The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 6, pp.1-9, Dec. 31, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.6.1
JIIBC 2018-6-1

IoT 환경에서 모바일 기반 빅데이터 처리 및 모니터링 기술

Mobile-based Big Data Processing and Monitoring Technology in IoT Environment

이승해[*], 김주호[**], 신동윤[***], 신동진[****], 박정민[*****], 김정준[******]

Seung-Hae Lee[*], Ju-Ho Kim[**], Dong-Youn Shin[***], Dong-Jin Shin[****], Jeong-Min Park[*****], Jeong-Joon Kim[******]

요 약 현재 이슈가 되고 있는 4차 산업혁명에서 다양한 빅데이터 기술들을 통하여 기존의 느린 속도 보다 빠른 분석 결과를 즉각적으로 받아 볼 수 있고, 모바일과 웹에서 실시간 모니터링을 하는 연구를 진행하였다. 먼저 IoT 기기인 Raspberry Pi를 이용하여 다양한 비정형 센서 데이터를 생성하고 센서 데이터를 실시간 수집하고, 수집한 데이터를 여러 개의 노드를 이용해 분산 저장한 뒤 저장된 센서 데이터를 가공, 정제 처리하여 분석 모델 및 알고리즘을 통해 분석 결과를 시각화하여 출력한다. 이러한 방법들을 이용한 진행은 IoT를 이용한 빅데이터 및 모바일 관련 분야에서 필요한 고급 인력을 양성 및 데이터를 효율적이고 빠르게 처리할 수 있다. 또한, 실시간 모니터링을 통하여 연구결과의 신뢰성을 확인할 수 있는 정보를 제공하고자 한다.

Abstract In the fourth industrial revolution, which has become an issue now, we have been able to receive instant analysis results faster than the existing slow speed through various Big Data technologies, and to conduct real-time monitoring on mobile and web. First, various irregular sensor Data is generated using IoT device, Raspberry Pi. Sensor Data is collected in real time, and the collected data is distributed and stored using several nodes. Then, the stored Sensor Data is processed and refined. Visualize and output the analysis result after analysis. By using these methods, we can train the human resources required for Big Data and mobile related fields using IoT, and process data efficiently and quickly. We also provide information that can confirm the reliability of research results through real time monitoring.

Key Words : IoT, Big Data, Mobile, Sensor Data, Visualization, Monitoring

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

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

Ⅰ. 서 론

현재 우리는 “빅데이터”라는 기술이 주가 되어 있는 스마트 시대에 살고 있다. 이 “빅데이터”는 작은 단위의 데이터로는 진행할 수 없었던 데이터의 새로운 발견이나 여러 가지 형태의 가치를 큰 단위의 데이터를 활용해 추출 해내는 기술이다. 기존에도 많은 양의 데이터가 존재하였지만, 특히 “빅데이터”라고 칭해 부르는 이유는 날이 갈수록 발전해 가고 있는 기술과 함께 데이터의 양이 무섭게 증가하고 있기 때문이다.

유튜브(Youtube), 페이스북(Facebook), 카카오톡(KakaoTalk), 트위터(Twitter)와 같이 우리가 생활하며 사용하는 모든 것들이 데이터가 된다. 유튜브 동영상을 시청하고, 페이스북 타임라인을 통하여 친구들의 상태를 확인하여, 카카오톡을 통해 실시간으로 대화하는 모든 것들이 데이터를 통해 이루어지고 저장된다. 이러한 데이터뿐만 아니라 사물에 센서를 부착해 실시간으로 데이터를 인터넷으로 주고받는 기술인 사물인터넷, 즉 IoT 기술은 쏟아지는 실시간 데이터를 생성하여 빅데이터 기술을 이용하면 저장, 관리, 분석할 수 있기 때문에 빅데이터 기술이 중요해지고 있다. 이렇게 센서로 수집한 데이터를 비정형 데이터라고 하며, 기존의 비정형 센서 수집, 처리, 분석방법이 수동적이라는 단점이 있다.

그 사례로 보일러 스위치에서 온도와 습도를 직관적인 온도와 습도의 상태만 볼 수 있었다. 하지만 본 연구에서는 현 상태만 볼 수 있었던 수동적이라는 단점을 개선하기 위해 IoT 기기인 Raspberry Pi를 이용하여서 실시간으로 센서 데이터를 생성하고 빅데이터 기술을 이용해 처리, 분석할 수 있는 방법을 제안한다[1,2].

제안하는 방법은 총 5단계로 1) IoT 기기 Raspberry Pi를 이용하여 다양한 비정형 센서 데이터 생성, 2) 센서 데이터 실시간 수집, 3) 여러 개의 노드를 이용한 분산 저장, 4) 저장된 센서 데이터 가공, 정제, 5) 가공, 정제된 결과를 시각화를 수행한다. 제안 사항들을 통해 온도와 습도의 직관적이고, 수동적인 정보만이 아니라 온도와 습도의 실시간 정보를 동시에 비교하여, 실시간 그래프를 통해 5단계의 방법의 유효성을 입증시킨다. 본 논문은 다음과 같이 구성된다. 2장에서는 관련 기술, 3장에서는 시스템 설계, 4장에서는 시스템 구현, 5장에서는 결론을 기술한다.

Ⅱ. 관련 연구

1. 빅데이터(Big Data)

빅데이터란 말 그대로 많은 양의 데이터를 말한다. 오늘날 데이터는 1년 단위로 그 양이 제타바이트 단위로 2배씩 증가하며, 이전에 저장할 수 없었던 데이터를 저장하고 많은 양의 데이터를 다양한 처리도구를 이용하여 분석한다. 빅데이터의 다양한 기술과 5단계의 수집, 저장, 처리, 분석, 시각화 단계로 사용하고 빠른 속도와 비용적으로 우수한 성능으로 구현이 가능한 기술이다[3,4,5].

2. HTML

HTML은 HyperText Markup Language의 약자로 웹 페이지를 위한 지배적인 마크업 언어이다. HTML은 여러 태그로 구성되어 있으며, 각 태그를 사용하여 원하는 형태의 웹을 구현할 수 있다. 인터넷 문서는 Hyper Text의 원리를 이용하고, 여러 문서를 링크시켜 다양한 정보를 손쉽게 검색하여 볼 수 있도록 만들어 준다. 이런 특성들과 쉬운 사용법들이 HTML이 대중화되는 기반이 되었다[6].

3. Java

Java는 Sun Microsystems가 1995년 처음 출시한 프로그래밍 언어이자 컴퓨팅 플랫폼으로, 수 많은 응용 프로그램 및 웹 사이트가 Java를 설치하지 않으면 작동되지 않는다. Java는 데이터 센터, 게임, 슈퍼컴퓨터, 휴대폰, 인터넷에 이르기까지 다양한 곳 어디에서나 기술이 존재하며 빠르고 안전한 기술이다[7].

4. AJAX

AJAX는 Asynchronos Javascript And XML의 약자로 HTML form 태그가 아니라 자바 스크립트를 통해서 서버에 데이터를 요청하는 것이다. 따라서 AJAX를 이용하면 서버에 로드되어있는 데이터를 페이지에 보여주기 위해 새로운 HTML 페이지로 갈 필요도 없고 새로고침을 할 필요가 없다. 필요한 부분만 로딩되고, 속도가 빠른 기술이다[8].

Ⅲ. 시스템 설계

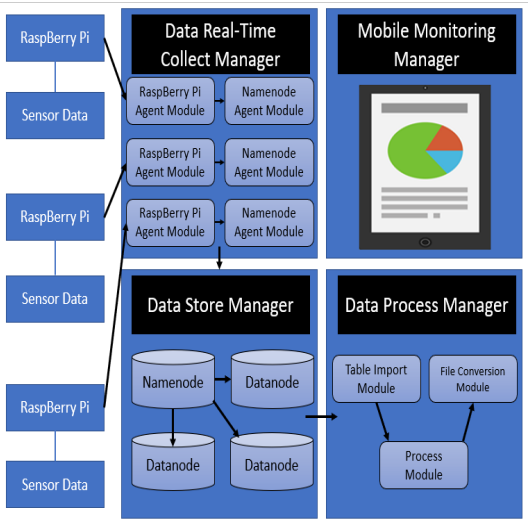

그림 1. IoT 환경에서 모바일 기반 빅데이터 처리 및 모니터링 시스템 개념도

Fig. 1. Mobile-based Big Data Processing and Monitoring System Conceptual Diagram in IoT Environment

그림 1은 빅데이터를 이용한 IoT환경에서 모바일 기반 처리 및 모니터링 시스템의 개념도를 보여준다. IoT 환경에서 모바일 기반 빅데이터 처리 및 모니터링 시스템은 센서 데이터를 생성하는 'RaspBerry Pi' , 발생되는 센서 데이터를 실시간으로 수집하는 'Data Real-Time Collect Manager', 수집된 데이터를 분산 저장하는 'Data Store Manager', 분산 저장된 데이터를 처리하는 'Data Process Manager', 처리된 데이터를 바탕으로 모바일에서 모니터링 할 수 있는 'Mobile Monitoring Manager' 4가지로 구성된다.

Data Real-Time Collect Manager는 IoT 기기인 'RaspBerry Pi'에서 발생하는 다양한 비정형 센서 데이터(온도, 습도, 초음파, …)를 실시간으로 수집하는 매니저이다.

Data Store Manager는 'Data Real-Time Collect Manager' 부분에서 수집되는 센서 데이터의 원본을 하둡을 이용하여 분산 저장하게 되며, 1개의 네임노드(Namenode)와 3개의 데이터노드(Datanode)를 가지고 완전 분산 모드를 구성한다.

Data Process Manager는 저장된 센서 데이터를 빅데이터의 처리 솔루션인 Hive에서 테이블로 사용하기 위한 'Table Import Module', 입력된 테이블에 불필요한 부분을 가공, 정제하는 'Process Module', 최종 분석을 위해 파일로 내보내는 'File Conversion Module'로 구성된다.

Mobile Monitoring Manager는 'Data Process Manager'의 처리된 결과를 기반으로 모바일 어플리케이션을 통해 그래프 등의 시각화 기술로 모니터링해주는 매니저이다.

Ⅳ. 시스템 구현

본 논문의 목적은 IoT 환경에서 모바일 기반 빅데이터 처리 및 모니터링 시스템을 구현하여 효율적인 시스템 관리 기능을 개발하는 것으로 관리자가 어디서든 모바일 어플리케이션을 이용하여 모니터링 함으로써 비정형 센서 데이터를 실시간으로 수집, 처리, 분석할 수 있는 시스템을 개발하는 것이다.

1. Data Real-Time Collect Manager

실시간으로 RaspBerry Pi에서 측정되는 센서 값에 대한 수집은 Flume을 이용하였다.

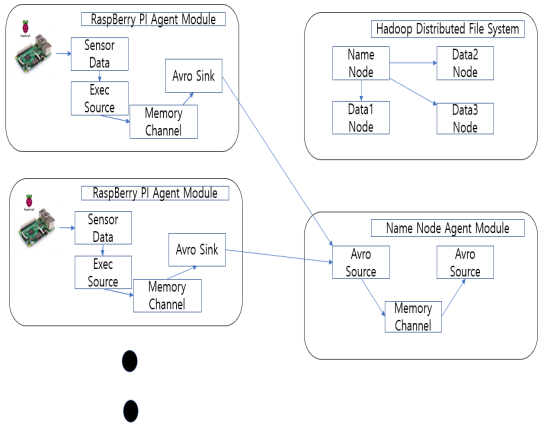

그림 2. Flume을 이용한 로그 수집 과정

Fig. 2. Log collection process using Flume

그림 2는 데이터가 생성되고 수집되는 과정을 표현한 아키텍처로 RaspBerry Pi의 센서를 통하여 데이터를 수집하고, 수집된 데이터는 로그수집 도구인 Flume을 사용하여 NameNode와 통신하여 데이터를 NameNode의 지정된 위치에 저장한다.

```
import Adafruit_DHT as Ris
import time
import csv
f = open('data.csv','w')
try:
 while True:
  wr = csv.writer(f)
  h, t = Ris.read_retry(Ris.DHT11,4)
wr.writerow(["{0:0.1f}c".format(t,h),"{1:0.1f}%".format(t,h)])
time.sleep(1)
except KeyboardInterrupt:
 f.close()
 print "Quit"
```

그림 3. RaspBerry Pi 코드
Fig. 3. Raspberry Pi cord

센서 데이터를 수집하기에 앞서 RaspBerry Pi의 센서에서 발생하는 데이터는 그림 3과 같은 Python 언어를 사용하여 수집한다. 그리고 csv와 time 라이브러리를 삽입하고, data.csv 파일을 생성하여 데이터를 쓴다. 데이터는 반복문을 통하여 온도, 습도를 1초 간격으로 측정하여 저장해 주고, 수집을 종료할 시에는 Quit를 화면에 표시하고 종료된다.

```
pi@raspberrypi:~/test_tem $ ls
Rtem.py
pi@raspberrypi:~/test_tem $ nano Rtem.py
pi@raspberrypi:~/test_tem $ python Rtem.py
^CQuit
pi@raspberrypi:~/test_tem $ ls
data.csv  Rtem.py
pi@raspberrypi:~/test_tem $
```

그림 4. data.csv 파일 확인 경로
Fig. 4. Data.csv File Verification Path

그림 3에서 작성한 Python 코드를 터미널에서 실행하여 데이터를 수집하고 일정 시간이 지난 후 수집을 종료한 후 해당 폴더에서 데이터를 확인해 보면 data.csv가 생성된 것을 확인할 수 있다.

센서 데이터가 수집되는 각각의 RaspBerry Pi에 Flume을 설치하고, 다음과 같이 Conf파일을 설정한다.

```
Rising02.sources = RisingSrc
Rising02.channels = RisingChannel
Rising02.sinks = Risingout
# Sources
Rising02.sources.execGenSrc.type = exec
Rising02.sources.execGenSrc.command =
tail -F /home/risingsunz/Downloads/data.txt
Rising02.sources.execGenSrc.batchSize = 10
Rising2.sources.execGenSrc.channels=memoryChannel
# Channels
Rising02.channels.memoryChannel.type = memory
Rising02.channels.memoryChannel.capacity = 100000
Rising02.channels.memoryChannel.transactionCapacity =
10000
# Sinks
Rising02.sinks.avroSink.type = avro
Rising02.sinks.avroSink.hostname = 172.16.0.1
Rising02.sinks.avroSink.port = 33333
Rising02.sinks.avroSink.batch-size = 1000
Rising02.sinks.avroSink.channel = memoryChannel
```

그림 5. RaspBerry Pi의 Flume 설정 명령어
Fig. 5. Flume setting command of RaspBerry Pi

flume의 구성에서 Source는 외부에서 이벤트를 입력받는 부분으로 RaspBerry Pi의 각 센서에서 측정되는 데이터는 데이터가 발생할 때마다 데이터가 추가되기 때문에 센서의 데이터 값들이 저장되는 경로에서 리눅스의 tail 명령어를 통하여 수집하고 한 번에 Source에 이벤트를 전달한다. Channel은 이벤트를 임시로 저장하여 Sink로 전달하는 역할을 하며 이벤트 크기는 100000개로 지정하고 트랜잭션의 크기는 10000개로 설정하였다. 마지막으로 Sinks는 이벤트를 외부로 출력하는 부분으로 NameNode로 전송되는 부분이기 때문에 전송받는 NameNode의 IP주소와 임의의 Port 주소 33333을 한 번에 받을 수 있는 이벤트의 수를 1000개로 지정하였다.

```
bin/flume-ng Rising - c conf/ --conf-file conf/sample.conf
-name Rising02 - Dflume.root.logger=INFO,console
```

그림 6. RaspBerry Pi의 Flume 실행
Fig. 6. Run Flume on RaspBerry Pi

그림 6은 그림 5의 Flume 설정파일인 Conf 디렉터리 아래의 sample.conf를 실행하여 NameNode로 전송해 주는 역할을 하는 명령어이다.

```
Rising01.sources = RisingSrc
Rising01.channels = RisingChannel
Rising01.sinks = HDFS
#source
Rising01.sources.avroGenSrc.type = avro
Rising01.sources.avroGenSrc.bind = 172.16.0.1
Rising01.sources.avroGenSrc.port = 33333
Rising01.sources.avroGenSrc.channels = memoryChannel
#channel
Rising01.channels.memoryChannel.type = memory
Rising01.channels.memoryChannel.capacity = 100000
#sink
Rising01.sinks.HDFS.type = HDFS
Rising01.sinks.HDFS.hdfs.path = hdfs://master:9000/sensor
Rising01.sinks.HDFS.hdfs.fileType = DataStream
Rising01.sinks.HDFS.hdfs.writeFormat = text
Rising01.sinks.HDFS.hdfs.batchSize = 1000
Rising01.sinks.HDFS.hdfs.rollSize = 0
Rising01.sinks.HDFS.hdfs.rollCount = 10000
Rising01.sinks.HDFS.hdfs.rollInterval = 1200
Rising01.sinks.HDFS.hdfs.useLocalTimeStamp = true
Rising01.sinks.HDFS.channel = memoryChannel
```

그림 7. NameNode 의 Flume 설정 명령어
Fig. 7. NameNode Flume Setup Command

2. Data Store Manager

RaspBerry Pi의 Sink를 통해서 출력되는 이벤트는 NameNode Flume 구조에서 Source 부분이 받게 된다. 이벤트를 수신받을 IP와 Port를 지정해 주고 사용하는 Channel의 타입은 RaspBerry Pi와 같이 메모리로 구현하였다. Sink에서는 NameNode에 분산 저장하기 위해 type을 HDFS로 지정해 주었고 파일이 저장될 경로를 지정하였다. 그리고 HDFS에 쌓이게 되는 센서 파일의 형식은 Text, 이벤트 수의 제한은 1000개, 파일을 나누는 기준은 20분으로 설정하였다.

```
bin/flume-ng Rising  -c conf/ --conf-file conf/flume.conf
--name Rising01  -Dflume.root.logger=INFO,console
```

그림 8. NameNode 의 Flume 설정 명령어
Fig. 8. NameNode Flume Setup Command

그림 8은 그림 7에서 환경 설정한 Flume의 설정파일인 Conf 디렉터리의 flume.conf 파일을 실행하여 HDFS 내로 파일을 실시간으로 분산 저장하는 명령어다.

```
FO - org.apache.flume.sink.hdfs.BucketWriter.open(BucketWriter.java:261)] C
reating hdfs://master:9000/sensor/FlumeData.1531811653682.tmp
2018-07-17 16:16:13,840 (hdfs-HDFS-roll-timer-0) [INFO - org.apache.flume.s
ink.hdfs.BucketWriter.close(BucketWriter.java:409)] Closing hdfs://master:9
000/sensor/FlumeData.1531811653682.tmp
2018-07-17 16:16:13,840 (hdfs-HDFS-call-runner-4) [INFO - org.apache.flume.
sink.hdfs.BucketWriter$3.call(BucketWriter.java:339)] Close tries increment
ed
2018-07-17 16:16:13,861 (hdfs-HDFS-call-runner-5) [INFO - org.apache.flume.
sink.hdfs.BucketWriter$8.call(BucketWriter.java:669)] Renaming hdfs://maste
r:9000/sensor/FlumeData.1531811653682.tmp to hdfs://master:9000/sensor/Flum
eData.1531811653682
```

그림 9. Flume을 통한 데이터 저장 과정
Fig. 9. Process of storing data through Flume

그림 9와 같이 RaspBerry Pi와 NameNode의 통신이 되면 Creating, Closing, Renaming 단계를 거쳐 HDFS의 지정된 위치에 데이터가 수집되는 것을 확인할 수 있다. Creating 단계에서는 데이터가 수집되는 과정에서 생기는 임시파일 즉 .tmp 파일을 생성하고 데이터의 수집이 완료되면 Closing 단계를 거쳐 Renaming 단계에서 .tmp 파일이 아닌 그림 7에서 지정한 text 형태로 데이터가 저장된다.

```
hadoop@hadoop-name:~$ hadoop fs -cat /sensor/FlumeData.1531811529648
18/07/17 16:20:24 WARN util.NativeCodeLoader: Unable to load native-hadoop
library for your platform... using builtin-java classes where applicable
2018-06-06 20:37:00,35C,88%
2018-06-06 20:37:01,35C,66%
2018-06-06 20:37:02,35C,76%
2018-06-06 20:37:03,35C,57%
2018-06-06 20:37:04,35C,50%
2018-06-06 20:37:05,35C,50%
2018-06-06 20:37:06,35C,50%
```

그림 10. 분산 저장된 데이터 저장 확인
Fig. 10. Confirming Saved Data Saved

그림 10은 수집된 로그 데이터를 cat 명령어로 확인한 그림이다. 수집된 로그 데이터의 형식은 “시간, 온도, 습도”로 구성되어있으며, 실시간으로 수집되는 센서 데이터는 처리 프로세스 단계를 거쳐 온도의 단위와 습도의 단위를 통일하는 작업을 진행한다.

```
create table lab (
time timestamp,
temperature string,
humidity string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘,’
STORED AS TEXTFILE
LOCATION ‘/sensor’;
```

그림 11. 데이터 처리 쿼리를 발생할 테이블 생성
Fig. 11. Create a table that will cause Data processing queries

3. Data Process Manager

hive 쉘을 실행시킨 후 그림 11처럼 HiveQL을 이용하여 HDFS의 /sensor 디렉터리에 저장된 데이터를 ‘,’단위로 분할하여 불러와 create table명령어를 사용하여 time, temperature, humidity 컬럼을 테이블에 생성한다.

```
INSERT OVERWRITE DIRECTORY ‘/lab3’
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘,’
select regexp_replace(time, “2018-06-06 ”, “”),
regexp_replace(temperature, “\C”, “”),
regexp_replace(humidity, “\%”, “”) from lab;
```

그림 12. 생성된 테이블에 대한 데이터 처리 쿼리 발생
Fig. 12. Generate Data processing query for generated table

이후에 그림 11에서 생성된 테이블에 대해 데이터 처리 쿼리를 발생시키게 되는데 테이블 생성 시 명시해 놓은 time, temperature, humidity 칼럼에 대해 regexp_replace를 사용하여 각 칼럼에서 다른 문자나 형식으로 대체하고 싶은 패턴을 입력한다. 본 그림에서는 공백으로 패턴을 대체하여 특수문자들을 제거하는 처리를 진행하게 된다. 그 후 HDFS 공간의 /lab3라는 디렉터리에 결과 파일을 저장하게 된다.

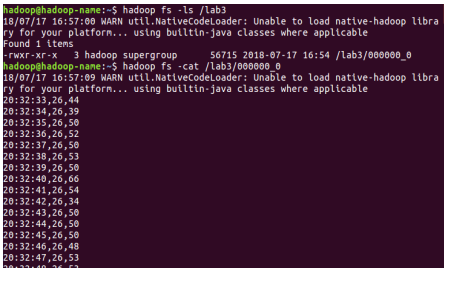

그림 13. HDFS내 /lab3에 저장된 결과파일 확인
Fig. 13. Check the results file stored in HDFS / lab3

```
for((;;)); do
hive -f ~/lab.hive
done
```

그림 14. Hive처리 프로세스 반복 쉘 스크립트
Fig. 14. Hive processing process iterative shell script

그림 13까지 모든 처리 과정을 완료하고 HDFS 내 /lab3 디렉터리를 확인하면 결과 파일 000000_0이 생성된 것을 확인할 수 있는데 cat 명령어로 파일을 열어보면 Hive에서 쿼리를 실행했던 것과 같이 년도, C, % 와 같은 패턴들이 공백으로 대체된 것을 확인할 수 있다.

Flume은 사용자가 지정한 실시간으로 데이터가 HDFS에 저장되기 때문에 데이터가 저장되는 대로 Hive의 처리도 지속적해서 이루어져야 한다. 따라서 Hive의 처리 프로세스를 쉘 스크립트 안에서 for 문을 사용하여 계속해서 반복되는 식으로 구성하였다.

```
create external table lab (
time timestamp,
temperature string,
humidity string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/sensor';
INSERT OVERWRITE DIRECTORY '/lab3'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
select regexp_replace(time, "2018-06-06 ", ""),
regexp_replace(temperature, "\C", ""),
regexp_replace(humidity, "\%", "") from lab;
drop table lab;
```

그림 15. 처리쿼리를 발생시킬 lab.hive
Fig. 15. Invoke the processing query with lab.hive

그림 14는 Hive 처리 스크립트가 반복되는 쉘 스크립트이며 그림 15는 쉘 스크립트 안에 반복되는 Hive처리 과정을 기술한 lab.hive 파일이다.

4. Mobile Monitoring Manager

처리 및 정제가 완료된 데이터를 실시간으로 Web, Mobile Application 환경에서 보여주기 위해서 Apache Tomcat 서버와 Eclipse를 연동하여 서버 구동을 하였다. 구축된 서버를 통하여 HTML과 JavaScript를 사용하여

시각화한 결과를 Web, Mobile Application에서 확인할 수 있도록 배포한다.

위에서 설명한 대로 Window 운영체제 기반의 서버에서 Eclipse를 사용하여 구성한 HTML 파일을 실행시키게 되는데 HTML에서 CSS를 사용하여 기본적인 UI들을 구성하였고 그래프의 실시간 처리가 이루어져야 하기 때문에 서버 안에 저장되는 데이터들을 실시간으로 가져와야만 했다. 그래서 비동기적으로 서버와 클라이언트가 통신할 수 있는 Ajax를 사용하여 데이터들의 컬럼들을 배열처럼 선언해주고 배열의 인덱스 값을 사용하여 어떤 값들을 그래프로 표현할 것인지 명시하고 온도, 습도, 온도 + 습도의 3가지 그래프로 구성하고 1초에 해당하는 interval을 기재하였다. 또한 혹시 모를 경우를 대비하여 서버 자체에서도 실행 파일과 웹이 동기화될 수 있도록 자동으로 서버를 refresh 하도록 설정해 주었다. 결과적으로 시간 초에 따른 온도, 습도, 온도+습도의 그래프를 출력할 수 있게 되었다.

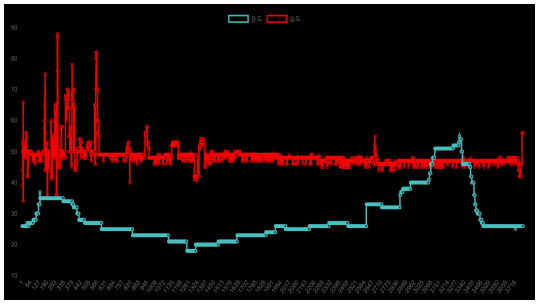

그림 16. 실시간 온도와 습도 그래프
Fig. 16. Real-time temperature and humidity graph

그림 16의 그래프는 RaspBerry Pi를 통해 온도데이터가 1초마다 수집되기 때문에 X축은 데이터의 순서가 되고 Y축은 각 데이터 값을 기준으로 최소~최대의 범주가 나타나게 된다. 파란 그래프는 온도를 나타내며 빨간 그래프는 습도를 나타내는데 둘의 반비례 관계를 한눈에 볼 수 있도록 구성하였고 온도가 높아졌을 때 반대로 습도는 낮아지는 반비례 결과를 시간별로 확인할 수 있었다.

애플리케이션 제작 프로그램인 Android Studio를 사용하여 애플리케이션을 제작하였다. 구현된 웹 홈페이지의 주소를 연동하여 웹 뷰 형식으로 제작하였고 웹 홈페이지를 제작할 때 그래프의 크기를 사용자가 사용하는 기기의 화면 크기에 맞게 자동으로 변화되게 제작하여 그래프의 크기는 스마트폰 화면에 맞춰져 있어 따로 설정하지 않았다.

그림 17. 어플리케이션 메인화면
Fig. 17. Application main screen

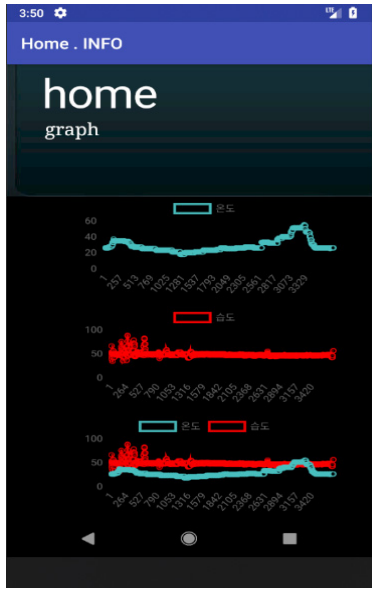

그림 18. 실시간 정보 페이지
Fig. 18. Real-time information page

그림 17은 애플리케이션의 메인 화면으로 실시간 정보를 확인하기 위한 버튼을 누르면 그림 18의 화면으로 전환된다. 그림 18에서는 시간에 따라 첫 번째 결과 그래프는 시간에 따른 온도의 변화량, 두 번째 결과 그래프는 시간에 따른 습도의 변화량을 나타내고 세 번째 결과 그래프는 온도와 습도의 변화량을 비교하는 그래프를 나타내는 것을 확인할 수 있다.

V. 결 론

본 논문에서는 기존의 비정형 센서 데이터 수집, 처리, 분석 방법이 수동적이라는 단점을 보완하기 위하여 1) IoT기기인 Raspberry Pi를 이용하여 다양한 비정형 센서 데이터 생성, 2) 센서 데이터 실시간 수집, 3) 여러 개의 노드를 이용한 분산 저장, 4) 저장된 센서 데이터 가공, 정제, 5) 가공, 정제 결과 시각화 단계들을 제안하였다. 제안 사항들을 통해서 기존 수동적인 과정의 문제점을 탐지하고 그것에 대응하는 전략 적용이 가능했고 빅데이터 시장에서 활용되는 서비스의 품질을 향상 시킬 수 있기 때문에 관련 기업들의 수익률 증대와 국가차원의 경제효과를 도모할 수 있고 4차 산업혁명에서 중요하게 여기어지는 IoT, AI, Cloud, Smart Factory 등 다양한 기술 분야에서 빅데이터, 모바일 관련 고급 인력을 양성할 수 있다. 그러나 서버의 사양이 부족하여 refresh를 진행 할 때 불안정한 모습을 보여 서버환경을 개선해야할 필요가 있고 비교 데이터의 양을 늘려볼 필요가 있다. 이것은 더 능동적인 빅데이터 처리과정을 진행해야 하는 목표와 다양한 분석을 목표로 두고 개선해 나아가면 현재 연구의 문제를 개선할 수 있다. 따라서 현재 연구와 개선사항의 연계를 통해 한정된 데이터에 국한되어 있는 것이 아닌 다양한 데이터를 통해 연구가 진행되도록 향후 연구에서 다룰 예정이다.

References

[1] Dong-Hak Kim, Pu-Sik Park, Byung-Seo Kim, “IoT-based IOS Application to Improve Heating and Cooling Satisfaction Level of Urban Railway Passenger,” The Journal of The Institute of Internet, Broadcasting and Communication, Vol.18 No.1, 2018.2, 1-8
DOI: https://doi.org/10.7236/JIIBC.2018.18.1.1

[2] Byeong-ho Cho, “Analysis and Design of Fruit e-Commerce System based on IoT,” The Journal of The Institute of Internet, Broadcasting and Communication, Vol.18 No.3, 2018.6, 135-141
DOI: https://doi.org/10.7236/JIIBC.2018.18.3.135

[3] Jae-Young Chang, “An Experimental Evaluation of Box office Revenue Prediction through Social BigData Analysis and Machine Learning,” The Journal of The Institute of Internet, Broadcasting and Communication, Vol.17 No.3, 2017.6, 167-173
DOI: https://doi.org/10.7236/JIIBC.2017.17.3.167

[4] Soon-duck Yoo, “A Study on Blockchain Ecosystem,” The Journal of The Institute of Internet, Broadcasting and Communication, Vol.18 No.2, 2018.4, 1-9
DOI: https://doi.org/10.7236/JIIBC.2018.18.2.1

[5] Sung-Sam Hong, Myung-Mook Han, “The Efficient Method of Parallel Genetic Algorithm using MapReduce of Big Data,” Journal of Korean Institute of Intelligent Systems, Vol.23 No.5, 2013.12, 385-391
DOI: http://dx.doi.org/10.5391/JKIIS.2013.23.5.385

[6] Hyun-Tack Seok, “Hybrid Web App Development for Eye movement at Mobile Devices,” The Journal of The Institute of Internet, Broadcasting and Communication, Vol.13 No.6, 2013.12, 249-254
DOI: http://dx.doi.org/10.7236/JIIBC.2013.13.6.249

[7] Chang-hun O, Yong-hui Cheon, “Implementation and Characteristics Analysis of Java-based Software Streaming Technology,” Journal of Korean Institute of Information Technology, Vol.11 No.1, 2013.1, 205-216
DOI: 10.14801/kiitr.2013.11.1.205

[8] Hyo-sang Kwon, Oh Yang, “The Implementation of the Solar Inverter Monitoring System using an AJAX,” Journal of the Korea Institute of Information and Communication Engineering, Vol.16 No.9, 2012.9, 1915-1922
DOI: http://dx.doi.org/10.6109/jkiice.2012.16.9.1915

Acknowledgement : This research was supported by X-mind Corps program of National Research Foundation of Korea(NRF) funded by the Ministry of Science and ICT(2017H1D8A1032103)

저자 소개

이 승 해(준회원)

• Seung-Hae Lee is currently a student at the Korea Institute of Industrial Technology. His research interests include Big Data, Data Processing, etc.

김 주 호(준회원)

• Ju-Ho Kim 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 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.