Journal of the Korean Association of Geographic Information Studies, 21(1) 2018, pp. 96-105 ISSN 1226-9719(Print)On
https://doi.org/10.11108/kagis.2018.21.1.096 ISSN 2287-6952(Online)

데이터 직렬화 기법을 활용한 차세대 브이월드 3차원 데이터의 호환성 개선 방안*

강지훈[1※] · 김현덕[1] · 김정택[2]

Improving Compatibility Method of New Vworld 3D Data Using the Serialization Technique*

Ji-Hun KANG[1※] · Hyeon-Deok KIM[1] · Jeong-Taek KIM[2]

요 약

공간정보 오픈플랫폼 지도서비스인 브이월드에서는 다양한 국가공간정보를 제공하고 있다. 최근 IT기술발달에 따라 사물인터넷, 자율주행차 등의 신산업과 융합할 수 있는 3차원 공간정보의 수요가 증가하고 있다. 3차원 공간정보 데이터는 구조가 복잡하고 용량이 커서 서비스를 위해 많은 컴퓨터 자원이 요구된다. 브이월드를 비롯한 3차원 지도 서비스는 대부분 성능을 고려해 바이너리 형태의 데이터를 구축하여 사용하고 있다. 그러나 이러한 형태의 데이터는 명세서에 대한 정확한 이해가 없는 경우에 다른 서비스에서 사용하기 어렵다. 따라서 본 연구에서는 바이너리 형태로 구축된 차세대 브이월드 3차원 포맷의 호환성을 개선하고 사용자의 편리성을 높이기 위해 데이터 직렬화 기법을 제안하였다. 바이너리 데이터와 데이터 직렬화 기법을 적용한 데이터의 성능을 테스트한 결과 직렬화된 데이터의 성능이 바이너리 데이터와 유사한 성능을 보였다. 따라서 직렬화가 적용된 3차원 데이터 포맷을 사용한다면 브이월드 성능의 저하 없이 다른 서비스와의 호환성을 얻을 수 있다.

주요어 : 공간정보 오픈플랫폼, 브이월드, 3차원 건물데이터, 포맷, 데이터 직렬화

ABSTRACT

The V-world, Spatial information open platform map service, provides various national spatial data. Recently, with the development of IT technology, demand for 3D geospatial data that can be merged with new industries such as Internet of Things(IoT) and

2018년 3월 13일 접수 Received on March 13, 2018 / 2018년 3월 22일 수정 Revised on March 22, 2018 / 2018년 3월 22일 심사완료 Accepted on March 23, 2018
* 본 연구는 국토교통부 도시건축개발사업 연구비지원(13도시건축A02)에 의해 수행되었음.
1 한국국토정보공사 공간정보연구원 Spatial Information Research Institute of Korea Land and Geospatial Informatix Corp
2 (주)씨엠월드 연구소 CmWorld INC. R&D center
※ Corresponding Author E-mail : kangdaejang@lx.or.kr

autonomous vehicles is increasing. Because 3D geospatial data is large and complex, many computer resources are used to provide map services. Most of the 3D map services, such as Vworld, are constructed binary data in consideration of performance. However, this type of data is incompatible because it is difficult to use in other services if there is no precise understanding of the specification. In this paper, we propose a data serialization method to improve the compatibility of new Vworld 3D format which is constructed in binary form. The performance of binary data and serialized binary data is tested and compared. As a result, it is expected that the data using the serialization technique will be similar to the binary data and contribute to improve compatibility.

KEYWORDS : Spatial Information Open Platform, Vworld, 3D Building Data, Format, Data Serialization

서 론

최근 사물인터넷(IoT), 자율주행차, 가상 및 증강현실 등의 기술이 발달함에 따라 이와 융합하여 다양한 부가가치를 창출할 수 있는 3차원 공간정보가 주목받고 있다. 특히 2차원 정보에서 높이값만 추가된 단순한 형태의 3차원 공간정보가 아니라, 현실세계와 동일하게 표현된 고정밀 3차원 공간정보에 대한 수요가 크게 증가하고 있다. 국토교통부에서는 이러한 공간정보에 대한 수요에 대응하고자 2012년부터 공간정보 오픈플랫폼 지도서비스인 브이월드를 출시하고 국가공간정보를 제공하고 있다.

브이월드의 서비스에 대한 통계에서 이러한 수요를 확인할 수 있는데, 3차원 지도 서비스가 오픈 API 서비스들 중에 가장 많이 활용되는 기능으로 나타났다(Jang *et al.*, 2016). 또한 정부 부처, 학계, 공공기관, 민간 업체 등 다양한 기관에서 연구나 개발을 목적으로 3차원 공간정보 데이터 제공 요청이 지속되고 있다(SpaceN, 2017(a); SpaceN, 2017(b); SpaceN, 2017(c); SpaceN, 2017(d)). 이렇게 3차원 공간정보 데이터의 활용도가 높아지고 수요가 증가하고 있으나, 3차원 데이터는 대체로 용량이 크고 복잡하여 공유하기에 어려움이 있다. 이는 3차원 공간정보의 활용 확산에 장벽이 되어 표준을 필요로 한다. 그러나 표준 포맷은 호환성과 표맷간 교환이 주된 목적이므로 다양한 포맷을 수용할 수 있는 구조로 이루어져 있어 서비스에는 적합하지 않다. 예를 들어 대표적인 표준 포맷인 CityGML의 경우 좌표계 적용이 엄격하고 오브젝트 모델링에 복잡한 3차원 폴리곤을 사용하기 때문에 보통 데이터를 서비스할 때 직접적으로 사용하지 않는다(Gaillard, 2015; Schilling *et al.*, 2016).

3차원 공간정보의 활용 확산을 지원하기 위해 최근 브이월드의 3차원 데이터 서비스 개선에 대한 연구가 진행되고 있다. Kim *et al.* (2017)는 3차원 지도서비스의 성능 향상을 위해 3D Tiles의 Batched 3D Model(B3DM)과 쿼드트리 알고리즘을 적용한 서비스 개선안을 제안하였다. 현재 브이월드의 3차원 지도서비스는 수많은 이종의 건물들이 각각 단일 모델로 구성되어 있어 모델 요청과 이를 화면에 그리는 드로우 콜(draw call)이 다수 발생한다. 따라서 모델 요청수와 드로우 콜을 줄일 수 있는 3D Tiles의 B3DM과 렌더링 대상의 탐색 속도를 향상시킬 수 있는 쿼드트리 알고리즘을 적용하는 방안을 제안하였다. Kang(2017)은 최근 3차원 지도 서비스 환경 변화를 반영하고, 3차원 공간데이터의 렌더링 속도를 개선하기 위한 방안을 모색하였다. HTML5 및 WebGL 환경에서 산술처리 속도를 향상시키고, 개별 구축된 모델링 데이터를 타일 단위로 렌더링함으로써 속도를 개선하는 방안을 제안하였다. 서비스 방식을

개선하는 연구 외에 포맷 자체를 개선하는 연구도 진행되었다. Kim *et al.*(2017)는 HTML5 및 WebGL 기반의 환경을 고려하여 3차원 건물데이터 포맷의 서비스 사양과 리소스(Resource) 데이터 구조를 개선한 새로운 포맷을 제안하였다. 이상의 연구들은 대부분 속도 측면의 서비스를 개선하기 위한 연구라 할 수 있다. 브이월드는 국가의 공간정보를 제공하고 활용도를 높이기 위한 목적으로 운영되고 있으므로, 상호운영성 측면에서의 개선도 필요하다. 3차원 공간정보와 같이 컴퓨터 자원을 많이 사용하는 서비스에서는 성능을 최대화하기 위하여 바이너리 형태로 데이터를 저장하고 서비스하는데, 브이월드 역시 서비스를 위한 3차원 데이터는 현재 성능을 고려하여 바이너리 형태로 구축되어 있다. 그러나 이러한 데이터는 포맷명세서가 정확하게 공개되지 않으면 사용하기 어려운 단점을 가지고 있다. 이와 반대되는 개념으로 직관적인 사용이 가능한 JSON이나 XML과 같은 텍스트 기반의 포맷도 있지만 데이터가 커지고 속도가 느리다는 단점이 있다. 이와 같은 데이터 형태의 장단점을 고려하여 본 연구에서는 Kim *et al.*(2017)에서 제안한 포맷을 활용하여 브이월드에 적합한 데이터 직렬화 기법의 도입을 제안하고자 한다. 이를 위해 우선 대표적인 데이터 직렬화 기법들을 비교분석하고, 서비스 성능 테스트를 수행하였다.

브이월드의 3차원 건물데이터 서비스 구조

브이월드의 3차원 건물데이터는 지도상에 건물을 위치시키기 위한 인덱스(Index) 파일, 3차원 건물의 지오메트리 정보를 가지고 있는 객체(Object) 파일, 건물 외벽의 이미지 정보인 텍스쳐(Texture) 파일로 구성된다(Kim *et al.*, 2017). 이와 같이 3개의 파일로 구성된 3차원 데이터는 클라이언트(Client)의 요청에 따라 그림 1과 같은 절차로 사용자에게 제공된다.

현재 브이월드 3차원 건물데이터의 인덱스 파일은 DAT 포맷, 객체 파일은 XDO 포맷, 텍스쳐 파일은 JPG 포맷으로 서비스되고 있다. Kim *et al.*(2017)는 서비스 사양과 리소스 구조를 개선하여 차세대 브이월드 3차원 데이터 사양을 제안했다. 새롭게 제안한 데이터를 현재 브이월드 3차원 데이터 구조와 비교해보면 인덱스 파일은 SPI, 객체 파일은 SPO, 텍스쳐 파일은 JPG로 대응된다. 이중 3차원 모델 정보를 서비스하기 위한 정보를 포함한 SPI 포맷의 데이터의 사양은 표 1, 2와 같다. 3차원 건물을 구성하는 모든 데이터는 성능을 위해 바이너리 형태로 저장되어 서비스되고, 데이터를 사용한 새로운 서비스 개발을 지원하기 위해 명세서를 공개하고 있다. 이는 3차원 서비스와 같이 컴퓨터 자원을 많이 사용하는 서비스에서 성능을 최대화하기 위한 방법이다. 하지만 바이너리 형태의 서비스 포맷은 명세서 공개만으로는 새로운 서비스를 개발하는 측면에서 오류의 가능성을 내포하고 있다. 명세서의 오탈자 하나나 작은 정보 누락으로도 데이터 전체를 사용할 수 없는 경우가 발생하기 때문이다. 또한 바이너리 인코딩 정보를 명세서에 명확히 정의했더라도 데이

TABLE 1. The main information of the new Vworld 3D index format

Name	Type	Byte	Remarks
Signature	U08	1*4	8A 54 9D 00
Version	U08	1	Version information is incremented by 1 from 1.
Level	U08	1	
Tile_X	U32	4	
Tile_Y	U32	4	
ObjType	U08	1	
ObjectCount	U32	4	
modelSummary	modelSummary	Sizeof(modelSummary) * ObjectCount	modelSummary Structure Reference

TABLE 2. Details of the new Vworld 3D index format model

Name	Type	Byte	Remarks
Type	U08	1	Structure type
Version	U08	1	Structure version
Building_id_length	U08	1	The length of Name does not exceed 255.
Building_id	I08	1* Building_id_length	
Latitude	F64	8	
Longitude	F64	8	
Elevation	F32	4	
Bounding_box	F32	4 * 6	It is treated as a float because it is information that does not require precision.
Texture_level_count	U08	1	
ModelFileName_length	U08	1	The length of the Name does not exceed 255
ModelFileName	I08	1* ModelFileName_length	

터 제작과정에서 발생한 실수로 인해 인코딩 명세와 다른 결과를 도출할 수도 있다. 뿐만 아니라, 프로그램 언어별로 데이터를 사용하기 위한 코드를 새로 작성해야 하는 불편함이 있다.

바이너리 형식으로 설계된 차세대 브이월드의 3차원 데이터 포맷이 공간정보 오픈플랫폼으로서 그 역할을 수행하기 위해서는 명세서 공개 외에 데이터 사용 편리성을 개선하기 위한 추가적인 지원이 필요하다.

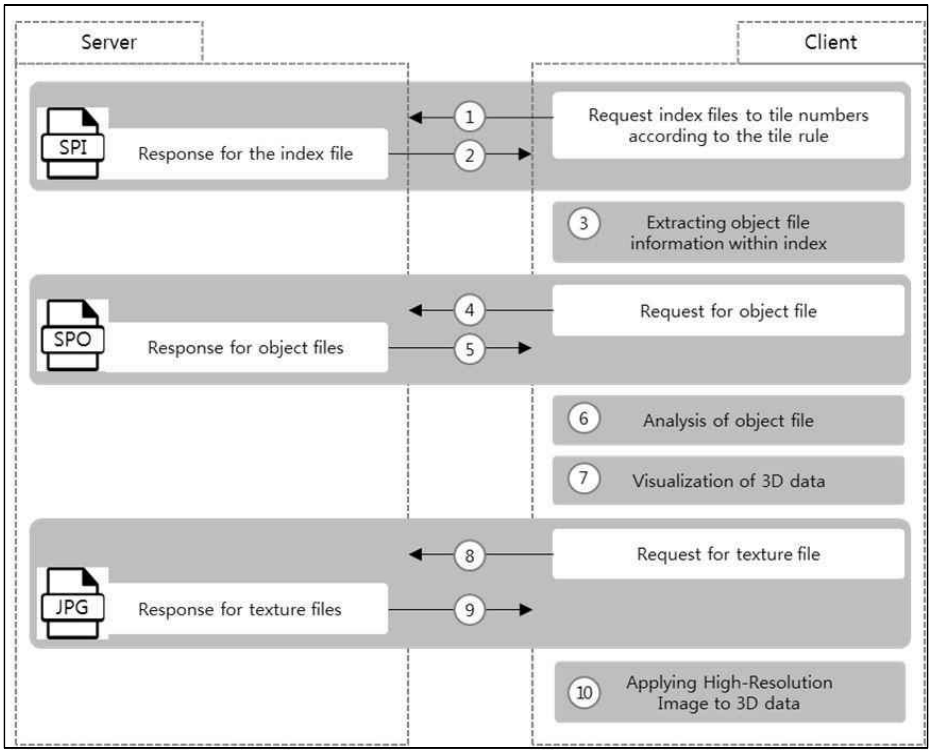

FIGURE 1. Service flowchart of Vworld 3D building data

데이터 직렬화 기법 비교

바이너리 형태로 제작된 차세대 브이월드 데이터의 호환성 개선을 위한 방법으로 데이터 직렬화 기법의 적용을 검토하였다. 검토 대상은 구글의 프로토콜 버퍼(Protocol Buffer)와 아파치(Apache)의 에이브로(Avro)이다. 프로토콜 버퍼는 구글의 검색 서비스를 비롯한 대부분의 서비스에 적용되어 사용되고 있고(Google, 2017), 관련된 참고자료를 찾기에 용이하다. 에이브로 방식의 경우 분산처리 프레임워크인 하둡(Hadoop)에 적용되어 있고, 별도의 컴파일 작업 없이 바로 사용할 수 있는 특징이 있다(Hunt, 2017). 기존의 바이너리 형태의 데이터(DAT, SPI)와 데이터 직렬화 기법이 적용된 인덱스 데이터(SPI-pbuf, SPI-avro)의 특징을 비교하면 표 3과 같다.

데이터 직렬화 기법은 특정 언어로 데이터의 구조를 선언한 후, 컴파일을 통해 데이터를 사용하기 위한 코드를 자동으로 생성한다. 프로토콜 버퍼와 에이브로에서 사용하는 데이터 선언 언어와 컴파일러가 지원하는 프로그램 언어는 표 4와 같다.

앞장에서 기술한 SPI 확장자를 가진 인덱스 데이터를 프로토콜 버퍼 문법으로 선언하면 그림 2와 같고, 에이브로 방식으로 선언하면 그림 3과 같다. SPI 포맷의 데이터는 바이너리 포맷이라 텍스트 뷰어를 통해 확인할 수 없고, 활용을 위해서는 포맷에 대한 정확한 명세서가 공개되어야 한다. 또한 프로그램 언어별로 데이터 정보를 읽거나 기록하기 위한 코드를 별도로 제작해야 한다. 하지만 프로토콜 버퍼나 에이브로 기법을 적용한 데이터는 그림 2와 그림 3과 같이 쉽게 데이터 구성을 확인할 수 있고, 사용하고자 하는 프로그램 언어만 선택하면 자동으로 데이터 입출력을 위한 라이브러리를 제작해 주어 활용이 편리하다.

서비스 성능 테스트

1. 테스트 환경

TABLE 3. Components of index file

Format	Strength	Weakness
DAT SPI	· It is fast, for it is binary format. · Small size because it does not contain separate field information. · No compilation required.	· Clients need to know the format definition correctly. · Encoding specifications and processing are required to maintain interoperability.
SPI-pbuf	· It is fast, for it is binary format. · No compatibility problems. · Clients can use the format definition without knowing it	· Storage of attribute type information increases file capacity. · Requires separate compilation and deployment tasks.
SPI-avro	· It is fast, for it is binary format. · No compatibility problems. · Clients can use the format definition without knowing it. · It can be used without compiling and distributing separately.	· Increase file capacity due to separate field information storage.

TABLE 4. Basic information of protocol buffer and avro

Item	Protocol buffer	Avro
Declaration	Proto2, Proto3	JSON
Supported languages	C++, Java, Python, and JavaScript	Java, C, C++, C#, Python, Ruby
Code compile	Necessary	Selectable

```
syntax = "proto3";
message ndatBuildingsPBuf {
    int32 version = 1;
    int32 level = 2;
    int32 idx = 3;
    int32 idy = 4;
    int32 objtype = 5;

  message ndatBuildingSummaryPBuf {
    int32 type = 1;
    int32 version = 2;
    string key = 3;
    double centerPosX = 4;
    double centerPosY = 5;
    float altitude = 6;
    float boxminX = 7;
    float boxminY = 8;
    float boxminZ = 9;
    float boxmaxX = 10;
    float boxmaxY = 11;
    float boxmaxZ = 12;
    string dataFileName = 13;
    int32 imgLevelCount = 14;
  }

  repeated ndatBuildingSummaryPBuf objarray = 6;
}
```

FIGURE 2. Declaration of protocol buffer method

```
{
   "namespace": "lx.vworld",
   "type": "record",
   "name": "NDatBuildingAvro",
   "fields": [
        {"name": "version", "type": "int", "default": 3 },
        {"name": "level", "type": "int"},
        {"name": "idx", "type": "int"},
        {"name": "idy", "type": "int"},
        {"name": "objtype", "type": "int", "default": 0 },
        {
          "name": "objArray",
          "type": {
            "type": "array",
            "items": {
                "type": "record",
                "name": "NDatModelSummaryAvro",
                "type": "record",
                "fields": [
                {"name": "type", "type": "int"},
                {"name": "version", "type": "int"},
                {"name": "key", "type": "string"},
                {"name": "x", "type": "double"},
                {"name": "y", "type": "double"},
                {"name": "h", "type": "float"},
                {"name": "boxminX", "type": "float"},
                {"name": "boxminY", "type": "float"},
                {"name": "boxminZ", "type": "float"},
                {"name": "boxmaxX", "type": "float"},
                {"name": "boxmaxY", "type": "float"},
                {"name": "boxmaxZ", "type": "float"},
                {"name": "filename", "type": "string"},
                {"name": "imglevcount", "type": "int"}
                ]
            }
          }
        }
   ]
}
```

FIGURE 3. Declaration of avro method

데이터 공개와 활용에 용이한 데이터 직렬화 기법을 적용한 데이터와 성능에 장점이 있는 기존 방식의 데이터의 서비스 속도를 비교하기 위한 테스트를 진행했다. 전체 서비스 절차 중 인덱스 파일을 요청하고 객체 파일 정보를 추출하는 작업에 해당하는 내용이므로 인덱스 데이터를 대상으로 했다.

테스트를 위한 시스템 환경은 표 5와 같은 장비를 동일 네트워크의 근거리 통신으로 구성했다(그림 4 참조). 서비스 요청 API는 현재 브이월드의 서비스 요청 API와 동일하게 구성하고 4개의 포맷 테스트를 위해 포맷의 형식을 지정하기 위한 파라미터(Proto)를 추가 했다(그림 5 참조).

테스트 데이터는 브이월드의 여의도 지역 데이터 249개를 대상으로 각각의 포맷별로 준비했다. 폴더 구조와 파일의 명명 규칙은 현재 브이월드와 동일하게 구성했다. 포맷별 데이터 생성은 DAT 포맷의 원본 데이터를 SPI 포맷으로 변환한 후, 프로토콜 버퍼와 에이브로 기법을 적용한 데이터를 생성했다.

2. 포맷별 서비스 성능 테스트

성능 비교를 위한 테스트 시나리오는 크게 두 가지로 구성했다. 우선 인덱스 데이터를 호출하여 응답받는 시간을 체크하기 위해 249개의 서로 다른 URL을 순차적으로 호출하고 다운로드

TABLE 5. Test Equipment Specification

	Hardware	Software
Server	CPU: Intel i5 6600 Memory: 8GB HDD: 2TB	windows 10
Client	CPU: Intel i7 7700 Memory: 16GB HDD: 1TB	windows 10, chrome

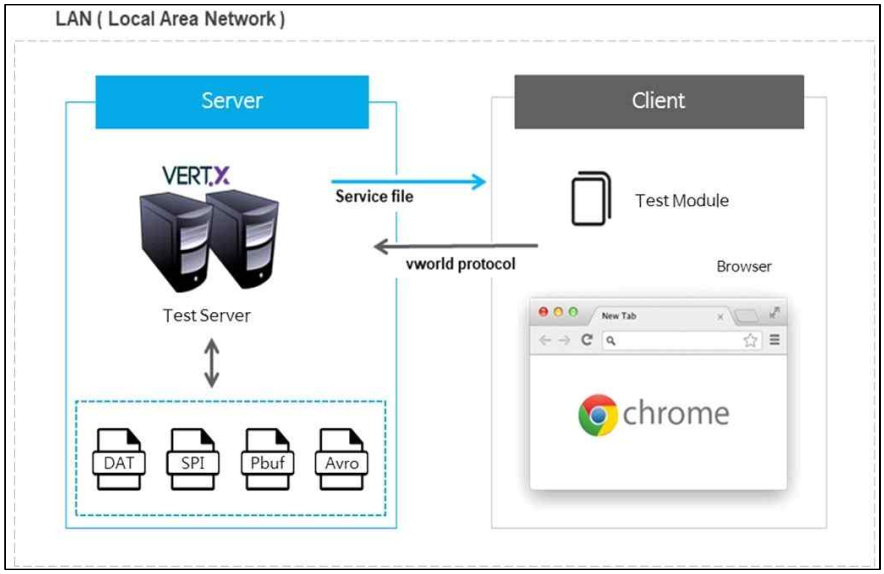

FIGURE 4. Configuration of system environment

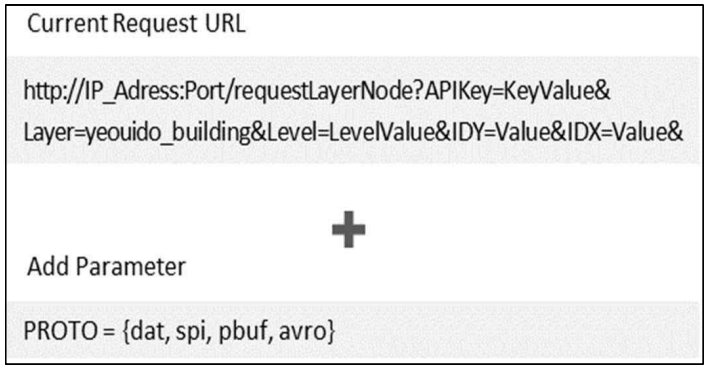

FIGURE 5. Rules of the service request API

받는데 걸리는 시간을 측정했다. 이후 다운로드 받은 249개 데이터의 정보를 해석하는데 걸리는 시간을 각 데이터별로 측정하였다. 이와 같은 방식의 테스트를 20회씩 반복하여 테스트를 진행하고 전체 소요시간과 평균시간을 계산해 4개 포맷을 비교했다. 예를 들어 테스트 데이터 3번의 테스트 결과는 표 6과 같다. 그리고 249개의 테스트 데이터 전체에 대한 테스트 결과를 정리하면 표 7과 같다.

파일의 크기는 현재 브이월드 인덱스 포맷인 DAT가 가장 크고 차세대 브이월드 포맷인 SPI와 직렬화 기법을 적용한 파일들이 상대적으로 작았다. 이는 차세대 브이월드 인덱스 포맷이 현재 브이월드 포맷에 비해 경량화되었고, 데이터 직렬화 기법은 차세대 포맷에 적용했기 때문이다.

테스트 데이터 3번의 결과를 살펴보면 첫 번째 테스트인 파일 요청 후 다운로드 시간은 에이브로 기법이 적용된 파일이 가장 빨랐다. 그리고 SPI, 프로토콜 버퍼, DAT 파일 순으로 나타났다. 공통적으로 첫 번째 요청에 대한 응답 시간이 가장 느리게 나타났는데, 두 번째 요청

TABLE 6. Result of 3rd test data

	Request time (ms)				Parsing time (ms)			
	DAT	SPI	SPI-pbuf	SPI-avro	DAT	SPI	SPI-pbuf	SPI-avro
File size	352byte	271byte	306byte	263byte	352byte	271byte	306byte	263byte
1	3.500	3.100	3.400	3.200	0.100	0.062	0.113	0.077
2	3.400	3.100	3.400	7.200	0.037	0.037	0.049	0.035
3	5.900	3.200	3.300	3.000	0.036	0.035	0.045	0.033
4	3.200	3.400	3.200	3.700	0.060	0.037	0.046	0.036
5	3.600	3.400	3.600	3.600	0.037	0.036	0.045	0.037
6	7.000	3.200	3.000	3.400	0.039	0.045	0.045	0.034
7	3.200	2.900	3.600	3.100	0.045	0.036	0.045	0.041
8	7.300	4.400	3.200	3.300	0.037	0.043	0.045	0.038
9	7.500	6.900	7.600	3.000	0.036	0.035	0.047	0.036
10	3.300	3.000	3.800	7.200	0.038	0.041	0.047	0.035
11	3.200	2.800	3.800	4.000	0.038	0.039	0.045	0.035
12	6.600	3.200	4.100	4.000	0.038	0.042	0.046	0.042
13	7.400	3.200	3.300	3.500	0.043	0.059	0.046	0.032
14	3.300	3.500	2.500	7.200	0.040	0.039	0.045	0.043
15	7.900	3.200	7.300	3.600	0.037	0.038	0.046	0.035
16	3.200	3.400	3.400	3.100	0.038	0.037	0.044	0.032
17	4.100	3.000	4.500	3.400	0.039	0.038	0.045	0.033
18	3.300	7.600	5.400	3.400	0.037	0.041	0.045	0.033
19	3.700	7.800	3.400	2.900	0.042	0.035	0.044	0.041
20	3.200	7.000	7.800	3.100	0.039	0.035	0.046	0.032
Sum.	93.800	81.300	83.600	78.900	0.856	0.810	0.979	0.760
Avg.	4.690	4.065	4.180	3.945	0.043	0.041	0.049	0.038

TABLE 7. Results of test summary

	Request time				Parsing time			
	DAT	SPI	SPI-pbuf	SPI-avro	DAT	SPI	SPI-pbuf	SPI-avro
File size (byte)	101985	78204	88725	76212	101985	78204	88725	76212
Sum. (ms)	20185.500	21204.900	21362.500	21928.700	263.316	245.046	299.609	230.022
Avg. (ms)	1009.275	1060.245	1068.125	1096.435	13.166	12.252	14.980	11.501

부터 네트워크가 최적화 경로를 찾은 것으로 판단된다.

두 번째 테스트인 파일을 읽는 속도 역시 에이브로가 가장 빨랐고 SPI, DAT, 프로토콜 버퍼 순으로 나타났다. 하지만 20회를 반복한 전체 시간이 아닌 1회를 기준으로 비교해보면 가장 빠른 포맷과 느린 포맷이 계속 변하는 것을 확인할 수 있다. 첫 번째 테스트와 유사하게 첫 번째 파일 해석 작업에서 가장 많은 시간이 소요되었는데, 이는 라이브러리 초기화 작업으로 인해 걸린 추가적인 시간으로 판단된다.

전체 데이터에 대한 테스트 결과는 3번 데이터와 또 다른 양상을 보인다. 첫 번째 테스트 시간은 미세하지만 DAT 파일이 가장 빨랐고 에이브로 기법을 적용한 파일이 가장 느렸다. 두 방식의 차이는 약 1.743초 정도로 큰 차이는 없다고 볼 수 있다. 두 번째 테스트의 경우 에이브로 방식의 파일이 가장 빨랐고 프로토콜

버퍼 방식의 파일이 가장 느렸다. 하지만 이 두 방식의 시간 차이는 첫 번째 테스트보다 더 미세한 0.07초 정도로 나타났다.

결 론

3차원 지도서비스는 데이터 취득기술의 발전과 컴퓨터 성능 및 네트워크 환경의 향상으로 많은 발전을 이루며 점차 보편화되고 있다. 국가에서 공간정보의 개방과 활용을 지원하기 위해 구축한 브이월드는 서비스 속도만큼 데이터를 활용하기 위한 환경도 중요하다. 본 연구는 브이월드의 3차원 데이터를 보다 쉽게 활용하기 위한 방법으로, 데이터 서비스를 위한 인덱스 포맷에 데이터 직렬화 기법을 적용하기 위해 서비스 성능을 테스트했다.

테스트를 네트워크 환경에서 진행하다 보니 테스트 결과가 일정하지 않고, 4개 포맷과 테스트 차수에 따라 지속적으로 변하는 현상을 보였다. 비록 내부적으로 구축한 환경이지만 네트워크 환경에서는 전송 지연, 전파 지연, 큐 지연, 노드 처리 지연 등으로 인해 일정한 속도를 담보할 수 없다. 따라서 이와 같이 일정하지 않은 결과를 보완하고 성능의 추세를 비교하기 위해 20회의 반복 테스트를 진행했다. 이를 통해 프로토콜 버퍼와 에이브로가 적용된 포맷이 기존에 구축된 바이너리 포맷과 비교해 성능이 떨어지지 않는 것을 확인했다.

차세대 브이월드 3차원 데이터 포맷에서 데이터 직렬화 기법의 도입을 검토한 이유는 호환성을 개선하기 위해서였다. 포맷별 서비스 성능 테스트 결과, 직렬화된 데이터가 바이너리 데이터와 비슷한 성능을 가진 것으로 나타났다. 따라서 성능의 저하 없이 호환성을 개선할 수 있으므로 직렬화 기법을 도입하는 것은 타당하다. 프로토콜 버퍼와 에이브로 방식은 각각의 특징이 있으므로 브이월드의 운영환경과 데이터 특성을 검토해 직렬화 기법을 선택하여 도입할 필요가 있다. KAGIS

REFERENCES

Gaillard, J., A. Vienne, R. Baume, F. Pedrinis, A. Peytavie, and G. Gesquiere. 2015. Urban data visualisation in a web browser. Proceedings of the 20th International Conference of the 3D Web Technology. Heraklion, Crete, Greece, Jun. 18-21. 2015. pp.81-88.

Google. 2017. Protocol Buffers Developer Guide. available at: https://developers.google.com/protocol-buffers/docs/overview

Hunt, P. 2017. Avro RPC Quick Start. available at: https://github.com/phunt/avro-rpc-quickstart(Accessed March 03, 2018).

Jang, H.S., D.K. Kim, J.M. Kim, and I.S. Jang. 2016. Evaluating client application status for V-World Open API service. Spatial Information Research. 24:367-376.

Kang, S.C. 2017. Rendering speed Improvement of 3D Geospatial data based on HTML5/WebGL using Tile-Rendering. M.A. Thesis, Univ. of Seoul, Seoul, Korea. pp.75-76. (강석찬. 2017. 타일렌더링 기법을 이용한 HTML5/WebGL 기반 3차원 공간데이터 렌더링 속도 개선. 서울시립대학교 대학원 석사학위논문. 75-76쪽).

Kim, H.D., J.H. Kang, and H.J. Kim. 2017. A study on the improvement of 3d building data format for spatial information open platform. Journal of the Korean Society for Geospatial Information Science 25(2):63-70 (김현덕, 강지훈, 김학준. 2017. 공간정보 오픈플랫폼 3차원 건물데이터 포맷 개선방안 연구.

한국지형공간정보학회지 25(2):63-70).

Kim, T.H., S.H. Yoo, and J.H. Go. 2017. Application of facilites for improving V-world 3D Map Service. Journal of the Korean Society for Geospatial Information Science 25(4):81-87 (김태훈, 유성환, 고준희. 2017. 브이월드 3차원 지도 서비스 향상을 위한 시설물 적용 방안 연구. 한국지형공간정보학회지 25(4):81-87).

Schilling, A., J. Bolling, and C. Nagel. 2016. Using glTF for streaming cityGML 3D city models. Proceedings of the 21st International Conference of the 3D Web Technology. Anaheim, CA, Jul. 22-24, 2016. pp.109-116.

SpaceN(a). 2017. Vworld Brief 2017. 9. p.3 (공간정보산업진흥원. 2017. 브이월드 Brief 2017. 9. 3쪽).

SpaceN(b). 2017. Vworld Brief 2017. 10. p.3 (공간정보산업진흥원. 2017. 브이월드 Brief 2017. 10. 3쪽).

SpaceN(c). 2017. Vworld Brief 2017. 11. p.3 (공간정보산업진흥원. 2017. 브이월드 Brief 2017. 11. 3쪽).

SpaceN(d). 2017. Vworld Brief 2017. 12. p.4 (공간정보산업진흥원. 2017. 브이월드 Brief 2017. 12. 4쪽).