JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 18, No. 10 : 2473~2481 Oct. 2014

REST 웹서비스 Open API를 사용한 분산처리 기법 연구

최 민*

Research of Distributed Computing with REST Open API Web Services

Choi, Min*

Department of Communication and Information Engineering, Chungbuk University, Cheongju, 362-763, Korea

요 약

차세대 정보시스템의 대부분은 스마트폰 등의 휴대용 모바일 기기에서 동작할 것이다. REST 웹서비스는 모바일 애플리케이션 개발 분야에서 매우 급속하게 대중화되고 있다. 이러한 REST 웹서비스의 사용은 단지 활용도가 증가하는 것 뿐 아니라, 모바일 애플리케이션을 개발하는 방법론 자체를 바꾸고 있다. 이는 REST 웹서비스가 제3의 애플리케이션 개발에 있어 빌딩블록과 같은 형태로 사용될 수 있기 때문이다. 본 연구에서는 소켓을 대신하여 REST 웹서비스 Open API를 사용한 분산 컴퓨팅을 제안한다. 이를 위해서, 본 연구에서는 REST 웹서비스 Open API가 기존의 소켓 기반 서비스에 비해 충분한 성능을 제공함을 분석적/실험적 방법을 통해 제시한다. 따라서, 향후 인터넷 통신 프리미티브 API로서 REST 웹서비스 Open API가 충분한 성능을 갖는다는 사실을 보여준다.

ABSTRACT

The majority of next generation information systems will be working on portable mobile devices such as smartphones. REST Open API web services have quickly become popular among mobile application development. The use of REST web services are not only growing in popularity but totally changing the way of mobile applications development. This is because REST web services could be used in application development as a form of building blocks, which is completely independent, compatible, to any platforms. With this strength, REST web services encourage third party application to build add-on functionality. This research proposes the use of REST web service to replace the use of socket APIs into major internet communication APIs. To this end, this paper also provide performance evaluation of the REST web services compared to the conventional socket APIs, focusing on scalability by analytical and experimental evaluations. Finally, we summarize the possibility whether REST Open API web services have enough performance as future major internet communication primitive APIs..

키워드 : REST 웹서비스, REST Open API, 분산처리

Key word : REST Web Service, REST Open API, Distributed Processing

접수일자 : 2014. 07. 18 심사완료일자 : 2014. 08. 20 게재확정일자 : 2014. 09. 02

* **Corresponding Author** Min Choi(E-mail:miin.chae@gmail.com, Tel:+82-43-261-3367)
Department of Communication and Information Engineering, Chungbuk National University, Cheongju 362-763, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2014.18.10.2473 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

최근 스마트폰/태블릿 등 스마트 기기의 활용이 급속하게 증가하고 있으며, 스마트폰/태블릿 기기들은 안드로이드/아이폰/윈도우폰을 비롯하여 매우 다양한 운영체제가 활용되고 있다. 이러한 현상은 디바이스 호환성의 저하를 유발하는 프래그멘테이션(device fragmentation)을 유발한다. 현재까지는 특정한 스마트폰 기기 플랫폼에 종속적인 애플리케이션을 각 플랫폼 고유의(native) API를 활용하여 개발하고 있다. 하지만, 머지않아 HTML5 기술과 REST 웹서비스 Open API기술[14]을 결합한 플랫폼 독립적인 모바일 애플리케이션을 통합앱스토어(WAC)를 통하여 제공하는 방법이 더욱 보편화될 것으로 예측한다. 즉, 보다 다양한 환경에서 동작하는 모바일 애플리케이션 개발에 공공/민간의 데이터베이스(DB)를 제공하고 편리하게 접근할 수 있도록 하기 위해서 REST 웹서비스 Open API 활용이 증가하고 있다. 이와 같이, REST 웹서비스 Open API 기술은 앞으로 모바일 클라우드 서비스의 실현을 위하여 주목할만한 기술중의 하나이다. 이와 같이, 웹서비스(web service) Open API를 이용한 스마트폰에서의 분산컴퓨팅 구현이 각광받고 있다. 웹서비스는 W3C에서 정의한 WSDL(Web Service Description Language)를 사용하여 웹서비스를 기술하며, UDDI(Universally Distibuted Directory Interface)를 사용하여 검색하고 SOAP 프로토콜을 활용하여 통신한다. 그러나, 이와 같은 기존의 웹서비스 기술 및 사용방법은 그 구성이 복잡하고 오버헤드가 큰 이유로 널리 활용되지 못하였다. 최근에는 스마트폰이 대중화 되면서 이와 같은 SOAP 방식 보다는 REST 방식의 웹서비스 활용이 더욱 확산되는 추세다. REST 웹서비스는 Roy Fielding에 의해 소개[17]된 것으로, 스마트폰의 활용 증대와 함께 REST 웹서비스의 사용이 점점 보편화되고 있다. 이에 따라, 오늘날의 많은 인터넷 회사들이 이미 SOAP 기반 웹서비스와 REST 기반 웹서비스로 동시에 병행하여 서비스를 제공하고 있다. SOAP 방식 웹서비스에 대해서는 충분한 연구가 진행되었으며, 기계가 이해할 수 있도록 엄격한 규약과 인터페이스를 정의하도록 설계되었다. 하지만, REST 방식의 웹서비스는 현재까지도 충분히 연구되지 않았다. REST 웹서비스는 사람이 사용하기 편리하도록 설계된 기술이고, 세부 인터페이스에 대해서는 표준에서 관여하지 않는다. 그럼에도 불구하고, 최근 스마트폰의 급속한 보급과 함께, REST 웹서비스 Open API의 활용이 빠르게 증가하고 있다. REST 웹서비스는 스마트폰 애플리케이션 개발에 있어 핵심기술 중의 하나다. 실제로, REST 웹서비스는 스마트폰에서 인터넷을 통해 DB 정보를 접근하기 위한 편리한 방법이며, 최근 스마트폰 애플리케이션은 공개된 Open API를 활용하여 다양한 공공/민간 DB로부터 REST 웹서비스를 활용하여 데이터를 접근하고 있다.스마트폰 애플리케이션 개발에 있어서는 소켓을 이용한 모바일-클라우드(클라이언트-서버) 통신방법을 많이 사용하고 있다. 그러나, 이러한 방법은 임시변통적(add-hoc)이며 체계적이지 않다. 이는 소켓이 동작하는 TCP/IP 프로토콜 스택의 체계를 논하는 것이 아니라, 소켓을 활용하는 애플리케이션 레이어(layer)의 통신 프로토콜이 체계적이지 않다는 것이다. 즉, 스마트폰 애플리케이션 제작자에 따라 필요에 따라 자신들의 프로토콜을 만들어 사용하므로 임시변통적으로 구현되는 경우가 많다. 따라서, 본 연구에서는 이러한 소켓통신 방식을 대체하여 분산컴퓨팅을 실현하는 방법으로 REST 웹서비스 기반의 Open API를 활용하는 것을 제안한다. 또한, 소켓 통신 방법 대비 REST 웹서비스를 사용하여 분산처리하는 방법을 실험적으로 비교/분석한다. 물론, 참고문헌 [16]에서도 본 논문에서 시도한 바와 같이 REST Open API의 활용 가능성에 대하여 언급하고 있으나, 이는 전통적으로 잘 알려진 웹서비스의 분류 중 REST 방식과 SOAP 방식의 비교이므로, 본 논문에서의 REST 방식과 소켓의 비교를 통한 연구와는 차이점이 있다.

Ⅱ. 관련연구

최근 스마트폰/태블릿 등 스마트 기기의 활용이 급속하게 증가하고 있으며, 스마트폰/스마트태블릿 기기들은 안드로이드 및 아이폰의 보급과 함께 널리 사용되고 있는 추세다. 따라서, 최근 3~4년 동안 스마트 디바이스 시장은 최근 전세계적으로 급격히 성장하고 있다. 일반 휴대폰 시장 연평균 증가율(CAGR)은 약 3%인데 반해 스마트폰의 CAGR은 2016년까지 19%로 전망된다[1, 2]. 전세계 모바일 SNS 이용자 수는 이미 2008년 2억명 돌파하였고, 2013년 10억명에 달할 것으로 예상하

고 있다. 심비안, 윈도우 등의 모바일 운영체제 시장 점유율 감소와 애플 아이폰과 안드로이드의 지속적인 증가 추세이며, Gartner는 2013년 스마트폰이 사실상 PC를 대체하고 주요 인터넷 디바이스가 될 것이라 전망한 바 있다[3]. W3C[4]를 중심으로 모바일 웹 플랫폼을 위한 Device API 표준화가 진행되고 있다. 2008년 12월에 Device API와 관련되는 다양한 표준화 이슈를 제기한 바 있고, 2009년 6월에 DAP Work Group을 발족하였다. W3C는 이미 HTML5의 1차 표준화를 마치고 Device API 표준안을 공개한 바 있다. 세부적으로 W3C 표준안에 적용하기 위해서 노력하는 단체들은 여러곳이 있으나 JIL(joint inovation lab) [5]과 OMTP BONDI [6]가 대표적이다. 최근 정보시스템 응용에서 웹서비스의 사용이 점점 확대되고 있다. 이에 따라, 오늘날의 많은 인터넷 회사들이 이미 SOAP 기반 웹서비스와 REST 기반 웹서비스로 동시에 병행하여 서비스를 제공하고 있다[7, 8]. SOAP 방식 웹서비스에 대해서는 충분한 연구가 진행되었으며, 기계가 이해할 수 있도록 엄격한 규약과 인터페이스를 정의하도록 설계되었다[9, 10]. 하지만, REST 방식의 웹서비스는 현재까지도 충분히 연구되지 않았다. REST 웹서비스는 사람이 사용하기 편리하도록 설계된 기술이고, 세부 인터페이스에 대해서는 표준에서 관여하지 않는다. 그럼에도 불구하고, 최근 스마트폰의 급속한 보급과 함께, REST 웹서비스 Open API의 활용이 빠르게 증가하고 있다[11-13]. REST 웹서비스는 스마트폰 애플리케이션 개발에 있어 핵심기술 중의 하나다.

기관명	서비스명	서비스 유형	주요내용 (간략 소개)	URL
경기도	기반정보 관리 서비스	REST	일일 전체 노선 및 정류소, 운행지역에 관 한 기반정보를제공한다. (기반정보는 텍스트 파일로 제공)	URL 바로가기
	기반정보 관리 서비스	SOAP	일일 전체 노선 및 정류소, 운행지역에 관한 기반정보를 제공한다. (기반정보는 텍스트 파일로 제공)	URL 바로가기
	버스 도착 정보 조회 서비스	REST	해당 정류소의 특정 노선에 대한 첫번째/두번째 도착예정 차량의 위치정보와 도착예정시간을 제공한다.	URL 바로가기
	버스 위치 정보 조회 서비스	REST	해당 노선의 운행차량에 대한 현재 위치(정류소)와 막차, 저상버스 여부 등에 대한 실시간 운행정보를 제공한다.	URL 바로가기
	버스도착정보 조회 서비스	SOAP	해당 정류소의 특정 노선에 대한 첫번째/두번째 도착예정 차량의 위치정보와 도착예정시간을 제공한다.	URL 바로가기
	버스위치정보 조회 서비스	SOAP	해당 노선의 운행차량에 대한 현재 위치(정류소)와 막차, 저상버스 여부 등에 대한 실시간 운행정보를 제공한다.	URL 바로가기
국민권익위원회	공개제안조회서비스	SOAP	국민신문고의 공개제안목록 및 상세항목을 조회하는 서비스입니다.	URL 바로가기
	공모제안조회서비스	SOAP	공모제안조회서비스	URL 바로가기
	나의민원서비스	SOAP	나의민원서비스	URL 바로가기
	설문조사조회서비스	SOAP	설문조사조회서비스	URL 바로가기
	우수제안조회서비스	SOAP	우수제안조회서비스	URL 바로가기
	전자공청회 조회서비스	SOAP	전자공청회조회서비스	URL 바로가기
	정책포럼 조회서비스	SOAP	정책포럼조회서비스	URL 바로가기

그림 1. 스마트모바일앱개발지원센터(SMAC)[8] 제공 공공정보 리스트

Fig. 1 Public information list provided by Smart Mobile App Center(SMAC)[8]

실제로, REST 웹서비스는 스마트폰에서 인터넷을 통해 DB 정보를 접근하기 위한 편리한 방법이며, 최근 스마트폰 애플리케이션은 공개된 Open API를 활용하여 다양한 공공/민간 DB로부터 REST 웹서비스를 활용하여 데이터를 접근하고 있다. 스마트폰 애플리케이션 개발에 있어서는 소켓을 이용한 모바일-클라우드(클라이언트-서버) 통신방법을 많이 사용하고 있다. 그러나, 이러한 방법은 임시변통적이며 체계적이지 않다. 이는 소켓이 동작하는 TCP/IP 프로토콜 스택의 체계를 논하는 것이 아니라, 소켓을 활용하는 애플리케이션 레이어의 통신 프로토콜이 체계적이지 않다는 것이다.

Ⅲ. 모바일-클라우드 환경에서 소켓을 대체하는 분산컴퓨팅을 위한 REST웹서비스 Open API

일반적으로, 스마트폰을 활용한 모바일-클라우드 컴퓨팅 환경에서 스마트폰에서 서버와 통신하기 위해서는 대부분 소켓(Socket)을 이용한 통신방식을 활용한다. 그러나, 모바일-클라우드(클라이언트-서버) 컴퓨팅에서 필요로 하는 프로시져 호출 관례는 원격프로시져호출의 형태가 일반적이다. 즉, 모바일 기기가 클라우드 컴퓨팅 플랫폼에 원격지 통신을 수행하는 목적은 대부분 모바일 기기에서 수행하기 어려운 작업을 대신 수행하거나, 방대한 계산을 클라우드 컴퓨팅 플랫폼에서 대신 처리하도록 위임하거나, 혹은 자신의 데이터를 다른 디바이스 혹은 다른 사람과 공유하기 위해서 데이터를 전송하는 경우가 많다.

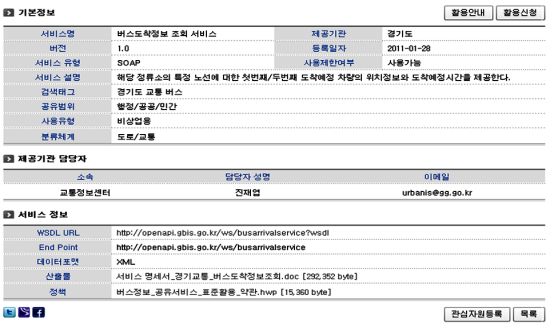

그림 2. SMAC[8] 제공 Open API의 상세정보 페이지

Fig. 2 Detailed information of Open API list by SMAC[8]

1. Request URL
http://openapi.naver.com/search
2. Request parameter
key string (mandatory) : key string for authentication
target string (mandatory) : adult
query string (mandatory) : search keyword as UTF-8 encoding
- Sample URL
http://openapi.naver.com/search?key=test&query=girl&target=adult
3. Response field
adult integer : 0 , 1 (0 – non adult, 1 – adult)

그림 3. 성인 검색어 판별 REST Open API
Fig. 3 REST Open API for adult search keyword checking

따라서, 단순히 소켓통신을 통해서 임의의 프로토콜에 따라서 메시지를 주고받는 것 보다는, 웹서비스를 통해 서버측에 특정한 연산 또는 데이터를 요청하고 그에 대한 수행결과를 전달받는 것이 더욱 필요하다. 결과적으로 본 연구에서는 모바일-클라우드 환경에서 소켓 대체하여 REST 웹서비스 Open API를 사용한다. 물론, REST 웹서비스 Open API는 애초에 이러한 계산 부하위임을 위한 분산컴퓨팅 환경의 구현을 목적으로 제안된 것은 아니다[15]. 대신에, 제3자서버에서 제공하는 기능을 스마트폰, 정보시스템 등에서 플랫폼 독립적으로 사용할 수 있도록 기능을 제공하기 위한 목적으로 도입되었다. 하지만, 본 연구에서는 웹서비스 Open API의 특성을 활용하여 시간이 많이 걸리는 작업을 고성능 컴퓨팅 서버에서 작업할 수 있도록 개인화된 웹서비스를 개발하여 서버에 탑재하고 이를 웹서비스 Open API로 호출하여 사용할 수 있도록 한다.

그림 2는 SMAC[8]에서 제공하는 특정 Open API에 대한 사용방법 상세정보 페이지이다. 여기서는 버스 도착정보 조회서비스에 대한 상세정보를 나타내고 있으며, 보다 자세한 사용방법은 “산출물” 항목에 있는 서비스명세서 문서를 통해 확인할 수 있다. 그림 3의 REST 웹서비스 Open API는 검색 서비스 Open API를 사용하기에 앞서, 해당 검색어가 19세미만 학생들이 열람하기 부적절한 성인검색어인지 여부를 판단하는 Open API 서비스이다. REST 웹서비스 Open API의 semantic은 RPC와 유사하다. REST 웹서비스 실행결과는 일반적으로 XML형태로 제공된다. 기존에는 이러한 XML 데이터를 해석하기 위해서 XML Parser [7]를 통해 데이터를 분석하고 사용자가 원하는 부분을 추출하여 활용하였다. 그러나, 웹서비스 조합을 편리하게 하기 위해서 본 연구에서는 REST 웹서비스를 객체로 변환하는 방법을 소개한다. 여기서 객체는 프로그래밍 프리미티브이며 일반적인 객체지향 프로그래밍 언어에서 활용하는 형태이므로 JAVA 등의 프로그래밍 언어에 익숙한 개발자들이 손쉽게 이해할 수 있다.

오늘날 REST 웹서비스는 스마트폰 애플리케이션 개발에 있어 핵심기술이 되어가고 있다. 왜냐하면, REST 웹서비스야말로 인터넷을 통해 정보를 접근하기 위한 가장 편리한 방법이다. 그리고, 최근 스마트폰 애플리케이션은 여러개의 REST 웹서비스로부터 정보를 얻어오기 때문에, 두 개 이상의 REST 웹서비스를 조합하여 사용함으로써 원하는 기능을 구현할 필요가 많다.

1. Request URL
http://openapi.naver.com/search

2. Request parameter
key string (mandatory) : key string for authentication.
target string (mandatory) : webkr
query string (mandatory) : search keyword as UTF-8 encoding
domain string : site domain for searching
display integer : search result numbers per a page (10 as default, 100 as max)
start integer : search start position (1 as min, 1000 as max)

- Sample URL
http://openapi.naver.com/search?key=test&query=keyword&display=10&start=1&target=webkr

3. Response field
rss : for debugging
channel : container for search results
lastBuildDate datetime : date of result generated
total integer : total number of documents
start integer : start value of search result documents
display integer : number of search result
item : search result including title, link, description
title string : search result document title
link string : hypertext link for search result document.
description string : summary for search result document

그림 4. 웹 DB로부터 검색하는 Open API
Fig. 4 Open API of REST web service for searching from web database

```
4. Error messages
000 : System error
010 : Your query request count is over the limit
011 : Incorrect query request
020 : Unregistered key
021 : Your key is temporary unavailable
100 : Invalid target value
101 : Invalid display value
102 : Invalid start value
110 : Undefined sort value
200 : Reserved
900 : Undefined error occurred
```

그림 5. REST 웹서비스 Open API의 에러 메시지
Fig. 5 Error Messages of REST Web Service Open API

```
try{
      URL text =
            new URL(strURLforOpenAPI);
      String test = text.toString();
XmlPullParserFactory parserCreator =
        XmlPullParserFactory.newInstance();
XmlPullParser parser =
        parserCreator.newPullParser();
parser.setInput( text.openStream(), null )
String tag;
int parserEvent = parser.getEventType();
while (parserEvent !=
            XmlPullParser.END_DOCUMENT ){
switch(parserEvent){
case XmlPullParser.TEXT:
tag = parser.getName();
break;
case XmlPullParser.END_TAG:
tag = parser.getName();
break;
}
parserEvent = parser.next();
}

}catch( Exception e ){
Log.e("dd", "Error in network call"+ e);
}
```

그림 6. REST 웹서비스 Open API를 사용한 클라이언트-서버 통신방법
Fig. 6 Client-Server Communication with REST Web Service Open API

그림 4는 스마트폰 등에서 활용할 수 있는 REST 웹서비스 Open API 실제 사례이다. 그림 4는 실제로 웹문서를 검색하고 그 결과를 반환하는 REST 웹서비스 Open API이다. 즉, 스마트폰 애플리케이션이 특정한 검색어를 입력받은 뒤 그 검색결과를 제공하는 목적인 경우, 다음과 같은 순서로 서비스를 사용하면 된다. 우선, 그림 3에서 제공하는 성인검색어 여부 판별을 위한 Open API를 먼저 사용한 후, 그림 4에서 제공하는 웹문서 검색기능 Open API를 사용하여 실제적으로 웹문서를 검색하고 결과를 화면에 제공한다.

그림 5는 에러메시지이다. 그림 3이나 그림 4의 Open API 호출을 수행하는 과정에서 오류가 발생하는 경우 그림 5에서와 같은 에러코드 중 하나를 반환한다. 예를들어, 네트워크 오류, 잘못된 인자값 입력, 잘못된 키값 오류 등의 에러상황을 나타낸다. 그림 6는 스마트폰 애플리케이션에서 위와같은 각각의 Open API를 활용하여 서버측과 통신하는 과정을 나타낸다. 특히, 그림 6 코드는 URL 접속을 통해 얻어온 데이터(REST Open API 실행 후 리턴 데이터)를 XML 파서에 서 분석하는 과정을 보여주고 있다.

이와 같이, REST Open API 웹서비스는 모바일 환경에서 클라우드 컴퓨팅 플랫폼의 컴퓨팅 능력을 활용하기 위해 편리한 통신방식이다. 사실 편리함의 정도는 객관적으로 비교할 수 없지만, 다른 척도(성능)에 대해서는 객관적인 비교가 가능하다. 따라서, 본 연구에서는 이러한 REST API를 활용한 통신방식이 소켓통신 방식에 비하여 어느정도의 성능을 갖고 있는지를 평가하고 그 결과를 제시하고자 한다. 일반적으로, 스마트폰을 활용한 모바일-클라우드 컴퓨팅 환경에서 스마트폰은 방대한 계산량을 수행하기에는 컴퓨팅 파워가 충분하지 않다. 따라서, 이러한 응용에서는 계산량이 많거나 계산시간이 오래걸리는 부분을 클라우드 컴퓨팅에서 대신수행하는 방법을 채택한다.

본 연구에서는 실제로 파이(pi)값을 병렬처리 방식으로 계산하는 예제를 구현하여 소켓통신 방식과 REST 웹서비스 Open API 방식의 성능을 비교하였다. 또한, REST 웹서비스 Open API를 사용하는 방법이 갖는 확장 가능성을 확인하기 위하여 성능을 시뮬레이션으로 분석하였다. 본 장에서는 파이(pi)값 계산 작업을 병렬화 하는 과정을 통해 모바일 클라우드 융합 애플리케이션을 개발하기 위한 프로시져를 보여준다. 첫번째

단계는 모바일 클라우드 융합 애플리케이션을 개발하는 데 있어서, 복수개의 클라우드 노드에서 병렬로 실행할 수 있는 작업을 찾아내는 것이다. 두 번째 단계는 종속성을 제거하는 것으로, 만약, 각 계산단계별로 종속성이 존재한다면 이는 병렬화 수행시 성능을 저하시킨다. 파이(pi)값을 구하는 방법은 급수, 다각형법, 등 여러가지를 활용할 수 있다. 본 연구에서는 모바일 클라우드 융합 플랫폼에서 파이(pi)값을 계산하기 위해서 무한 급수 방법 중 종속성을 제거하여 병렬화하는 알고리즘을 활용한다. 무한급수를 이용하는 방법은 다각형법을 이용하는 것보다 더 쉽고 정확하게 π 값을 계산할 수 있다. $\tan^{-1}(x)$ π 값을 계산하는데 있어 위의 맥클로린(Maclaurin) 수열을 통해 $\tan^{-1}(x)$의 값을 나타낼 수 있다.

$$\tan^{-1}(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots + (-1)^n \frac{x^{2n+1}}{2n+1} + \cdots \quad (1)$$

본 연구에서는 모바일 클라우드를 활용한 병렬수행을 통해π 값 계산의 성능을 향상시키고자 하므로, 이러한 방법은 적합하지 않다. 앞의 무한급수에 의한 표현은 8개의 node(혹은 processing element)에서 작업을 수행하고자 할 때 바로 이전 iteration에서 변경된 데이터가 다음 번 iteration에서 사용되는 순차적으로 종속인 관계가 있다. 즉, 앞 항의 계산결과를 반드시 알아야만 그 다음 항의 계산을 수행하여 전체를 구할 수 있다. 따라서, 클러스터 컴퓨팅 환경에서 아무리 많은 수의 node가 존재한다 할지라도 결국은 serialized 되어 수행된다. 따라서, 병렬화 수행가능한 알고리즘이 제안되어야할 필요가 있다. 본 연구에서는 파이값 계산 알고리즘을 다소 수정하여 함수의 급수표현을 적분표현으로 바꾸어 병렬수행에 매우 적합한 형태로 변경한다. 이는 위 식(1)의 양변을 x로 미분함으로써 다음 식(2)를 얻을 수 있다.

$$\pi = 4\tan^{-1}(t) = \int_a^b \frac{4}{1+t} (\forall a,b \in R) \quad (2)$$

식 (2)의 표현은 식 (1)의 무한 급수 표현과는 다르게, 적분 표현의 형태이므로 계산을 병렬화 하는데 매우 유리하다. 또한, 계산을 여러노드에 분산 처리하는 데이터 병렬화을 실현하기에 적합하다. 위의 파이값 병렬화는 작업 병렬화 및 파티셔닝을 통하여 모바일 클라우드 융합 플랫폼에서 실행된다.

Ⅳ. 성능분석

본 연구에서는 제안된 REST 웹서비스 Open API를 활용한 부하 오프로딩 실험을 위하여 두 가지 방법으로 성능분석을 진행하였다. 첫 번째는 시스템 모델링을 통한 시뮬레이션을 활용한 분석이며, 두 번째는 실제 구현을 통한 성능분석이다.

첫째로, 시뮬레이션에 의한 시스템 성능을 평가하기 위한 REST 웹서비스 Open API의 시스템 모델은 그림 7과 같다. REST Open API 웹 서비스 서버 시스템은 크

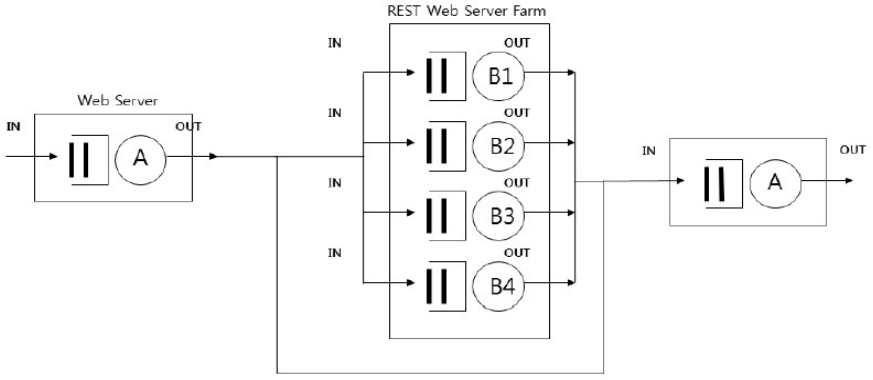

그림 7. 평가 시스템 모델
Fig. 7 Evaluation System Model

게 2가지 요소로 구성된다: (1) 웹 서버, (2) REST Open API 웹 서비스 서버 팜. 그리고, 그림 7에 도시한 바와 같이 REST Open API 웹 서비스 서버 팜 내에는 다수의 노드들로 구성된 서버들이 존재한다. 따라서, 각각의 요청은 본 시스템에 진입한 후 다수의 큐에서 동시에 서비스 받을 수 있다. 즉, 아파치 웹 서버(Apache Web Server)에 진입한 계산 요청은 REST Open API Web Service 처리를 위하여 B1에서 B4중 어느 하나에 해당하는 노드의 큐에 전달될 수 있는 모델이다.

모든 요청은 우선 REST 웹 서버로 이동하기 전에 아파치 웹 서버를 통해야 한다. 요청은 1000 requests/sec 에부터 15000 requests/sec의 속도로 웹 서버에 도착하는 것을 가정하였다. 부하를 균등하게 밸런싱하기 위해서, REST Open API웹 서버 구성은 여러 개의 병렬 클라우드 또는 클러스터 아키텍처로 구성하는 것을 가정하였다.

웹서비스 사용 요청은 아파치 웹 서버에 도착한다. 요청에 대한 초기화 프로세스는 아파치 웹 서버에서 수행된다. 그 후, 해당 요청이 REST Open API 웹 서비스를 수행하기 위한 것이라면, REST 웹 서버 Farm으로 이동한다. 만약, 해당 요청이 단지 웹 서버에 대한 요청이라면 웹 서버에서 처리 후 OUT 노드로 이동한다. 그림 7에서 REST 웹 서비스 서버 팜을 바이패스하는 경로를 둔 것은 이 때문이다. 본 성능평가를 위해 사용하는 시스템 모델은 외부 로부터 입력과 출력을 갖는 오픈 대기 네트워크의 일종이다. 분석을 간단하게 하기 위해서, 시스템의 불완전한 전송에 대해서는 그 가능성을 배제한다. 따라서, 그림 7에서 아파치 웹 서버 노드로 다시 회귀하는 경로는 추가하지 않았다. 그리고, 이러한 가정은 일반적인 사용자 웹 서비스 요청에 대해서 그 신뢰성을 100% 보장하지 않기 때문이다.

그림 8는 단일 서버(병렬 구성이 아닌)에서, 포아송 분포를 따르는 HTTP 요청이 시스템에 도착하는 속도에 따라 본 시스템의 큐에서 처리를 대기(큐 내부 작업수)하는 시간을 분석한 것이다.

REST 웹 서버 팜은 하나 이상의 컴퓨팅 서버를 통해 병렬로 구현될 수 있다. 다음 그래프는, 15,000request/sec 의 서비스 처리속도를 갖는 REST 웹서비스 Open API 서버를 복수로 구성하여 비용대비 성능 측면에서 효율적인 서비스를 제공하고자 할 때, 최적의 병렬 노드수를 계산하기 위한 실험이다.

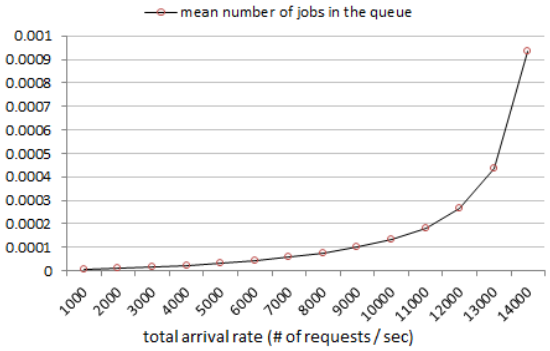

그림 8. 도착속도에 따른 큐 내부의 작업 수
Fig. 8 Mean number of jobs depending on total arrival rate

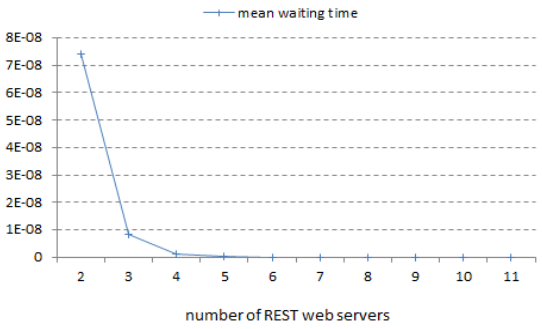

그림 9. REST 웹 서비스 서버 팜의 노드수 증가에 따른 평균 대기시간의 변화
Fig. 9 Mean waiting time depending on the number of REST web servers

위 실험을 위하여 본 연구에서는 M/M/m 오픈 큐잉 네트워크으로 그림 3의 구성 요소 B를 모델링하였다. 이 때, m 값을 2부터 11까지 증가시킬 때 나타난 mean waiting time을 그림 9에 도시하였다. 이것은 REST Open API 웹 서비스 서버가 여러개 노드로 구성된 상황을 의미한다. 실제로, 그림 9에서 나타난 바와 같이 가장 비용대비 성능 효율적한 병렬 REST 웹서비스 Open API 구성은 4개 서버를 도입하는 경우임을 알 수 있다.

다음은 실제 시스템 구현을 통한 성능분석 결과를 제시한다. 본 연구에서는 REST 웹서비스 Open API 서버 팜을 구성하는 개별 웹 서비스 노드에 Jersey 1.6 REST 서버 4개를 병렬로 배치하였다. 따라서, 이러한 REST

Open API 서버 플랫폼에 REST Web Service를 개발하여 탑재하기 위하여 Java programming language를 사용하여 웹서비스를 구현하였다. 총 노드 수는 1대부터 8대까지의 실제 노드를 활용하여 실험을 진행하였다. 각 노드은 2GB 메모리를 가진 코어2 듀오2GHz 시스템이며, 1Gbps 이더넷으로 연결된다. 그림 9에서와 같이 노드 수가 증가함에 따라서, 파이(pi) 값 계산을 위한 전체 수행시간(평균 요청 대기시간에 비례)은 줄어든다. 본 실험에서 활용한 병렬화 파이 값 계산 알고리즘은 모든 참여노드에서 동일한 분량을 분산하여 처리(계산)하도록 하므로, 각 노드에서의 수행시간은 거의 유사하다. 물론, 최종수행 시간은 통신 오버헤드가 포함되며, 순수 계산시간과는 차이가 있다. x축의 각 열은 계산 참여 노드 수를 나타낸다. 예를들어, 파이값 계산 알고리즘이 4개 참여노드에게 동일한 분량의 데이터를 분산하여 처리하는 경우를 소켓 통신을 사용하는 방법과 REST Open API를 사용하는 방법으로 각각 구현하여 실행하고 성능을 측정한 결과를 나타낸다.

REST 웹서비스 Open API를 적용한 성능향상의 주요 원인은 다음과 같다. 앞서 언급한 바 있듯이, 소켓을 이용한 통신방법에서 프로토콜 설계는 임시변통적(add-hoc)으로 체계적이지 않은 경우가 많다. 즉, 소켓 자체의 성능 문제라기 보다는 이 소켓을 활용하여 애플리케이션 레이어(layer)에서 설계하는 통신 프로토콜이 체계적이지 않다는 것이다. 즉, 스마트폰 애플리케이션 제작자에 따라 필요에 따라 자신들의 프로토콜(메시지 교환)을 만들어 사용하므로 임시변통적이고 주먹구구식으로 구현하는 경우, 불필요한 메시지 교환이 발생하여 성능이 저하된다. 반면, REST 웹서비스 Open API를 사용하는 방식은 Apache Tomcat 웹서버/웹애플리케이션 서버의 성능을 그대로 취할 수 있으므로 불필요한 메시지교환 없이 수행할 수 있다.

결과적으로 그림 10에 따르면, 각 노드수의 변화(1부터 8까지)에 따른 성능향상 평균값으로 약 5.2%을 얻었다. 참고로, 그림 10에서 짙은색 막대바는 소켓통신 방식을 활용한 통신 수행시간을 측정한 것이고, 옅은색 막대바는 REST 웹서비스 Open API에 의한 통신 수행시간을 측정한 것이다. 또한, 이번 실험결과에 있어서도 노드 수가 4개 이상일 때 노드 1개를 추가하는 비용에 비하여 얻어지는 성능향상 효과가 미미한 것을 확인할 수 있다.

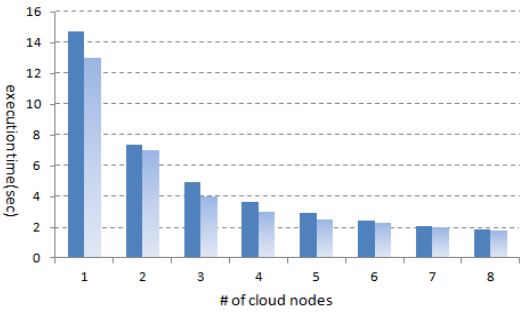

그림 10. 클라우드 노드수에 따른 실행시간
Fig. 10 Execution time as increasing number of cloud nodes

Ⅴ. 결 론

REST 웹서비스 Open API를 소개하고 실제적인 응용을 예로들어 REST 웹서비스 Oepn API의 사용법을 확인하였다. 기존 스마트폰 애플리케이션에서는 주로 클라이언트-서버간의 소켓 통신을 활용하여 분산환경을 구현하였는데 반하여, 본 연구에서는 REST 웹서비스 Open API를 사용하여 서버측에 데이터를 전달하고, 서버에서 데이터를 처리한 후 그 결과를 XML로 수신한다. 따라서, 이렇게 수신된 XML을 파싱하여 원하는 데이터를 수신함으로써 기존의 소켓 통신기법을 충분히 대체할 수 있다. 뿐만 아니라, 본 연구에서 제안하는 방법은 아파치 웹서버와 REST 웹서비스 확장 라이브러리를 통해 구현하므로 소켓에 비해 매우 신뢰성있는 클라이언트-서버 아키텍쳐를 손쉽게 구축할 수 있는 장점이 있다.

본 연구에서는 모바일-클라우드 환경을 위한 REST 웹서비스 Open API를 활용한 통신방법을 제안한다. 일반적으로 모바일-클라우드(클라이언트-서버) 환경에서는 소켓을 이용하여 원격지 통신을 구현하는 경우가 대부분이다. 그러나, REST 웹서비스 기반의 Open API 방식을 적용하는 것이 성능측면에서 유리함을 성능분석을 통해 제시하였다.

감사의 글

본 논문은 2012학년도 충북대학교 학술연구지원사업의 연구비 지원에 의하여 연구되었음(This work was supported by the research grant of the Chungbuk National University in 2013).

REFERENCES

[1] MarketResearch.com, Market Research Projects Smartphone Market Growth at 19% CAGR through 2016, http://www.marketresearch.com/corporate/aboutus/press.asp?view=3&article=2775.

[2] B. Keane and A. Sabadra, M. Diamond, Industry Update : Mobile Payments Strategy, Deutsche Bank Market Research, Mar. 2012.

[3] http://www.netxt.com/gartner2013-years-mobile-phone-will-replace-the-pc-into-a-major-internet-device/

[4] W3C, http://www.w3c.org

[5] JIL, http://www.jil.org/

[6] BONDI, http://bondi.omtp.org/

[7] http://www.ajaxonomy.com/2008/xml/web-services-part-1-soap-vs-rest

[8] SMAC, http://www.smac.or.kr/

[9] Xia Zhao, "RESTful Web Service Composition : Extracting a Process Model from Linear Logic Theorem Proving", *2011 IEEE International Conference on Next Generation Web Services Practice (NWeSP)*, pp. 398-403, Oct. 2011.

[10] Haibo Zhao, "Towards Automated RESTful Web Service Composition", *2009 International Conference on Web Services (ICWS)*, pp. 189-196, July 2009.

[11] Zheng Li and Liam O'Brien, "Towards Effort Estimation for Web Service Compositions using Classification Matrix", 2010

[12] Cesare Pautasso, Olaf Zimmermann, Frank Leymann, "RESTful Web Services vs. Big Web Services : Making the Right Architectural Decision", *International Conference on WWW*, pp. 805-814, Apr. 2008.

[13] Rosa Alarcon, Erik Wilde, and Jesus Bellido, "Hypermedia-Driven RESTful Service Composition", pp. 111-120, *ICSOC* 2010.

[14] Cesare Pautasso, "RESTful web service composition with BPEL for REST," *Data and Knowledge Engineering*, vol. 68, no. 9, pp. 851-866, September 2009.

[15] Y. Noh, Y. Byun, D. Lee, "Load Balancing for RFID Middleware", *J. Korea Inst. Inf. Commun. Eng.* Vol. 17, No. 10 : 2288~2293, Oct. 2013

[16] AlShahwan, F. Moessner, K and Carrez, F, "Evaluation of Distributed SOAP and RESTful Mobile Web Services.", I*nternational Journal on Advances in Networks and Services*, Vol. 3, No. 34, pp. 447-461, 2011.

[17] Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000.

Min Choi

received the M.S. and Ph.D. degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST) in 2003 and 2009, respectively. From 2008 to 2010, he worked for Samsung Electronics as a Senior Engineer. Since 2011 he has been a faculty member of Department of Information and Communication of Chungbuk National University. His current research interests include embedded system, computer architecture, and mobile cloud.