The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 6, pp.237-242, Dec. 31, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.6.237
JIIBC 2018-6-32

Spatial MongoDB를 위한 공간 연산자

Spatial Operator for Spatial MongoDB

곽광진*, 윤하영**, 신동윤***, 신동진****, 박정민*****, 김정준******

Kwang-Jin Kwak*, Ha-Young Yoon, Dong-Yoon Shin***, Dong-Jin Shin****, Jeong-Min Park*****, Jeong-Joon Kim******

요 약 최근 인터넷과 SNS의 발전으로 미디어 데이터가 늘어나고 있으며. 사진이나 동영상은 공간 태그를 가지고 있는 경우가 많으므로 이를 분석하기 위한 많은 기술이 발전하고 있다. SNS와 같은 자유도가 높은 데이터를 처리하기 위해서 NoSQL이 각광을 받고 있으나 대부분의 NoSQL은 공간 데이터에 대한 연산 및 질의가 미비하다. 따라서 본 논문에서는 대표적인 NoSQL 중 MongoDB를 이용하여 공간 연산자를 추가하기 위한 시스템을 설계 및 구현하였다. 본 연구를 통해 다양한 연산자를 사용할 수 있음을 확인하였으며 연산자를 이용하여 다양한 서비스를 할 수 있을 것으로 기대된다.

Abstract Recently, media data is increasing due to the development of Internet and SNS. Since photographs and videos often have geo-tags, many techniques have been developed to analyze them. In order to process various kind of such as SNS, NoSQL has been covered. However, most NoSQL does not have enough computation and query about spatial data. Therefore, in this paper, we designed and implemented a system for adding spatial operators using MongoDB among the representative NoSQL. Through this study, it is confirmed that various operators can be used and it is expected that various services can be performed using operators.

Key Words : MongoDB, Document Based Database, Spatial Operator, NoSQL, GIS

Ⅰ. 서 론

최근 Facebook, Instagram, Twiter와 같은 소셜 네트워크 서비스(Social Network Service, SNS)의 발전으로 미디어 데이터의 내용이 늘어나고 있다. 이러한 SNS들은 사진이나 동영상과 같은 매체를 통해 소통을 하는데 미디어 데이터는 공간 데이터와 결합하여 장소의 정보 또는 개인의 정보와 결합하여 사용되고 이를 분석하여 다양한 서비스를 하는 방향으로 진화되고 있다. SNS 데이터는 다양한 방법으로 작성되어지기 때문에 기존의 관계형 데이터베이스보다 NoSQL을 선호한다. 특히 문서 기반 데이터베이스(Document-based Database)는 형식에 제한이 없이 쓸 수 있기 때문에 SNS 또는 로그 데이터 처리에 매우 유리하다[1-3].

*준회원, 한국산업기술대학교 스마트팩토리융합학과
**준회원, 한국산업기술대학교 컴퓨터공학과
***준회원, 한국산업기술대학교 컴퓨터공학과
****준회원, 한국산업기술대학교 스마트팩토리융합학과
*****정회원, 한국산업기술대학교 컴퓨터공학과
******정회원, 한국산업기술대학교 컴퓨터공학과
접수일자: 2018년 9월 5일, 수정완료: 2018년 11월 5일
게재확정일자: 2018년 12월 7일

Received: 5 September, 2018 / Revised: 5 November, 2018 / Accepted: 7 December, 2018
******Corresponding Author: jjkim@kpu.ac.kr
Dept. of Computer Engineering, Korea Polytechnic University, Korea.

그러나 대표적인 문서 기반 데이터베이스인 MongoDB[4]는 공간 데이터를 처리를 지원하지만 다양한 연산자를 지원하지 않음으로 단순한 공간 연산만을 지원하기에 공간 객체간의 관계성을 표현하거나 공간 객체의 다양한 표현을 하기에 부족하다[5].

따라서, 본 연구에서는 MongoDB를 이용하여 다양한 공간 연산자를 이용하여 객체간의 관계 및 연산을 구현하고자 한다.

II. 관련연구

1. MogoDB

MongoDB[4]는 대표적인 문서 기반 데이터베이스로 비구조적이며, 분산 확장성이 용이하게 설계되었다. 또한 문서의 형태로 다중 계층 구조를 지원하므로 내장 문서와 배열 등 다양한 형태의 저장 구조를 지원한다.

메모리에서 모든 연산을 수행하고 디스크에는 스레드를 통해 비동기식으로 디스크에 쓰기 때문에 속도 또한 빠른 장점을 가지고 있다. 이러한 장점으로 널리 사용되는 NoSQL이지만 공간 연산자 지원이 미비하다는 단점을 가지고 있다[6].

MongoDB가 지원하는 공간 연산자는 Box, Center, Polygon의 3가지 형태를 지원한다. MongoDB에서 지원하는 공간 연산자는 지정된 구역 내의 객체들을 찾는 기능만을 수행하며 2D 또는 2D Sphere 두 종류의 인덱스를 사용하지 않으면 사용할 수 없다는 단점을 가지고 있다. 또한 공간 연산자를 사용하기 위해 사용되는 인덱스의 경우 2D는 255까지의 좌표 범위에서 사용할 수 있고, 2D Sphere는 GeoJson만을 지원하기 때문에 널리 쓰이는 WKT84 좌표체계는 파싱을 통해 GeoJson으로 바꾸어 사용하여야 하는 불편한 점도 가지고 있다[7].

2. OpenGIS Simple Features Specificaion

OGC(Open GIS Consortium)의 OpenGIS Simple Features Specification[8]은 2차원 공간의 데이터 타입과 공간 데이터 연산을 정의하고 있다.

그림 1에서 보는 바와 같이 공간 데이터 타입은 점, 선, 면과 복합 속성으로 구성되어진다. 선에는 Line, LineString, LinearRing으로 구성되어지며, 면은 Polygon의 하위 속성을 갖는다. 복합 속성은 MultiPoint, MultiPoygon, MultiLineString 공간 속성의 배열을 다룬다. MultiSurface, MultiCurve는 개념적인 형태는 있지만 크게 쓰이지는 않고 하위 모델의 타입이 주로 사용된다.

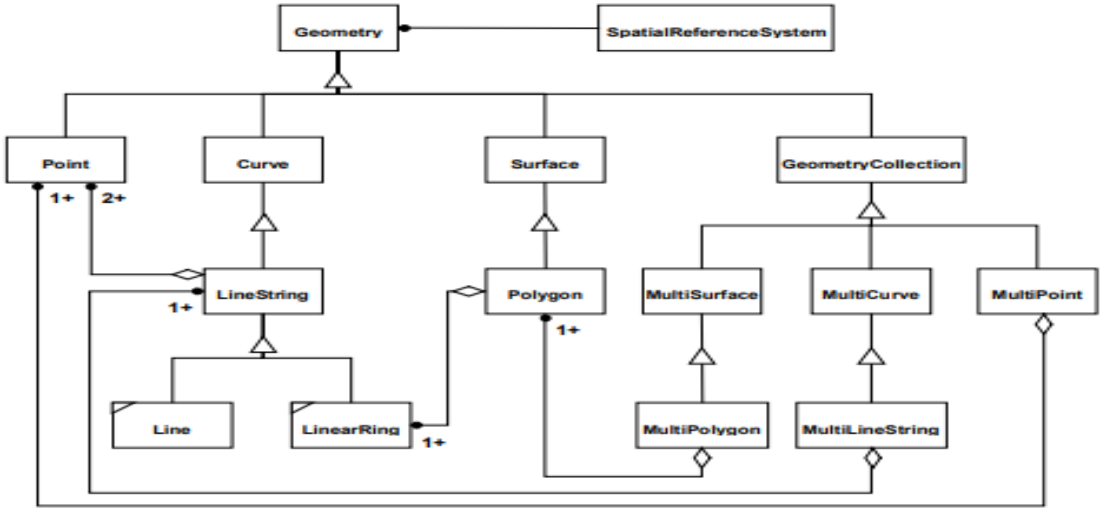

그림 1. OGC Simple Feautres Specification 공간 데이터 타입
Fig. 1. OGC Simple Feautres Specification Spatial Data Type

표 1. OGC Simple Feautres Specification 공간 연산자
Table 1. OGC Simple Feautres Specification Spatial Operator

구분	연산자 형식	반환 타입
관계 연산	Equals(g1 Geometry, g2 Geometry)	integer
	Disjoint(g1 Geometry, g2 Geometry)	integer
	Touches(g1 Geometry, g2 Geometry)	integer
	Within(g1 Geometry, g2 Geometry)	integer
	Overlaps(g1 Geometry, g2 Geometry)	integer
	Crosses(g1 Geometry, g2 Geometry)	integer
	Intersects(g1 Geometry, g2 Geometry)	integer
	Contains(g1 Geometry, g2 Geometry)	integer
분석 연산	Intersection(g1 Geometry, g2 Geometry)	Geometry
	Difference(g1 Geometry, g2 Geometry)	Geometry
	Union(g1 Geometry, g2 Geometry)	Geometry
	SymDifference(g1 Geometry, g2 Geometry)	Geometry
	Buffer(g1 Geometry, d Double Precision)	Geometry
	ConvexHull(g1 Geometry)	Geometry

표 1에서 보는 바와 같이 공간 연산자는 관계 연산자와 분석 연산자로 나뉘어진다. Equals는 g1과 g2가 동일한 영역인지를 비교하고, Disjoint는 g1과 g2가 다르면 참을 반환한다. Touches는 g1과 g2가 경계면이 맞 닿아 있으면 참을 반환한다, Within은 g1이 g2를 포함하고 있는가를 판별하고, Contains는 g1이 g2에 포함되는지를 나타낸다. Crosses는 g1과 g2가 교차 하는 가를 나타내고 Intersects는 겹치는 부분이 있는지를 나타낸다. Contains는 g1이 g2에 포함 되는지 여부를 나타낸다.Ovelaps는 g1, g2가 겹치는 부분이 있는지를 반환한다. 분석 연산자의 Intersection은 g1, g2가 겹치는 부분을 반환한다. Difference는 g1에서 g2의 겹친 영역을 제외한 영역을 반환하고, Union은 g1, g2 두 개의 폴리곤을 합친 영역을 반환한다. SymDifference는 Union으로 구한 영역에서 Intersection으로 구한 영역을 뺀 여집합과 같은 연산자이다. Buffer는 공간영역을 확대하거나 축소시킬 때 사용하며, ConvexHull은 최소 둘레를 갖는 폴리곤을 구한다.

OGC의 Simple Feature Specification은 2차원 공간에서 필요한 다양한 연산자를 정의하고 있다.

3. JTS Library

JTS(JAVA Topology Suite)[9] 라이브러리는 vividsolution에서 제공하는 오픈 소스 공간 라이브러리이다. 2차원 데카르트 평명상에서 선형 지오메트리를 처리하기 위한 알고리즘을 완성시켰으며, OGC Simple Feature Specification의 모든 관계 연산자와 분석 연산자를 구현하였다.

벡터 기반 지리 정보 소프트웨어의 핵심 구성 요소로 설계 되었으며, 기하학에서 사용할 수 있는 범용 알고리즘을 제공한다.

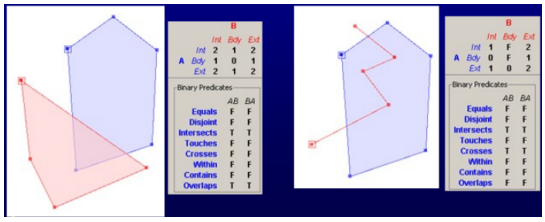

그림 2. JTS 라이브러리 관계 연산자
Fig. 2. JTS Library Relation Operator

그림 2에서 보는 바와 같이 JTS 라이브러리의 관계 연산자는 두 객체 사이의 관계를 정의한다. 공각 객체를 지원하는 경량 데이터 베이스에서는 BBOX(Boundary Box)와 Circle 연산자만을 관계 연산자로 지원하는 것에 비하여 다양한 관계를 정의할 수 있게 구현되었다.

Ⅲ. 시스템 설계

본 장에서는 본 논문에서 제안하는 Spatial DB를 위한 공간 연산자 연구의 설계 및 구현에 대하여 설명한다.

1. 시스템 설계

본 논문에서 제안하는 시스템은 다음 그림 3과 같다.

그림 3에서 보는 바와 같이 본 논문의 아키텍처는 4개의 주요 모듈로 구성되어진다.

공간데이터를 MongoDB 내부에 공간 데이터를 저장하고 이를 데이터통신 모듈로 통해 공간데이터를 서버로 가져오거나 질의 결과를 저장한다.

데이터관리 모듈은 MongoDB로부터 불러온 공간 데이터를 서버의 메모리에 저장하고 GeoJson형태의 공간 데이터 타입을 WKT 형태로 변환한다.

질의관리 모듈은 클라이언트로부터의 질의를 분석하고 실행계획을 세우며, 공간 연산자 및 타입을 관리한다.

질의 처리 모듈은 질의 관리 모듈의 실행 계획에 따라 공간 질의를 수행하고 질의 결과를 MongoDB 또는 클라이언트에게 전송하는 역할을 한다.

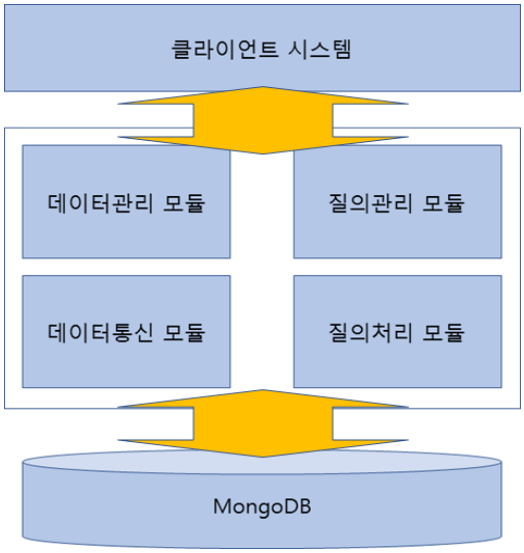

그림 3. Spatial MongoDB 아키텍처
Fig. 3. Spatial MongoDB 아키텍처

Ⅳ. 시스템 구현

본 시스템은 Ubuntu 16.0.4 환경에서 구현 하였으며, 구현 환경은 CPU I5-7500, RAM 16GB에서 구현하였다. MongoDB는 3.8.1 버전을 사용하였으며, Vividsolution의 JTS 라이브러리를 사용하여 공간연산을 처리하였다. 실험에 사용한 데이터는 국토교통부 국토지리정보원[10]에서 제공한 대한민국의 강과 철도의 좌표를 사용하였다.

실험에 사용된 데이터는 우리나라의 20개(740KB)의 철도 좌표와 총 1016개(80MB)의 강 좌표를 이용하였다.

실험에 사용된 두 가지 데이터는 선형 데이터이므로 포함 관계를 가지고 있지 않기 때문에 교차에 대한 연산인 Crosses 연산을 이용하였다.

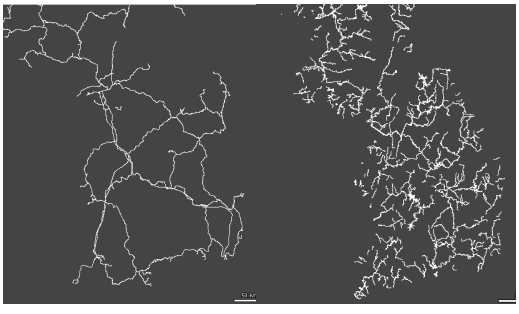

그림 4. 실험데이터의 LineString
Fig. 4. Line String of Experimental data

그림 4에서 보는 바와 같이 실험에 사용되는 강과 철도의 데이터는 LineString의 형태로 나타나며, 철도의 데이터는 20개의 노선을 나타내고 있으며, 강의 데이터는 약 1020개의 강을 나타내고 있다. 실험 데이터의 범위는 북한을 포함하고 북한 북쪽의 두만강과 압록강과 연결된 중국과 연해주 쪽의 지류도 포함되어 있다.

```
2018-08-31T13:22:29.442+0900 [clientcursormon]  mapped (incl journal view):320
2018-08-31T13:22:29.442+0900 [clientcursormon]  connections:0
2018-08-31T13:24:04.725+0900 [initandlisten] connection accepted from 127.0.0.1:
33542 #1 (1 connection now open)
2018-08-31T13:24:04.751+0900 [initandlisten] connection accepted from 127.0.0.1:
33544 #2 (2 connections now open)
2018-08-31T13:24:05.277+0900 [conn2] end connection 127.0.0.1:33544 (1 connectio
n now open)
2018-08-31T13:24:05.277+0900 [conn1] end connection 127.0.0.1:33542 (1 connectio
n now open)
```

그림 5. 데이터통신 모듈
Fig. 5. Data Communication Module

그림 5에서 보는 바와 같이 데이터통신 모듈은 서버에서 MongoDB에 접속을 할 수 있게 해주며 MongoDB의 자원을 효율적으로 관리하기 위한 pool 관리를 돕는다. 또한 질의 결과를 MongoDB에 반영하는 역할을 해준다.

```
public static Geometry geojson2Geometry(DBCursor cursor) throws ParseExcep
    Geometry g = null;
    WKTReader wktReader = new WKTReader();
    String parse = cursor.next().get("geometry").toString();
    String[] s = parse.split(" : ");
    String front = s[1].split(",")[0].toUpperCase().substring(1, s[1].spli
    if(front!="LINESTRING"){
        g=wktReader.read("LINESTRING (0 0, 1 1)");
        return g;
    }
    s[2] = s[2].substring(0, s[2].length()-3);
    String rear[] = s[2].split(",");
    for(int j=0; j < rear.length; j++){
        if(j%2 != 0){
            if(rear.length - 1 == j){
                rear[j] = rear[j].substring(0, rear[j].length()-1);
            }else{
                rear[j] = rear[j].substring(0, rear[j].length()-1) + ", ";
            }
        }else{
            rear[j] = rear[j].substring(2, rear[j].length()-1);
        }
    }
    String coord = "";
    for(int j=0; j < rear.length; j++){
        coord += rear[j];
    }
    String full = front+" (" + coord + ")";
    try{
    g = wktReader.read(full);
    }catch(Exception e){
        System.out.println(parse);
    }
    return g;
}
```

그림 6. 질의관리 모듈
Fig. 6. Query Managing Module

그림 6에서 보는 바와 같이 질의 관리 모듈은 읽어들인 MongoDB의 데이터가 GeoJson의 형태이므로 이를 처리하기 위해서 WKT좌표로 변환하여 질의 처리 모듈에 전달한다.

```
int intersect = 0;
int fullcount = 0;
while(cursor1.hasNext()){
    g = geojson2Geometry(cursor1);
    DBCollection coll2 = db.getCollection("river");
    DBCursor cursor2 = coll2.find();
    while(cursor2.hasNext()){
        g2 = geojson2Geometry(cursor2);
        if (g.crosses(g2)==true){
            System.out.println(g.toString());
            System.out.println(g2.toString());
            intersect++;
        }
        fullcount++;
    }
}
```

그림 7. 질의처리 모듈
Fig. 7. Query Processing Module

그림 7에서 보는 바와 같이 질의 처리 모듈은 질의 관리 모듈에서 전달 받은 공간객체를 공간 연산 함수를 이용하여 처리하고 그에 따른 결과를 반환한다. 그림 .은 Crosses 연산을 하는 과정을 보여주고 있다.

데이터관리 모듈은 객체를 저장하는 역할을 하며, 시스템에서 처리된 데이터를 클라이언트로 보내는 역할을 한다. 본 논문에서는 질의를 처리하기 위한 결과까지 구현을 하였으며, 추후에 다른 서비스와 연동할 계획이다.

실험 결과는 다음과 그림 8과 같다.

```
LINESTRING (1238038.945299999 2504931.7377, 1238039.075 2504932.4963, 1238039.457 2
LINESTRING (1263392.246 2481355.280099999, 1263361.24 2481367.0624, 1263299.228 248
20320
11
```

그림 8. 실험 결과
Fig. 8. Experiment result

그림 8에서 보는 바와 같이 Nested loop를 통해 Crosses 연산을 한 결과 총 20320 번의 비교 연산을 하였고, 11개의 교차 구간이 있음을 확인하였다. 수행 시간은 평균 89.2 초가 소요되었다.

연산에 대한 결과는 데이터의 양이 많음으로 전수 검사를 할 수 없었으며, 샘플 검사를 하였을 때 100%의 정확도를 보였다.

그러나 연산량에 비해 소요시간이 많이 소모된 점은 공간객체에 대하여 정확히 비교하기 위해서는 각 선분에 대한 선형 방정식을 수행하여야하기 때문으로 추정된다.

$$x = x_1 + s(x_2 - x_1) \quad (1)$$
$$y = y_1 + s(y_2 - y_1)$$

식(1)의 x, y 해가 폐구간에 존재하면 교차 구간이 존재한다고 할 수 있기 때문이다. 즉, 위 방정식을 각 객체 내에 모든 선분과 비교를 해야 알 수 있기 때문에 연산 속도가 느릴 수 밖에 없다.

따라서 속도 문제를 개선하기 위해서는 인덱스를 통해 검색 범위를 좁혀야 한다는 것을 알 수 있었다.

Ⅳ. 결론

본 논문에서는 NoSQL 중 대표적인 MongoDB에서 공간연산이 부족한 점을 극복하기 위하여 서버에서 이를 처리하여 사용자에게 제공하기 위한 시스템을 설계 및 구현하였다.

기존의 MongoDB에서 지원하지 않는 다양한 연산을 처리할 수 있음을 확인하였으며, 이를 통해 다양한 서비스를 제공할 것으로 기대된다.

그러나 이를 좀더 효율적으로 활용하기 위해서는 공간인덱스와 같은 다양한 기능을 필요로 하다. 따라서 본 연구 과제의 향후 과제로 공간인덱스에 대해 연구하고 공간 뿐만 아니라 시공간에 대해서 추가적인 연구를 하고자 한다.

References

[1] Jing Han, Haihong E, Guan Le, and Jian Du, “Survey on NoSQL Database,” Conf. of Pervasive computing and applications, 2011 6th international conference on IEEE, pp. 363-366, 2011.
DOI: https://dx.doi.org/10.1109/icpca.2011.6106531

[2] Maximilian Walther and Michael Kaisser, “Geo-spatial Event Detection in the Twitter Stream,” Conf. of European conference on information retrieval, pp. 356-367, 2013.
DOI: https://doi.org/10.1007/978-3-642-36973-5_30

[3] Jae-Young Chang, “An Experimental Evaluation of Box office Revenue Prediction through Social Bigdata Analysis and Machine Learning,” Journal of The Institute of Internet, Broadcasting and Communication, Vol. 17, No. 3, pp. 167-173, Jun, 2017.
DOI: https://doi.org/10.7236/JIIBC.2017.17.3.167

[4] MongoDB, http://www.mongodb.com

[5] Sarthak Agarwal, K. S. Rajan, “Performance analysis of MongoDB versus PostGIS/PostGreSQL databases for line intersection and point containment spatial queries,” IEEE International Conference on Cluster Computing Workshops, pp. 32-40, Sept, 2012
DOI: https://doi.org/10.1007/s41324-016-0059-1

[6] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain,

Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff and Raghotham Murthy, “An implementation approach to store GIS spatial data on NoSQL database,” Conf. of Geoinformatics International Conference on IEEE, pp. 1-5, 2014. DOI: https://doi.org/10.1109/geoinformatics.2014.6950846

[7] Longgang Xiang, Juntao Huang, Xiaotian Shao and Dehao Wang, “A MongoDB-Based Management of Planar Spatial Data with a Flattened R-Tree,” Journal of International Geo-Information, Vol. 5, No . 7, pp. 119, Nov 2016. DOI: 10.3390/ijgi5070119

[8] Open GIS Consortium, “OpenGIS Simple Features Specificaion For SQL Revision 1.1,” OpenGIS Project Document, May, 1999.

[9] JTS Library, http://www.vividsolutions.com

[10] National Geographic Information Institute, http://ngii.go.kr

저자 소개

곽 광 진(준회원)

• Kwang Jin Kwak received his MS in computer Science at Konkuk University in 2010 and 2016, He is currently a studying at Korea Polytech University for a doctor’s course in Department of Smart Manufacturing Engineering. His intersts GIS, Information Retrieval, Text Mining, Database, NoSQL, etc.

윤 하 영(준회원)

• Ha-Young Youn is currently studying at Korea Polytech University for a bachelor's degree in computer engineering in 2018. His research interests include the Database systems, Big Data, Data Analysis, etc.

신 동 윤(준회원)

• Dong-Youn Shin is currently studying at Korea Polytech University for a and is attending computer engineering. His research interests include Big Data, Data Processing, etc.

신 동 진(준회원)

• Dong-Jin Shin received his BS in Enginerring at Korea Polytechnic University in 2018. He is currently a Master's course in the department of Smart Manufacturing Engineering at Korea Polytechnic University. His research interests include Big Data, Internet of Things(IoT), Network Security, etc.

박 정 민(정회원)

• Jeong-Min Park received his BS in Computer Science at Korea Polytechnic University in 2003. He received his MS and PhD in at SungKyunKwan University in 2005 and 2009, respectively. He is currently a professor at the department of Computer Science at Korea Polytechnic University. His research interests include Cyber Physical System(CPS), Autonomic Computing, Software Engineering, etc.

김 정 준(정회원)

• Jeong-Joon Kim received his BS and MS in Computer Science at Konkuk University in 2003 and 2005, respectively. In 2010, he received his PhD in at Konkuk University. He is currently a professor at the department of Computer Science at Korea Polytechnic University. His research interests include Database Systems, BigData, Semantic Web, Geographic Information Systems (GIS) and Ubiquitous Sensor Network (USN), etc.

※ 이 성과는 2018년 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2017R1A2B4011243).