WSDL 2.0을 이용한 ETSI 표준 M2M 플랫폼과 이종의 웹 애플리케이션 간의 서비스 차원의 통합 방안 연구

박윤정¹ · 민덕기^{2*}

Research on Service-Level Integration of ETSI M2M Platform with Heterogeneous Web Applications based on WSDL 2.0

Yunjung Park¹ · Dugki Min^{2*}

¹ Department of Computer, Information&Communications Engineering, Konkuk University, Seoul 143-701, Korea

² Department of Computer Science and Engineering, Konkuk University, Seoul 143-701, Korea

요 약

ETSI 표준 M2M 플랫폼은 M2M을 위한 공통 서비스를 제공하는 플랫폼으로, 본 논문에서는 WSDL 2.0 표준 문서를 이용하여 ETSI 표준 M2M 플랫폼 서비스를 SOAP, HTTP 및 CoAP 형태의 웹 서비스로 제공 가능하게 함으로 서비스 차원에서 이종의 웹 애플리케이션과 연동하기 위한 방안을 제시하고 있다. 본 논문에서 제안하는 서비스 차원의 통합 방안은 (1) M2M 플랫폼의 서비스 인터페이스를 WSDL 2.0 문서로 기술하는 단계, (2) 제시된 웹 클라이언트 생성기에 따라 WSDL 2.0 문서를 토대로 SOAP, HTTP 및 CoAP 웹 클라이언트를 생성하는 단계, (3) 제시된 웹 프로토콜 변환기로 웹 서비스 실행 시 이종의 프로토콜을 상호 변환하여 주는 단계를 포함하고 있다.

ABSTRACT

ETSI M2M Platforms provide general infra services for machine-to-machine communication. In this paper, we propose a service-level integration methodology which can integrate a ETSI M2M platform with heterogeneous web applications based on WSDL 2.0. Considered heterogeneous web applications are the ones that use SOAP, HTTP and CoAP protocols for inter-communication. The proposed WSDL 2.0-based service-level integration methodology includes the followings; (1) We present a method of how to describe standardized M2M platform service interfaces into a WSDL 2.0 document. (2) A proposed web client generator is used to generates SOAP, HTTP and CoAP client codes from the WSDL 2.0 document. (3) We propose protocol conversion proxies which translate protocols into other protocols in web service run-time.

키워드 : M2M, 사물통신, WSDL 2.0, 웹 서비스

Key word : M2M, Machine-to-Machine, WSDL 2.0, Web Service

접수일자 : 2013. 07. 05 심사완료일자 : 2013. 07. 18 게재확정일자 : 2013. 08. 02

* **Corresponding Author** Dugki Min(E-mail:dkmin@konkuk.ac.kr, Tel:+82-2-340-3490)

Department of Computer Science and Engineering, Konkuk University, Seoul 143-701, Korea

Open Access <http://dx.doi.org/10.6109/jkiice.2013.17.8.1842>

print ISSN: 2234-4772 online ISSN: 2288-4165

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License(<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
Copyright © The Korea Institute of Information and Communication Engineering.

I. 서 론

기존의 사물통신(Machine-to-Machine : 이하 M2M) 시스템은 단일 M2M 서비스 사업자 별로 배포되어 동일한 사업자의 서비스만 연동되어 M2M 솔루션 간의 상호연동 제약이 발생하였다[1]. 때문에 효율적이고 개방된 M2M 서비스 제공을 위한 M2M 솔루션 통합 표준화가 다방면에서 진행되고 있다[2]. 이 중 유럽통신표준협회(European Telecommunications Standards Institute, ETSI)에서는 2009년에 M2M TC를 설립하고 M2M 엔티티 통합을 위한 M2M 플랫폼 환경 표준을 정의하고 있다[3]. ETSI의 M2M 표준은 현재 M2M 플랫폼 분야에 많은 영향을 미치고 있으며, 2012년도에 설립된 국제 M2M 협력체인 oneM2M에도 대부분의 표준이 반영되어 있다는 점에서 주목할 만하다[4].

ETSI 표준 M2M 플랫폼은 다양한 분야에서 활용 가능하지만[5,6], 그 중에서도 단말이 플랫폼에 데이터를 전송하고, 네트워크 서비스는 플랫폼으로부터 데이터를 가져와 서비스를 제공하는 데이터 전송 구조에 최적화되었다[7]. M2M 플랫폼 하단에는 단말을 위한 디바이스 애플리케이션 그리고 M2M 플랫폼 상단에는 버티컬 서비스 제공을 위한 네트워크 애플리케이션이 위치하며 이들은 상호 자신의 서비스를 제공한다.

현재의 ETSI 표준 M2M 플랫폼은 Representational State Transfer(REST) 아키텍처를 기반으로 설계되었으며, HyperText Transfer Protocol(HTTP) 및 Constrained Application Protocol(CoAP)[8] 통신을 위한 인터페이스를 정의하고 있다. 실제 M2M 플랫폼 구현 시에는 저사양 디바이스에 최적화된[9] CoAP보다는 HTTP를 이용하여 구현하며, 때문에 HTTP 프로토콜만 지원하는 M2M 플랫폼에서는 CoAP 프로토콜 기반의 디바이스 애플리케이션이나, Simple Object Access Protocol(SOAP) 프로토콜 기반의 네트워크 애플리케이션의 접근이 용이하지 않다. 따라서 저사양 디바이스를 위한 CoAP과 비즈니스 서비스에서 널리 사용되고 있는 SOAP 기반의 웹 서비스를 M2M 서비스화 하기 위해서는 M2M 플랫폼과의 별도의 연동 방안이 필요하다.

ETSI M2M 플랫폼과 SOAP, HTTP, CoAP 기반의 이종 애플리케이션의 상호 연동을 위해서는 M2M 플랫폼이 제공하는 서비스 인터페이스에 애플리케이션들이 원하는 서비스 형태로 접근 가능하여야 할 것이다. 본

논문은 실제 M2M 플랫폼 서비스 제공 방식에 영향을 미치지 않고도 SOAP 애플리케이션에게는 SOAP 서비스로, HTTP 애플리케이션에게는 HTTP 서비스로 그리고 CoAP 애플리케이션에게는 CoAP 서비스로 인식되도록 하여 서비스 차원으로 통합하는 방법을 제시한다. 그림 1은 SOAP/HTTP/CoAP 서비스의 형태로 노출된 ETSI M2M 플랫폼 통합 서비스에 대한 개념도이다.

그림 1. SOAP, HTTP, CoAP 서비스로 노출된 ETSI M2M 플랫폼 통합 서비스

Fig. 1 ETSI M2M platform services exposed as SOAP, HTTP, CoAP services for service integration

본 논문에서 제시하는 서비스 차원의 통합 방안은 Web Services Description Language (WSDL) 2.0[10] 표준을 기반으로 한다. WSDL 2.0은 기존의 SOAP을 위한 WSDL 1.1과 HTTP를 위한 Web Application Description Language (WADL)를 개선하여 SOAP과 HTTP의 웹 서비스 구체적인 기술을 모두 지원하는 문서 표준으로, 웹 서비스 공통 부분과 프로토콜에 특화된 부분을 나누어 기술하는 구조를 가지고 있다.

본 논문에서 제안하는 M2M 플랫폼과 웹 애플리케이션 간의 통합 방안은 추상적으로 정의되어 있는 M2M 플랫폼 서비스 인터페이스를 구체적인 형태의 WSDL 2.0 문서로 기술하는 방법과 WSDL 2.0 문서를 토대로 SOAP/HTTP/CoAP의 웹 클라이언트 및 서비스 요청 메시지를 생성하는 방법 그리고 M2M 플랫폼과 웹 애플리케이션 간의 상호 프로토콜 변환 방법을 포함하는 전체적인 통합 구조를 제시하고 있다.

본 논문은 2장에서는 ETSI 표준 M2M 플랫폼과 웹 애플리케이션 간의 통합 구조를, 3장과 4장에서는 통합 구조에 대한 상세한 설계 및 이에 대한 구현을 소개하며, 마지막 5장 결론으로 구성되어 있다.

II. ETSI M2M 플랫폼과 이종의 웹 애플리케이션 간의 통합 구조

ETSI에서 정의하고 있는 Service Capability Layer (SCL)는 M2M 플랫폼에 대응되는 용어로, 위치에 따라 네트워크 SCL 및 디바이스/게이트웨이 SCL로 존재하며, 단말인증, 원격제어, 데이터 수집, 서비스 관리 등 M2M 네트워크 및 서비스 관리를 위한 공통 인프라 기능을 제공하는 역할을 한다. SCL에서 M2M 서비스에 필요한 공통 인프라 기능을 구현하여 제공하기 때문에, M2M 버티컬 서비스 제공자는 중복 구현 없이 버티컬 서비스에 필요한 로직만 작성하면 된다. SCL에서 제공되는 플랫폼 서비스는 REST 방식의 인터페이스 규격에 따라 접근 및 제어가 가능하다[11].

그림 2. ETSI M2M 플랫폼과 이종의 웹 애플리케이션[11]
Fig. 2 Integrated structure of ETSI M2M platform and heterogeneous web applications[11]

ETSI M2M 아키텍처 상에는 두 종류의 애플리케이션이 있다. 첫 번째 네트워크 애플리케이션(Network Application, NA)은 M2M 네트워크 플랫폼으로부터 데이터를 받아와 서비스를 제공하는 애플리케이션으로, 사용자에게 특정 도메인에 대한 서비스 제공자 역할을 한다. 두 번째 디바이스/게이트웨이 애플리케이션 (Device Application, DA/Gateway Application, GA)은 발생한 데이터를 M2M 디바이스/게이트웨이 플랫폼으로 전송하는 애플리케이션으로, 센서 등의 원시 데이터를 생성하고 전달하는 데이터 제공자 역할을 한다. 서빙 서비스

제공을 위한 SOAP, 디바이스 지원을 위한 CoAP 그리고 양쪽 모두 적용 가능한 HTTP의 이종의 웹 애플리케이션으로 구성된 M2M 솔루션의 구조는 그림 2와 같다.

M2M 플랫폼 서비스와 이종의 웹 애플리케이션 간의 통합을 위해서는 M2M 플랫폼이 다른 웹 애플리케이션의 접근을 지원해야 한다. 본 논문에서는 WSDL 2.0 문서를 이용한 M2M 플랫폼 서비스 인터페이스의 웹 서비스 기술 방법, 웹 클라이언트 코드 생성, 웹 프로토콜 변환 방안을 제시하며, 이는 각각 M2M 플랫폼 접근을 위한 웹 애플리케이션의 설계-구현-실행 단계에서 적용된다. 그림 3은 ETSI M2M 플랫폼과 이종의 웹 애플리케이션 간의 통합 과정을 도식화 하고 있다.

그림 3. ETSI M2M 플랫폼과 웹 애플리케이션의 통합 과정
Fig. 3 Process of integration between ETSI M2M platform and web applications

M2M 플랫폼 서비스 접근을 위한 설계 단계에서는 웹 서비스에 대한 기술이 필요하다. M2M 애플리케이션은 M2M 플랫폼과 표준화된 인터페이스 (m1a, d1a[12])를 통해 서로 데이터를 주고받는다. 그러나 M2M 인터페이스 표준은 추상적으로 기술되어 있기 때문에 M2M 표준 문서를 기반으로 웹 애플리케이션의 개발이 용이하지 않다. 반면에 WSDL 2.0 문서는 웹 클라이언트의 자동 생성이 가능하도록 구체적인 웹 서비스를 기술이 가능하기 때문에 개발이 용이하다. 본 논문의 3장 1절에서는 추상적인 M2M 플랫폼 서비스의 표준 인터페이스를 구체적인 웹 서비스를 기술하는 WSDL 2.0 문서

의 생성 방안을 제시하고 있다.

구현 단계에서는 앞서 작성된 WSDL 2.0 문서를 토대로 M2M 플랫폼 서비스에 접근하기 위한 HTTP/SOAP/CoAP 웹 클라이언트 및 XML 형태의 웹 서비스 요청 메시지에 대한 생성이 필요하며, 3장 2절에서는 이를 위한 웹 클라이언트 생성기를 제시하고 있다.

실행 단계에서는 M2M 플랫폼과 이종의 웹 애플리케이션 간의 네트워크 통신을 위한 프로토콜 변환이 필요하다. 3장 3절에서는 프록시를 이용하여 상호 프로토콜을 변환하는 웹 프로토콜 변환기를 제시하고 있다.

본 논문에서는 M2M 상에서 M2M 플랫폼 서비스와 이종의 웹 애플리케이션 간의 상호 데이터 전달에 초점을 맞추기 위하여 ETSI M2M 자원 중 데이터 전달과 관련된 핵심 자원인 SCL, Application, Container, Content-Instance 자원으로 지원 범위를 한정하였다.

III. ETSI M2M 플랫폼과 이종의 웹 애플리케이션 간의 통합 구조 설계

3.1. ETSI M2M 표준 인터페이스 기반 WSDL 2.0 문서 생성

ETSI M2M에서는 여러 M2M 플랫폼 서비스 간에 동일한 접근 방법을 제공하기 위해 M2M 플랫폼/애플리케이션 간의 통신 인터페이스 표준 문서[12]를 제공하고 있다. 이 문서는 M2M 플랫폼 서비스에서 필요한 자원을 정의하고, 자원을 처리하기 위하여 REST[13] 기반의 4가지 메소드(Create, Update, Retrieve, Delete)를 정의하여 제공하고 있다. 예를 들어 M2M 애플리케이션 정보를 저장하는 <application> 자원은 내부 데이터 접근 주소와 애플리케이션 고유 ID 등의 정보를 포함하며, 이 자원의 처리를 위해 자원을 생성하는 applicationCreate, 자원을 검색하는 applicationRetrieve, 자원을 갱신하는 applicationUpdate, 자원을 삭제하는 applicationDelete 인터페이스를 제공한다.

M2M 자원 접근을 위하여 M2M 인터페이스 표준 문서에는 응답/요청 메시지의 종류를 나타내는 'primitiveType', 메시지 전달 주소인 'targetID', 메시지 전송 주체인 'requestingEntity', 응답 메시지의 응답 코드인 'statusCode', 생성/수정된 자원의 URI 정보인 'resourceURI' 그리고 M2M 자원에 대한 기술을 정의하고 있다. M2M 자원 정보는 메시지의 Body에 포함되지만, 나머

지는 헤더 정보에 표기된다. HTTP 프로토콜을 이용할 시, 각 구성 요소는 대부분 표 1과 같이 매핑된다[12].

표 1. M2M 인터페이스 표준 구성요소와 HTTP 매핑
Table. 1 Mapping between M2M interface specification and HTTP

구성 요소	설 명
primitiveType	(Header) HTTP method (POST, GET, PUT, DELETE)
requestingEntity	(Header) Authorization:
targetID	(Header) Host:
resourceURI	(Header) Location:
statusCode	(Header) HTTP Status Code
<resource>	HTTP Body

표 2는 <application> 자원 생성 요청을 위한 인터페이스 규격이고, 그림 4는 표 1을 참조하여 HTTP Request 메시지로 매핑한 예시이다.

표 2. <application> 자원 생성 요청 규격

Table. 2 Request specification of <application> resource creation

applicationCreateRequestIndication	
구성 요소	설 명
requestingEntity	요청자의 식별자
targetID	메시지 전달 할 주소
primitiveType	메시지 종류
<application> 자원	등록 할 자원에 대한 표현

<pre> POST /scls/scl1/applications HTTP/1.1 Host: nscls.example.com Authorization: Basic QWxhZGRpbjpvchsegHscGVulHNlc Content-Type: application/xml Content-Size: 248 <?xml version="1.0" encoding="UTF-8"?> <application xmlns:tns="http://uri.etsi.org/m2m" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <expirationTime>2011-10-27T18:00:00</expirationTime> </application> </pre>	<p>primitiveType</p> <p>targetID</p> <p>requestingEntity (Base64 Encoding)</p> <p><application> resource</p>
--	--

그림 4. <application> 자원 생성 요청을 위한 메시지 예시
Fig. 4 Example of request message for <application> resource creation

표 3은 <application> 자원 생성 요청 후 <application> 자원이 성공적으로 생성되었을 때의 응답에 대한 인터페이스 규격이고 그림 5는 이를 HTTP Response 메시지로 매핑 하였을 때의 예시이다.

표 3. <application> 자원 생성에 대한 응답 규칙

Table. 3 Response specification of <application> resource creation

applicationCreateResponseConfirm (successful case)	
구성 요소	설 명
primitiveType	메시지 종류
statusCode	응답 코드
resourceURI	생성된 <application> URI
<application> 자원	등록 된 자원에 대한 표현

그림 5. <application> 자원 생성 응답 메시지 예시

Fig. 5 An example of response message for <application> resource creation

M2M 인터페이스 표준 문서에서는 HTTP 프로토콜에 대한 매핑 방안을 제시하고 있으며, 이는 WSDL 2.0 문서로 표현 할 수 있다. 그림 6은 그림 4의 HTTP Request 메시지에 대한 WSDL 2.0 표현이다.

그림 6. <application> 자원 생성 M2M 인터페이스의 WSDL 2.0 표현(HTTP)

Fig. 6 WSDL 2.0 representation of M2M interface for <application> resource creation (HTTP)

M2M 플랫폼 서비스 요청 인터페이스에서 정의한 구성요소를 WSDL 2.0 형식으로 표현하면 표 4와 같이 표현 할 수 있다. 또한 WSDL 2.0은 HTTP 웹 서비스 외에도 SOAP 웹 서비스도 표현할 수 있기 때문에,

HTTP 웹 서비스의 WSDL 2.0 형식 기술은 표 5와 같이 SOAP 웹 서비스 형식으로 표현할 수 있다.

표 4. M2M 인터페이스-WSDL 2.0 (HTTP) 간의 매핑 규칙

Table. 4 Mapping rules for M2M interface-WSDL 2.0 (HTTP)

ETSI M2M	WSDL 2.0
primitive Type	<operation ref="{RequestIndication 식별자}" whhttp:method="{ 매칭되는 HTTP 메소드}" />
requesting Entity	<endpoint whhttp: authenticationScheme="basic" />
targetID	<operation whhttp:location="{local path}" /> <endpoint address="{service_address}" />

표 5. M2M 인터페이스-WSDL 2.0 (SOAP) 간의 매핑 규칙

Table. 5 Mapping rules for M2M interface-WSDL 2.0 (SOAP)

ETSI M2M	WSDL 2.0
primitive Type	<operation ref="{RequestIndication 식별자}" />
requesting Entity	SOAP Header의 <Authheader>에서 정의
targetID	<operation wsoap:action="{ 매칭되는 서비스 메소드}" /> <endpoint address="{service_address}" />

표 5에 정의된 매핑 규칙에 따라 SOAP 웹 서비스 형태로 기술된 <application> 자원 생성 인터페이스에 대한 WSDL 2.0 표현은 그림 7과 같다.

그림 7. <application> 자원 생성 M2M 인터페이스의 WSDL 2.0 표현 (SOAP)

Fig. 7 WSDL 2.0 representation of M2M interface for <application> resource creation (SOAP)

본래 WSDL 2.0에서는 CoAP 웹 서비스 기술을 지원하지 않지만 CoAP은 저사양 기기의 REST 지원을 위해 HTTP와 유사하게 설계되었기 때문에 WSDL 2.0의 HTTP 웹 서비스 기술로 CoAP 메시지 생성이 가능하다. 그림 8은 그림 5의 WSDL 2.0을 기반으로 <application> 자원 생성 요청을 주고받는 CoAP 메시지의 예시이다. (실제 CoAP 헤더는 bit 단위로 인코딩되어 전송된다.)

그림 8. <application> 자원 생성을 위한 CoAP 프로토콜 요청/응답 메시지 예시

Fig. 8 An example of CoAP protocol request/response messages for <application> resource creation

3.2. WSDL를 이용한 웹 클라이언트 생성기

웹 클라이언트 생성기(Generator)는 WSDL 2.0 문서의 내용을 토대로 타겟 프로토콜인 SOAP, HTTP, CoAP 형식을 따른 웹 클라이언트 및 웹 서비스 요청 메시지를 생성하는 역할을 한다.

그림 9. 웹 클라이언트 생성기를 통한 웹 클라이언트 및 M2M 요청 메시지 생성

Fig. 9 Generation of web application client and M2M request message using web client generator

웹 클라이언트 생성기는 M2M 애플리케이션과 SCL 간의 통신을 다루고 있으므로, 표 6과 같이 해당 구간의 데이터 전달과 관련된 자원 제어를 위한 인터페이스 [12]에 대한 실행 코드를 생성한다.

표 6. 웹 클라이언트 생성기 지원 M2M 인터페이스
Table. 6 M2M interfaces supported by web client generator

관련 자원	지원하는 인터페이스
SCLs	sclsRetrieve, sclsUpdate
SCL	sclRetrieve, sclUpdate, sclDelete
Applications	applicationsRetrieve, applicationsUpdate
Application	applicationCreate, applicationRetrieve, applicationUpdate, applicationDelete
ApplicationAnnc	applicationAnncRetrieve
Containers	containersRetrieve, containersUpdate
Container	containerCreate, containerRetrieve, containerUpdate, containerDelete
ContainerAnnc	containerAnncRetrieve
ContentInstances	contentInstancesRetrieve
ContentInstance	contentInstanceCreate, contentInstanceRetrieve, contentInstanceDelete

웹 클라이언트 생성기에서 생성하는 M2M 플랫폼으로의 요청 메시지는 XML 포맷을 지원하며, 기본 값으로 각 자원에서 필수로 요청되는 요소 및 속성들을 포함하고 있다. 동일한 상황에서 요청 메시지는 프로토콜과 상관없이 모두 동일하다. 웹 클라이언트는 Java 기반으로 작성되었다. 그림 10~12는 웹 서비스 생성기를 통해 생성된 <application> 갱신(Update) 요청을 처리하는 Java 기반의 SOAP (Apache Axis2 기반) / HTTP (Apache HttpComponents 기반) / CoAP (Californium 기반) 클라이언트의 예시 코드를 보여주고 있다.

그림 10. SOAP 클라이언트 예시
Fig. 10 An example of SOAP client

```

HTTP Client (Apache HttpComponents)

String serviceURL = "http://nsl.example.com/scs/sc1/applications/app1";

/* Update expirationTime of application resource */
String requestMsg = "<?xml version='1.0' encoding='UTF-8'?>
<application xmlns:tns='http://uri.etsi.org/m2m'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' apid='app1'>
<expirationTime>2013-06-30T18:00:00</expirationTime>
</application>";

PutMethod req = new PutMethod(serviceURL);
req.setRequestEntity(new StringRequestEntity(requestMsg));
HttpClient httpClient = new HttpClient();
...
int result = httpClient.executeMethod(req);
System.out.println(req.getResponseBodyAsString());
    
```

그림 11. HTTP 클라이언트 예제
Fig. 11 An example of HTTP client

```

CoAP Client (Californium)

URI serviceURL = new URI "CoAP://nsl.example.com/scs/sc1/applications/app1";

/* Update expirationTime of application resource */
String requestMsg = "<?xml version='1.0' encoding='UTF-8'?>
<application xmlns:tns='http://uri.etsi.org/m2m'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' apid='app1'>
<expirationTime>2013-06-30T18:00:00</expirationTime>
</application>";

Request request = new PUTRequest();
request.setURI(serviceURL);
request.setPayload(requestMsg);
...
request.execute();
Response response = request.receiveResponse();
System.out.println(response.getPayloadString());
    
```

그림 12. CoAP 클라이언트 예제
Fig. 12 An example of CoAP client

3.3. 웹 프로토콜 변환기

HTTP로 구현된 ETSI M2M 플랫폼으로는 직접 다른 프로토콜의 웹 서비스 접근이 허용되지 않는다. 예를 들어 SOAP 웹 클라이언트는 HTTP 기반 M2M 플랫폼 서비스에 바로 접근하지 못한다. 이러한 경우에는 웹 프로토콜의 변환 과정을 통하여 상호 접근이 가능하다.

그림 13. 웹 프로토콜 변환기를 이용한 통합 구조
Fig. 13 Integration structure using web protocol translator (proxy)

본 논문에서는 그림 13과 같이 이중의 웹 애플리케이션 간에 프록시(Proxy) 형태의 프로토콜 변환기를 배치 함으로 상호 프로토콜 간 변환 메커니즘을 제공하였다. 만약 M2M 네트워크 플랫폼(NSCL)에 접근하는 네트워크 애플리케이션(NA)이 SOAP 프로토콜을 사용한다면, 이 NA는 SOAP-HTTP 프록시에게 요청을 대신 NSCL에 전달할 것을 요청한다. SOAP-HTTP 프록시는 이 요청을 HTTP로 변환하여 NSCL로 보내고, 응답이 오면 응답 메시지를 SOAP 형식으로 변환하여 다시 NA로 전달한다. NA는 특정 도메인에 대한 서비스를 제공하는 버티컬 서비스로, M2M 네트워크 플랫폼(NSCL)에게는 웹 클라이언트 역할을 하지만 실제 서비스 소비자에게는 웹 서비스를 제공하는 서버 역할을 한다.

만약 디바이스 애플리케이션(DA)이 CoAP 프로토콜을 사용하고, M2M 게이트웨이 플랫폼(GSCl)은 HTTP 프로토콜만 허용한다면, CoAP-HTTP 프록시를 통하여 상호 프로토콜 변환이 가능하다. 이 변환 과정은 CoAP 프로토콜 표준에서 정하는 규칙을 따른다[8][14]. CoAP 프로토콜을 사용하는 DA가 GSCl에 요청 메시지를 보낼 때에 CoAP-HTTP 프록시는 CoAP 프로토콜을 HTTP 프로토콜로 변환하고, 응답 시에는 HTTP 프로토콜을 CoAP 프로토콜로 변환하여 준다.

프록시를 사용하는 경우에는 프록시 주소와 서비스를 요청하고자 하는 M2M 플랫폼 주소의 두 가지 주소가 필요하다. WSDL 2.0 표준은 서비스 연결점(주소)을 정의하는 <endpoint> 요소에 필요 시 네임스페이스가 명시된 속성 정의의 추가를 허용하고 있다[10]. 따라서 프록시 정보를 위하여 WSDL 2.0 문서상에 네임스페이스 및 프록시 주소를 포함하는 속성 정의를 추가하였다. SOAP 프로토콜에 대한 프록시 주소는 'wsproxy:soapProxy' 속성으로 정의하고, CoAP 프로토콜에 대한 프록시 주소는 'wsproxy:coapProxy' 속성으로 정의하였다. 그림 14는 프록시 정의에 대한 WSDL 2.0 문서 표현이다.

```

<description ... xmlns:wsproxy="http://webservice.proxy.com/proxy">
<!-- proxy for SOAP web service -->
<endpoint ... wsproxy:soapProxy="{proxy_address}" > ... </endpoint>

<!-- proxy for CoAP web service -->
<endpoint ... wsproxy:coapProxy="{proxy_address}" > ... </endpoint>
    
```

그림 14. WSDL 2.0 문서에서의 프록시 정의
Table. 14 Proxy definition of WSDL 2.0 document

IV. ETSI M2M 플랫폼과 이종의 웹 애플리케이션 간의 통합 구조 구현

앞서 제시한 ETSI M2M 플랫폼과 이종의 웹 애플리케이션 간의 통합 구조 설계를 검증하기 위하여 스마트 미터링 시나리오를 기반으로 SOAP/CoAP 웹 애플리케이션과 HTTP 기반의 M2M 플랫폼 구축하여 상호 연동 시스템을 구현하였다. 스마트 미터링 시스템에서는 냉장고, 세탁기, 에어컨과 같은 각 가전 기기가 사용하는 전력량 데이터를 M2M 플랫폼으로 전송하고, 스마트 미터링 서비스는 M2M 플랫폼으로부터 전력량 데이터를 전달 받아 소비자에게 종합적인 전력 소비 정보를 제공한다. 본 시스템에서 가전 기기는 CoAP 웹 서비스로 구축하였으며, 스마트 미터링 서비스는 SOAP 웹 서비스로 구축하였다. 다만, 현재 CoAP을 지원하는 가전 기기는 널리 보급되고 있지 않기 때문에 가전 기기의 구현은 에뮬레이터로 대체하였다. 스마트 미터링 시스템의 구성도는 그림 15와 같다.

그림 15. 스마트 미터링 시스템 구조도
Fig. 15 Structure of smart metering system

스마트 미터링 시스템의 M2M 서비스 구축을 위하여 M2M 플랫폼에는 필수 서비스인 SCL과 Application을 연결하는 Network/Gateway Application Enablement (N/GAE), 서로 다른 SCL을 연결하는 Network/Gateway Generic Communication(N/GGC), 자원의 등록 및 정보를 관리하는 Network/Gateway Reachability, Addressing and Repository(N/GRAR) 그리고 네트워크 등의 설정을 관리하는 Network/Gateway Remote Entity Management (N/GREM)이 구현되어 있으며, 이들은 서비스 종류에 따라 각각 외부의 SCL 및 애플리케이션과 연결되어 있다. 본 구현에서 HTTP 서버는 Apache Tomcat을, SOAP 서버는 Apache Axis2를, 그리고 CoAP 서버는 Californi-

um을 이용하여 구축하였다. 스마트 미터링 시스템의 내부 구현 설계도는 그림 16과 같다.

그림 16. 스마트 미터링 시스템 내부 구현 설계도
Fig. 16 Implementation of smart metering system

스마트 미터링 시스템에서 제공하는 웹 서비스 클라이언트의 UI는 그림 17과 같다.

그림 17. 스마트 미터링 웹 서비스 UI
Fig. 17 Smart metering web service UI

본 시스템 구축을 통하여 앞서 제시한 WSDL 2.0을 이용한 ETSI 표준 M2M 플랫폼 서비스와 이종의 웹 애플리케이션 간의 상호 연동을 확인할 수 있었다.

V. 결 론

본 논문에서는 ETSI 표준 M2M 플랫폼에서 제공하는 서비스의 인터페이스를 WSDL 2.0 표준 문서로 기술하는 방법을 제공하고, 작성된 WSDL 2.0 문서로 웹 클라이언트 및 요청 메시지를 생성해주는 웹 클라이언트 생성기 및 서로 다른 웹 프로토콜을 변환하는 웹 프로토콜 변환기를 개발하여, ETSI 표준 M2M 플랫폼 서비스를 SOAP, HTTP 및 CoAP 형태의 웹 서비스로 제공하도록

하였다. 이를 통하여 ETSI 표준 M2M 플랫폼이 SOAP, HTTP 및 CoAP 형태의 이중의 웹 애플리케이션과 연동이 가능한 기반을 마련함으로써 M2M 활용 범위를 넓힐 수 있었으며, 또한 추상적인 M2M 인터페이스 표준을 WSDL 2.0 문서로 구체화하고 웹 클라이언트를 생성하여 줌으로써 개발자로 하여금 용이한 M2M 애플리케이션 개발을 가능하게 하였다.

감사의 글

본 연구는 2011년도 한국연구재단 국제기관간 MOU지원사업(2011-0030918)과 2012년도 연구개발특구진흥재단 연구소기업전략육성사업(A2012DD006)의 지원에 의하여 이루어진 연구입니다.

REFERENCES

- [1] K.Kim, "M2M Communication Module and Machine Platform for M2M Vertical", *TTA M2M Standard · Technology · Service Seminar*, 2011.
- [2] M.Chen, J.Wan and F.Li, "Machine-to-Machine Communications: Architectures, Standards and Applications", *KSII Transactions on Internet and Information Systems*, vol. 6, no. 2, 2012.
- [3] E.Scarrone, "ETSI Standard are enabling global M2M solution", *3rd M2M Workshop*, 2012.
- [4] Korea Communications Agency, "Current State of oneM2M Establishment and Standardization", *Korea Communications Agency Trend&Insight*, 2012.
- [5] ETSI TC M2M, "Machine to Machine Communications", *ETSI TC M2M Overview*, 2011.
- [6] S.Jung, W.Chung, "M2M Technology based Global Healthcare Platform", *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 14 Issue 11, 2010.
- [7] Y.Park, H.Wu, H.Paek, D.Min, "Adaption and Assessment of ETSI M2M Standard in Smart Home Environments", *Journal of Korea Society of IT Services*, Vol. 11 Issue 3, 2012.
- [8] IETF Std. CoRE WG Draft-ietf-core-coap-17, *Constrained Application Protocol (CoAP)*, IETF, 2013.
- [9] K.Min, Y.Kim, S.Yoo, H.Kim, H.Jung, "Implementation of CoAP Protocol for USN Environment", *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 15 Issue 5, 2011.
- [10] W3C Std. *Web Services Description Language (WSDL) Version 2.0 Part1: Core Language*, W3C, 2007.
- [11] ETSI Std. M2M TS 102 690 V1.1.1, *Machine-to-Machine communications (M2M); Functional architecture*, 2011.
- [12] ETSI Std. M2M TS 102 921 V1.1.1, *Machine-to-Machine communications (M2M); mla, dla and mld interfaces*, 2012.
- [13] R.T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", Ph. D. dissertation, University of California Irvine, Irvine, CA, 2000.
- [14] IETF Std. CoRE WG Draft-ietf-core-http-mapping-00, *Best Practices for HTTP-CoAP Mapping Implementation*, 2013.

박윤정(Yunjung Park)

건국대학교 컴퓨터정보통신공학과 박사과정

※관심분야 : 사이버-물리 시스템, 사물통신, 분산 미들웨어 및 서비스 융합

민덕기 (Dugki Min)

건국대학교 컴퓨터공학부 교수, 공학박사

※관심분야 : 분산/병렬 시스템, 사이버-물리 시스템, 빅 데이터, 클라우드 컴퓨팅, 분산 미들웨어