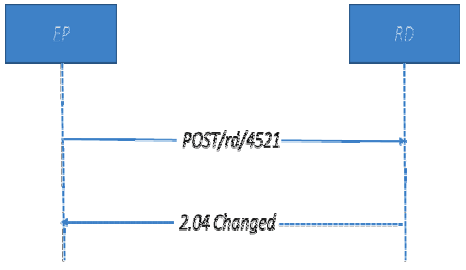


그림 4. EP 자원 정보의 갱신 과정
Fig. 4. Updation Process of Endpoint Resource

EP는 RD를 검색한 다음 RD의 등록 인터페이스를 통하여 EP를 등록할 수 있다. EP는 POST방식으로 등록을 요청하며 이때 EP 자원 목록을 전달한다. 이 자원 목록은 RFC6690이나 JSON CoRE Link Format으로 될 수 있다. 그림 3은 EP의 등록 과정을 보여주고 있다.

EP는 POST나 PUT 방식으로 RD에 등록한 자신의 정보를 갱신할 수 있다. 이때 EP의 생명주기나 상세정보를 갱신할 수 있다. 그림 4는 EP 자원 정보의 갱신 과정을 보여주고 있다. 여기서 요청메시지는 RD의 4521라는 자원 주소와 함께 페이로드에 갱신 정보를 포함하여 전달한다. 데이터 전송이 성공적인 갱신이 이루어질 경우 2.04 메시지를 반환한다.

RD에 등록한 EP들은 생명주기가 끝나면 RD에서 삭제되며 이 EP의 자원 주소도 삭제된다. 하지만 생명주기가 끝나기 전에 EP를 삭제할 필요가 있을 경우 EP는 RD에게 삭제 요청을 한다.

Ⅲ. IoT 노드 등록 서비스 설계

RD 기반 IoT 노드와 이동 단말의 CoAP 프로토콜 스택을 그림 5와 같이 보여주고 있다. 여기서의 물리계층은 유무선 근거리통신망이나 무선 PAN(Personal Area Network)을, IP와 UDP를 적용할 수 있다.

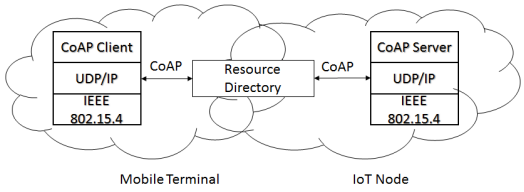


그림 5. RD 기반 IoT 노드와 이동 단말의 CoAP 프로토콜 스택
Fig. 5. CoAP Protocol Stack Based on RD of IoT Node and Mobile Terminal

그림 5에서는 물리계층은 IEEE 802.15.4 를 이용하고, 그 위에 응용 계층에 IoT 노드는 CoAP 서버를, 이동 단말에는 CoAP 클라이언트를 구현할 수 있다. 그리고 IoT 노드는 RD를 통해 자신을 등록하고, 이동 단말은 IoT 노드를 검색하여 상황 정보를 수집할 수 있다.

그림 6은 IoT 노드의 CoAP 서버의 동작과정을 보여주고 있다. 먼저 CoAP 서버에게 IP 주소를 할당한다. 그

리고 노드 정보를 수집하기 시작되고, 서버에서 RD로 POST 방식으로 Request 메시지를 이용하여 요청하고 URI경로가 등록된다. 이때 등록 자원 값이 페이로드에서 저장하여 전송된다. 그리고 RD는 서버에서 Request를 받은 다음에 노드 ID정보는 IoT 노드 정보를 수정하고 성공 메시지 2.04를 반환한다. 그리고 IoT 노드 ID정보가 존재하지 않는 경우에는 새로운 IoT 노드 정보를 추가하고 성공 메시지 2.01를 반환한다. 마지막으로 CoAP 서버에서 리소스를 초기화 상태로 설정한다.

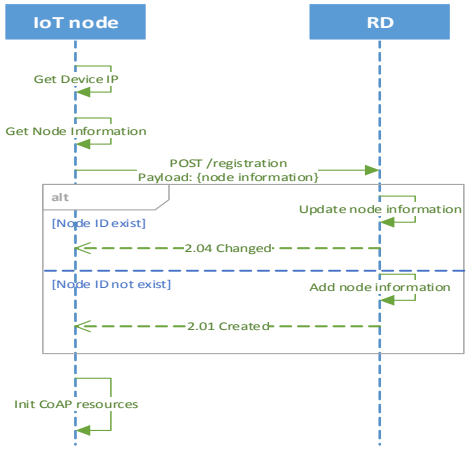


그림 6. IoT 임베디드 노드 등록 흐름도
Fig. 6. Node Registration Sequence Diagram

그림 7은 이동 단말의 동작 과정을 보여주고 있다. 먼저 IoT 노드의 CoAP 서버에서 RD로 GET 방식으로 Request 메시지를 요청하고 URI경로로 검색한다. IoT 노드 정보 리스트 값을 추가하고 이동 단말에 전송한다. 그리고 사용자는 IoT 노드를 선택하고, 해당 IoT 노드의 세부 유닛을 확인하고 선택한다. 그리고 IoT 노드와 세부 유닛을 제어할 수 있다. 예를 들어 사용자가 온도 조회 기능을 선택할 경우 이동 단말은 RD로 GET 방식으로 Request 메시지를 전송하고 unit001의 get_tmp이라는 리소스 주소를 받는 다. 이를 주소를 이용하여 IoT 노드에 온도 값을 요청하면, 온도 값을 받을 수 있다.

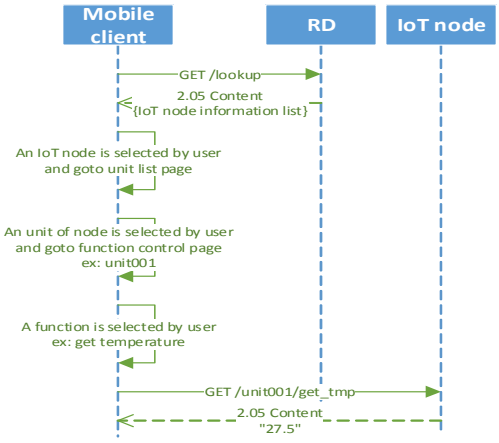


그림 7. IoT 노드로부터 상황 데이터 수집 흐름
Fig. 7. Sequence Diagram for Collection of Context Data from IoT Node

IV. IoT 노드 등록 서비스 구현 및 결과

CoAP 기반의 통신을 구현하기 위하여 C언어 기반의 Libcoap 라이브러리와 JAVA언어 기반의 Californium(Cf) 프레임웍을 사용한다. Libcoap은 RFC7252의 표준에 의하여 작성되었으며 Cf와 같은 기타 프로그램언어로 작성한 라이브러리와 통신이 가능하다. IoT 노드에서 실행하는 프로그램은 인텔에서 제공하는 리눅스 환경에서 실행하므로 우리는 Libcoap을 이용하여 CoAP통신을 제공한다. 이동 단말은 안드로이드 플랫폼을 이용하고 RD는 자바 프로그램으로 작성한다. IoT 임베디드 노드에 CoAP 서버, 사용자 단말에 CoAP 클라이언트, RD를 구현하여 상호 통신, IoT 노드 등록/추가/삭제/검색 기능 및 동작을 검증한다[6,7].

그림 8는 IoT 노드에서 실행 결과를 보여주고 있으며, Resource Directory로부터 CoAP 메시지를 볼 수 있다. 여기서 CoAP 메시지는 CoAP 헤드와 Response code를 보여주고 있다.

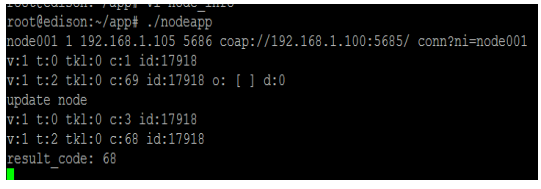


그림 8. IoT 노드의 CoAP 메시지 통신화면
Fig. 8. CoAP Message Communication of IoT Node

그림 9는 RD의 실행 결과를 보여주고 있으며, 이를 통하여 사용자 단말과 IoT 노드에서 요청한 CoAP 메시지는 CoAP 헤드와 Response code와, Option과 Payload 크기 및 값을 보여주고 있다.

IoT 임베디드 노드는 인텔 에디슨(Edison) 보드를 이용하여 그림 10과 같이 구현한다. IoT 노드에 CoAP 서버가 구현되며, CoAP 서버를 통해 사용자 단말의 CoAP 클라이언트와 통신하고, 수집된 상황 정보를 제공할 수 있다. 여기서 IoT 노드는 LED와 온도 센서를 갖고 있다.

```
==[ CoAP Request ]=============================================
MID     : 17918
Token   :
Type    : CON
Method  : GET
Options: {"Uri-Port":5685, "Uri-Path":"conn", "Uri-Query":"ni=node001"}
Payload: 0 Bytes
================================================================
==[ CoAP Response ]============================================
MID     : 17918
Token   :
Type    : ACK
Status  : 2.05
Options: {"Content-Format":"text/plain"}
Payload: 1 Bytes
----------------------------------------------------------------
0
```

그림 9. RD의 CoAP 메시지 통신화면
Fig. 9. CoAP Message Communication of Resource Directory

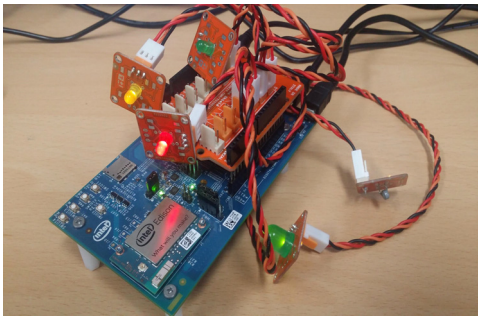


그림 10. IoT 임베디드 노드
Fig. 10. IoT Embedded Node

그리고 사용자의 이동 단말에서 CoAP 클라이언트를 안드로이드 5.0 를 이용하여 개발한다. CoAP 클라이언트를 구성하기 위하여 CF CoAP 프레임워크를 사용한다. CoAP 클라이언트는 NodeList, NodeDetail, Actuator, Sensor, Nodeupdate 액티비티 등 6 액티비티로 구성한다.

그림 11. 이동 단말의 Nodedetail 화면
Fig. 11. Nodedetail View in Mobile Terminal

그림 11에서 Nodedetail 화면을 보여 준다. 이는 노드 정보를 보여 줄 뿐만 아니라 각 노드정보도 보여 줄 수 있다. 여기에 노드타입은 센서와 구동체로 구분한다.

그림 12에서 IoT 임베디드 노드로부터 수집된 온도 센서 데이터를 이동 단말에서 보여 준다. Get data 버튼을 누르면 실시간 온도 정보를 받을 수 있다. 여기서 IoT 노드를 삭제하려면 Nodelist화면에서 가상 노드를 길게 누른 다음에 삭제 버튼을 누르면 된다. 노드 삭제 후에는 화면에서 No Response 메시지가 나타난다.

그림 12. 이동 단말의 센서 화면
Fig. 12. Sensor View in Mobile Terminal

그림 13에서 IoT 노드 갱신 화면을 보여주고 있다. 정보를 수정한 다음에 업데이트 버튼을 누르며 업데이트 요청을 RD에게 보내서 해당 노드의 정보를 갱신한다.

그림 13. 이동 단말의 Nodeupdate 화면
Fig. 13. Nodeupdate View in Mobile Terminal

V. 결 론

최근 모바일 환경에서 CoAP 프로토콜 기반의 IoT노드와 이동 단말 간의 연결성 연구가 진행되고 있다. 본 논문에서는 모바일 인터넷 환경에서 CoAP 프로토콜을 이용하여 센서와 구동체를 갖는 IoT 임베디드 노드를 RD에 등록하여 사용자가 이동 단말을 통하여 등록된 IoT 노드를 조회하고 제어하는 구조를 제시한다. 더불어 모바일 환경에서 CoAP 서버를 갖고 있는 IoT 임베디드 노드, CoAP 클라이언트를 포함하는 이동 단말, RD 기반의 등록 서비스를 개발하여 이들 간의 등록과 상호 통신할 수 있다. 이를 통해 이동 환경에서도 저전력, 작은 메모리용량 혹은 패키지 손실이 높은 제한된 네트워크에서 사용자와 IoT 노드 간의 연동할 수 있다. 더불어 이동 단말을 통해 센서의 실시간 상황 정보를 모니터링 하거나 여러 구동체를 제어할 수 있다.

Reference

[1] Z. Shelby, B. Frank, D. Sturek, “Constrained Application Protocol (CoAP)”, RFC 7252, June, 2014.

[2] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

[3] Hartke, K., "Observing Resources in CoAP", draft-ietf- core-observe-16 (work in progress), December 2014.

[4] Christian Lerche, Nico Laum, Frank Golatowski, Dirk Timmermann, “Connecting the Web with the Web of Things: Lessons Learned From Implementing a CoAP-HTTo Proxy”, Mobile Adhoc and Sensor Systems (MASS), 2012 IEEE 9th International Conference on, Oct 2012

[5] Shelby, Z., Koster, M., Bormann, C., and P. Stok, "CoRE Resource Directory", draft-ietf-core-resource-directory-05 (work in progress), June 2015.

[6] Wen-Quan JIN, Do-Hyeun Kim “Implementation and Experiment of CoAP Protocol Based on IoT for Verification of Interoperability”, The Journal of The Institute of Internet, Broadcasting and Communication (JIIBC), Vol. 14 No. 4, August 2014.

[8] Jian Wang, June SaKong, Ho-Young Kwak, Do-Hyeun Kim.“Design and Implementation of IoT Middleware Using Data Refinement Scheme based on IETF CoAP”, The Journal of The Institute of Internet, Broadcasting and Communication (JIIBC), Vol. 15 No. 6, Dec. 2015.

저자 소개

항 뢰(준회원)

- 2011년 9월 ~ 2015년 8월 : 제주대학교 컴퓨터공학과 학사
- 2015년 9월 ~ 현재 : 제주대학교 대학원 컴퓨터공학과 석사과정

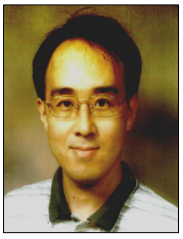
<주관심 분야> : 사물인터넷, 모바일 네트워크, 웹 서비스

김 문 권(준회원)

- 2008년 9월 ~ 2013년 8월 : 연변과학기술대학 전산학과 학사
- 2013년 9월 ~ 2015년 2월 : 제주대학교 대학원 컴퓨터공학과 석사
- 2015년 3월 ~ 현재 : 제주대학교 대학원 컴퓨터공학과 박사과정

<주관심 분야> : 사물인터넷, 임베디드 시스템

김 도 현(종신회원)



- 1990년 3월 ~ 1995년 3월 : 국방과학연구소 전산망연구실 연구원
- 2004년 9월 ~ 현재 : 제주대학교 공과대학 컴퓨터공학과 교수
- 2013년 1월~현재 : 대한전자공학회 M2M/IoT 연구회 회장

<주관심 분야> : 서비스 컴퓨팅, 사물인터넷, 모바일 컴퓨팅

※ "이 논문은 2015년도 제주대학교 학술연구지원 사업에 의하여 연구되었음"
교신저자 : 김도현(Dept. of Computer Engineering, Jeju National University)