아두이노 기반 사물인터넷을 위한 통합 프로그래밍 방법에 대한 연구[1)]

김가영 최광훈

전남대학교 전자컴퓨터공학부
kirayu15@gmail.com kwanghoon.choi@jnu.ac.kr

창병모

숙명여자대학교 소프트웨어학부 컴퓨터과학전공
chang@cs.sookmyung.ac.kr

A Study on Unified Programming Method for Arduino-based Internet of Things

Gayoung Kim Kwanghoon Choi
Dept.Electronics and Computer Engineering, Chonnam National University, Gwangju

Byeongmo Chang
Dept.of Computer Science, Sookmyung Women's University, Seoul

요 약

본 논문에서는 아두이노 기반 사물인터넷 프로그램 개발을 위한 통합 프로그래밍 환경을 설계하고 구현하는 것을 목표로 진행 중인 연구 내용을 요약한다. 사물인터넷(Internet of Things)의 등장은 하나의 장치가 아니라 여러 개의 다양한 분산 장치에서 프로그래밍 하는 환경을 보편화한다. 다양한 장치에서 프로그램을 하나씩 개발하면 프로그래머의 부담이 증가하고 모든 프로그램을 테스트 하는 것이 복잡해지는 문제점이 있다. 이러한 문제점을 해결한 하나의 언어로 통합된 프로그래밍 환경을 설계하고 구현하였다. 사물인터넷 통합프로그래밍 언어를 테스트 할 수 있는 환경을 만들어 평가하였다. 제안하는 방법의 장점은 프로그래머가 단일 컴퓨터와 하나의 언어로 코드를 작성하고 직접 실행하여 테스트 할 수 있는 환경을 제공한다.

1. 서 론

모바일 시대를 넘어 다가온 초연결 사회는 인터넷을 통해 사람 간의 연결, 사람과 사물, 사물 간의 연결되어 정보가 생성되고 공유/활용되는 새로운 사회를 말한다. 이러한 초연결 사회를 이끄는 사물인터넷은 각광을 받고 있다. 사물과 주위 환경으로부터 정보를 얻는 센싱기술, 사물이 인터넷에 연결되도록 지원하는 유무선 네트워크 기술, 각종 서비스 형태에 적합하게 정보를 가공하고 처리하는 서비스 인터페이스 기술이 핵심적이다. 초연결 사회는 이제 단일 컴퓨터가 아닌 다른 여러 분산 컴퓨터를 이용하여 조화롭게 프로그래밍 하는 작업을 필요로 한다.

그러나 다른 여러 분산시스템을 이용한 사물인터넷 프로그램을 개발할 때, 프로그래머는 3가지의 문제를 겪게 된다. 첫째 사물인터넷 시스템 장치들의 네트워크를 어떻게 구성할 것인지, 둘째 다양한 장치들에서 프로그램을 어떻게 개발할 것인지, 마지막으로 프로그램을 개발할 때 어떻게 진행하고 업데이트할 것인지에 대한 문제를 겪게 된다. 분산된 다양한 장치에 각각의 프로그램을 개발하는 것은 프로그래머의 부담을 증가시키는 요인이다. 또한 분산된 프로그램으로 인해 실행 시 발생하는 문제를 찾거나 테스팅이 복잡하며 어려워진다.

본 논문에서는 앞에서 언급한 문제 중 세 번째 문제를 해결하기 위한 아두이노를 기반 사물인터넷 통합 프로그래밍 언어를 Python으로 설계하고 구현하였다. 또한 통합 프로그래밍 언어 동작을 평가하기 위해서 사물인터넷 개발에 주로 사용되는 장치들을 이용하여 홈 모니터링 시스템 환경을 만들어 실험한 내용을 요약한다. 하나의 장치에서 다양한 분산시스템을 이용한 사물인터넷 프로그램을 개발하고 테스트 할 수 있는 환경을 제공하는 데 목적이 있다.

2. 관련 연구

아두이노와 라즈베리 파이를 위한 고급 프로그래밍모델들이 제안되었다. Arduino Service Interface Programming(ASIP) [1] 모델은 아두이노와 같은 마이크로컨트롤러에서 실행되는 코드의 소프트웨어 아키텍처를 제공한다. 이 모델은 마이크로 컨트롤러들과 클라이언트들 간의 메세지 전달을 위한 텍스트 프로토콜을 포함한다. IBM의 오픈 소스 프로젝트인 Node-Red(NR) [2]는 IoT 어플리케이션과 서비스들을 구축하기 위한 데이터 플로우 프로그래밍 모델을 사용한다. 그러나 이 모델은 라즈베리 파이와 같은 개별적인 장치들의 런타임으로 설계되었으며

1) 이 논문은 2017년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.2017R1A2B4005138).

Node.js 기술을 가진 Java Script를 이용하여 개발되었다.

사물인터넷을 개발하기 위한 통합된 프로그래밍 모델들이 있다. Fabryq[3]는 개발자가 임베디드-게이트웨이-클라우드 프로그램에서 RPC 함수 호출을 통해 클라우드와 임베디드 장치가 상호 작용하는 집중된 Java Script 프로그램으로 작성할 수 있다. 이와 다른 많은 기술들은 개발자가 MGC 프로그램을 신속하게 탐색하고 프로토타입을 만들 수 있도록 한다. Ravel [4]은 모델-뷰-컨트롤러 아키텍처의 분산된 확장을 이용하여 3계층의 응용 프로그램을 프로그래밍 하기 위한 방법을 제안한다. 이는 특정 계층의 속성과 구현을 포함하는 새로운 공간을 추가한다.

다양한 장치를 통합하여 개발할 수 있는 모델들이 많이 제안되고 있지만, 프로그래밍 언어에 대한 이해도가 높아야 사용이 가능하다. 이 연구에서 제안하는 통합 프로그래밍 환경은 Python 기반으로 설계, 구현되어 보다 쉽게 사물인터넷 개발 환경을 만들고자 한다.

3. 사물인터넷 통합 프로그래밍 환경 설계 및 구현

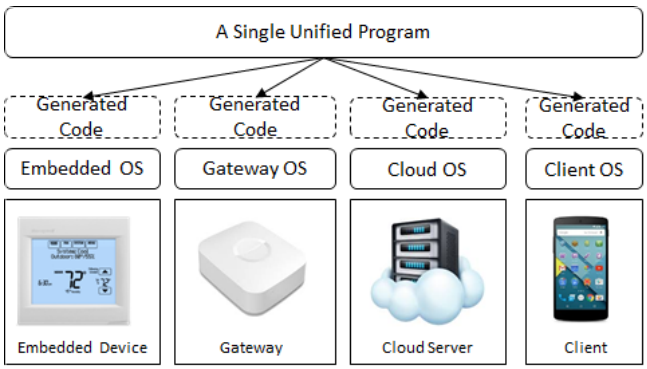

그림 1. IoT 프로그램에 대한 통합된 언어 접근 방식

(그림 1)은 통합 사물인터넷 프로그램 실행 구성도이다. 사용자가 통합된 사물인터넷 프로그램과 함께 통신 방법을 제시해주면 통합 프로그래밍 환경에서 사용자가 제시한 통신방법과 옵션에 맞춰 프로그램을 위치별로 분할해준다.

(그림 2)는 간단한 홈 모니터링 시스템이다. 해당 시스템에는 다양한 센서를 가진 아두이노 기반의 임베디드 시스템, MySQL이 동작하고 있는 PC기반의 클라우드 서버, Java 프로그램이 동작하고 있는 클라이언트, 그리고 임베디드 시스템, 클라우드와 클라이언트 사이에 있는 라즈베리 파이 기반의 게이트웨이가 있다.

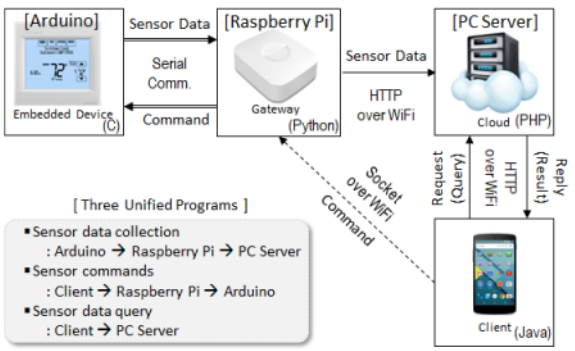

그림 2. 홈 모니터링 시스템 예시

표 1. 장치간의 통신 방법

From	To	Protocol
Arduino	Raspberry	Serial
Raspberry	Arduino	Serial
Raspberry	Cloud	HTTP
Client	Raspberry	Socket
Cloud	Client	HTTP
Client	Cloud	HTTP

임베디드 시스템에 사용된 센서는 문/창문의 여닫힘을 알 수 있는 마그네틱 센서, 비밀번호를 입력할 수 있는 4*4 키패드, 비밀번호 입력상태와 메세지를 출력하는 LCD, 경보음과 경고등에 Piezo 센서와 LED, 사람을 인식하는 인체감지센서, 카메라를 회전시키는 서보 모터가 있다.

(그림 2)에 나타낸 장치들은 (표 1)과 같이 다른 통신방법을 사용한다. 아두이노와 라즈베리 파이는 Serial 통신, 라즈베리 파이와 PC는 HTTP 통신, 라즈베리 파이와 클라이언트는 Socket 통신, PC와 클라이언트는 HTTP 통신을 사용한다.

(그림 2)에서 언급한 홈 모니터링 시스템을 사물인터넷 통합 프로그래밍 언어를 이용하여 작성할 수 있다. 사물인터넷 통합 프로그래밍 언어는 Python을 이용하여 작성할 수 있으며 Python은 쉽게 배울 수 있는 가장 잘 알려진 프로그래밍 언어 중 하나이기 때문에 선택했다.

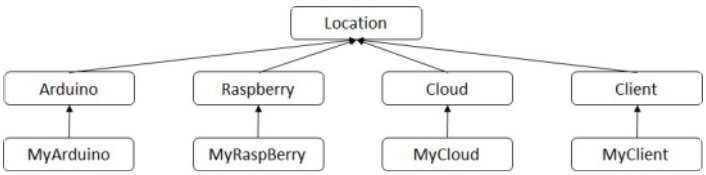

그림 3. 위치를 나타내는 통합 프로그램 클래스

(그림 2)에서 사용된 장치들의 위치 정보를 표현하기위해서 (그림 3)과 같이 클래스 개념을 이용한다. 예를 들어, MyArduino는 Arduino 위치 클래스를 확장하여 통합 프로그램으로 작성된다. Arduino 위치 클래스를 더 확장하고 싶다면 MyArduino가 아닌 다른 클래스 명을 사용하면 확장할 수 있다. 다른 클래스의 확장 또한 동일하게 사용할 수 있다.

(그림 2)의 홈 모니터링 시스템 중 문의 여닫힘 정보를 모으는 프로그램을 (그림 4)와 같이 Python으로 작성할 수 있으며, 확장된 위치정보에 따라 작은 프로그램으로 분할된다. 통합 프로그램에서 사용되는 라이브러리들은 _import_list와 같이 제시하여 사용할 수 있다. 여기서 아두이노는 Python을 통한 프로그래밍을 지원하지 않아 PrettyPrinter를 이용하여 최종 결과물을 얻어내 실행할 수 있다.

(그림 4)에서와 같이 다른 장치 간의 통신은 원격 함수 호출을 통해 이뤄지며, 이를 제외하고 기존과 동일한 프로그래밍 방법이 적용된다. 또한 각 장치에서 제공하는 기본 함수를 작성하여 실제로 동작하는 것처럼 속여 결과를 확인할 수 있다. 예를 들어, 아두이노는 digitalRead(), pinMode() 등을 기본적으로 제공해주고 있는데 이를 직접 작성할 수 있다. 그리고 처음 setup()을 호출한 후 loop()

를 계속적으로 호출할 수 있도록 작성할 수 있다.

```
class MyArduino(Arduino):
        _int_doorPin = 2
        _int_preState = LOW
        _int_postState = LOW

        def _void_setup():
                pinMode(doorPin, INPUT)

        def _void_loop():
                readDoorSensor()

        def _void_readDoorSensor():
                postState = digitalRead(doorPin)

                if postState == HIGH and preState == LOW:
                        MyRaspberry.reqSend("0")
                        preState = postState

                if postState == LOW and preState == HIGH:
                        MyRaspberry.reqSend("1")
                        preState = postState


class MyRaspberry(Raspberry):
        _import_list = ['urllib.request']
        _url = "http://168.131.152.196/common.php"

        def reqSend(val):
                if val == 48:  # ''0''
                        pic_bin = takeAPhoto()
                        MyCloud.recordDoorState(_url , 'c', pic_bin)
                if val == 49:  # ''1''
                        pic_bin = takeAPhoto()
                        MyCloud.recordDoorState(_url, 'o', pic_bin)

        def takeAPhoto():
                f = open(imgPath, 'r+')
                pic_bin = f.read()

                return pic_bin
class MyCloud(Cloud):
        _import_list = ['os', 'pymysql']

        def recordDoorState(openclose, pic_bin):
                d_pic_loc = save_picture(pic_bin)

                isSuccess = save_doorState(openclose, d_pic_loc)

        def save_picture(pic_bin):
                num = 1
                pic_loc = str(num) + ".jpg"

                while os.path.isfile(pic_loc):
                        num = num+1
                        pic_loc = str(num) + ".jpg"

                return pic_loc

        def save_doorState(openclose, d_pic_loc):
                conn = pymysql.connect(host='localhost', user='user', password='password',
                        db='dbname', charset='utf8')
                try:
                        with conn.cursor() as cursor:
                                sql = "insert into doorlist(d_openclose, d_pic_loc)
                                        values(%s, %s)"
                                cursor.execute(sql, (openclose, d_pic_loc))
                                conn.commit()
                        conn.close()
                        return True
                except:
                        conn.close()
                        return False
```

그림 4. 문의 마그네틱 센서 정보를 모으는 통합 프로그램 예시

4. 논의 사항

현재 사물인터넷 통합 프로그래밍 개발환경은 통신방법을 임의로 컴파일러 내에 제시해주는 방법을 사용하고 있다. HTTP, Serial, Socket 통신 방법에 대해서는 컴파일 코드가 존재하지만, BLE, Zigbee 등과 같은 다양한 통신 방법에 대한 컴파일 코드가 없어 통신 방법 제시에 제약이 매우 크다. 모든 통신 방법에 대해서 컴파일 코드를 작성하기에는 통신에 필요한 내용이 달라 프로그램 작성이 복잡하다. 기본적인 통신 방법을 사용하여 사물인터넷 프로그램을 작성하기에는 문제가 없지만 다양한 통신 방법을 이용할 때 통신 제약으로 어려움을 겪을 수 있다.

(그림 5)와 같이 프로그램에 통신과 관련된 사용 방법을 모두 제시해주는 방법을 이용한다면 프로그램을 자주 수정하고 실행할 때는 빠르고 편할 것이다. 그러나 통신 방법을 바꾼다면 수정해야 할 부분이 많아질 것이다. 그에 비해 프로그램 입력으로 json과 같은 파일을 넣어준다면 통신 방법의 수정이 매우 편할 것이며 다양한 통신 방법을 이용할 수 있어 통신 방법의 제한적인 문제점을 해결할 수 있을 것으로 보인다.

컴파일러 내 직접 제시 방법

```
commuTable = {
                'Arduino' : {'Arduino' : None, 'Raspberry' : 'Serial',
                             'Cloud' : 'http', 'Mobile' : 'Bluetooth'},
                'Raspberry' : {'Arduino' : 'Serial', 'Raspberry' : None,
                               'Cloud' : 'http', 'Mobile' : 'Socket'},
                'Cloud' : {'Arduino' : None, 'Raspberry' : 'http',
                           'Cloud' : None, 'Mobile' : 'Socket'},
                'Mobile' : {'Arduino' : None, 'Raspberry' : 'Socket',
                            'Cloud' : 'http', 'Mobile' : None}
               }
```

프로그램 입력 제시 방법

```
{'comm' : [
            {'from' : 'Arduino', 'to' : 'Raspberry', 'commu' : 'Serial',
             'option' : ['baudrate=9600']},
            {'from' : 'Raspberry', 'to' : 'Arduino', 'commu' : 'serial',
             'option' : ['device=/dev/ttyACM0', 'baudrate=9600']},
            {'from' : 'Raspberry', 'to' : 'Cloud', 'commu' : 'HTTP',
             'option' : ['ipaddress = http://192.168.0.14/common.php',
             'data transfer = post']}
```

그림 5. 통신 방법의 직접 제시와 사용자 임의 제시 비교

5. 결론 및 향후 연구

본 논문에서 아두이노를 기반으로 하는 사물 인터넷 통합 프로그래밍 언어를 설계하고 구현하였다. 또한 아두이노, 라즈베리 파이, Web기반 DB 서버를 이용한 홈 모니터링 시스템 환경에서 통합 프로그래밍 언어의 동작을 확인했다. 통합 프로그래밍 방법은 프로그래머가 장치 위치를 나타내는 클래스 개념을 사용하고 매우 간단한 함수 호출을 통해서 개발이 가능하도록 하는 것이다. 이는 분산된 장치에 프로그램을 개발하고 테스팅을 해야만 했던 프로그래머들의 부담을 줄일 수 있을 것이다.

향후 통합 사물인터넷 프로그래밍 언어의 통신방법 확장 및 입력에 대한 논의사항들을 해결할 예정이다.

6. 참고문헌

[1] G. Barbon, M. Margolis, F. Palumbo, F. Raimondi, and N. Weldin. Taking Arduino to the Internet of Things: The ASIP programming model. Computer Communications, 89-90:128-140, 2016.

[2] IBM. Node-Red. https://nodered.org/.

[3] W.McGrath, M.Etemadi, S.Roy, and B.Hartmann. Fabryq: Using phones as gateways to prototype internet of things applications using web scripting. In Proceedings of the 7th ACM SIGCHI symposium on Engineering Interactive Computing Systems, EICS 2015, Duisburg, Germany, June 23-26, 2015, pages 164-173, 2015.

[4] L.Riliskis, J.Hong, and P.Levis. Ravel: Programming IoT Applications As Distributed Models, Views, and Controllers. Proceedings of the 2015 International Workshop on Internet of Things Towards Applications, pages 1-6, 2015.