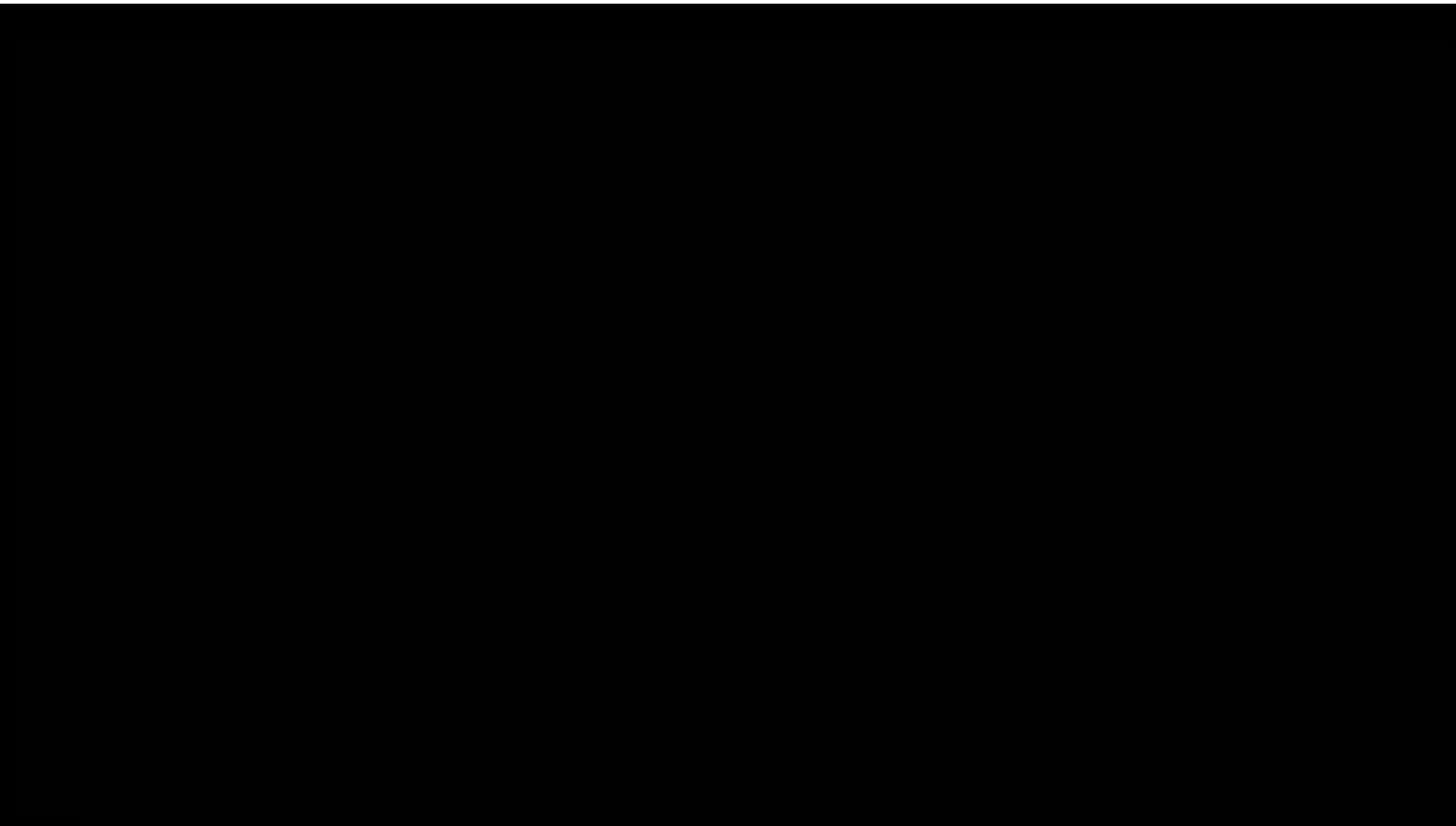


# 지진경보시스템 소개 및 이상징후 탐지 연구 소개

경북대학교 권영우

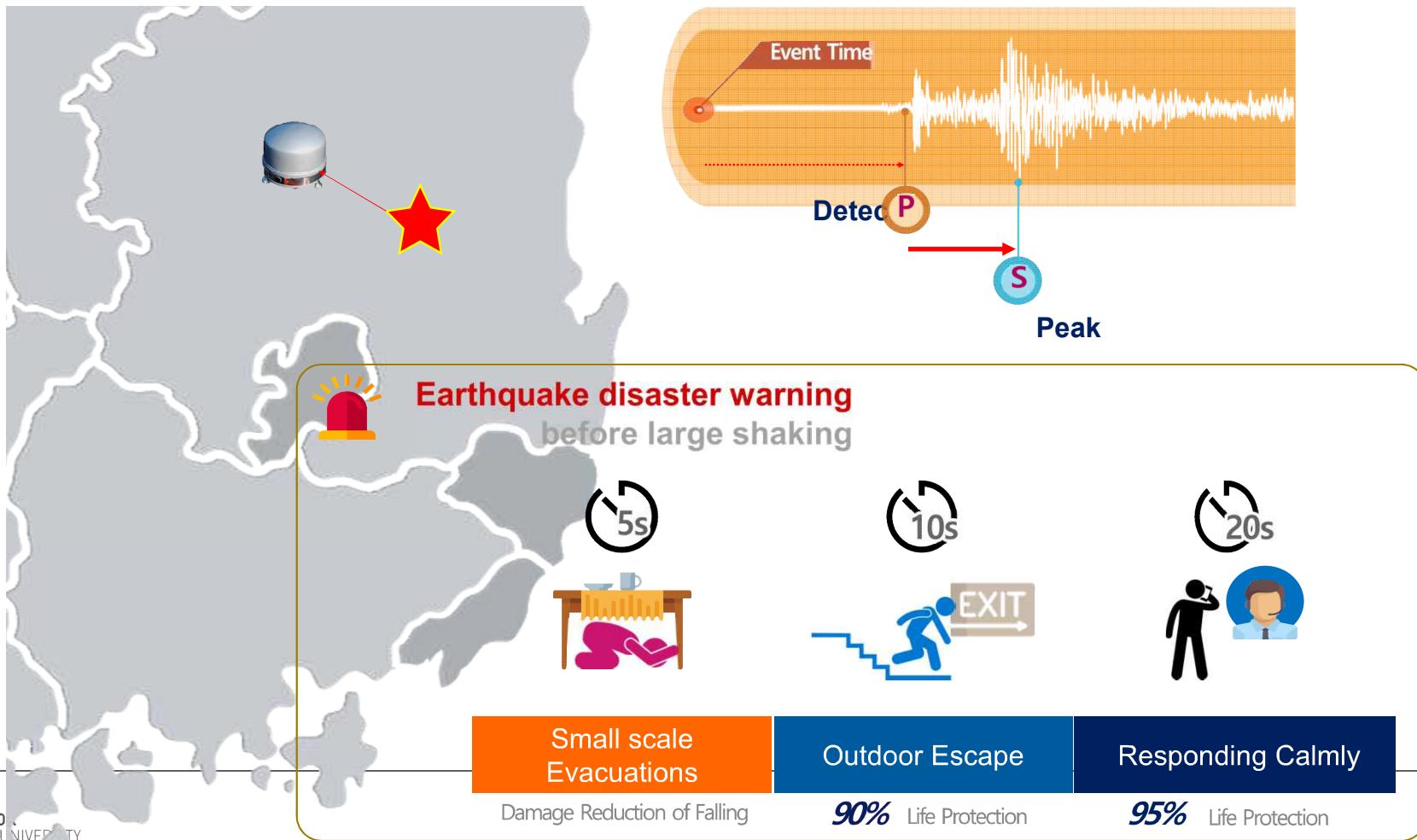
# 지진조기경보란?



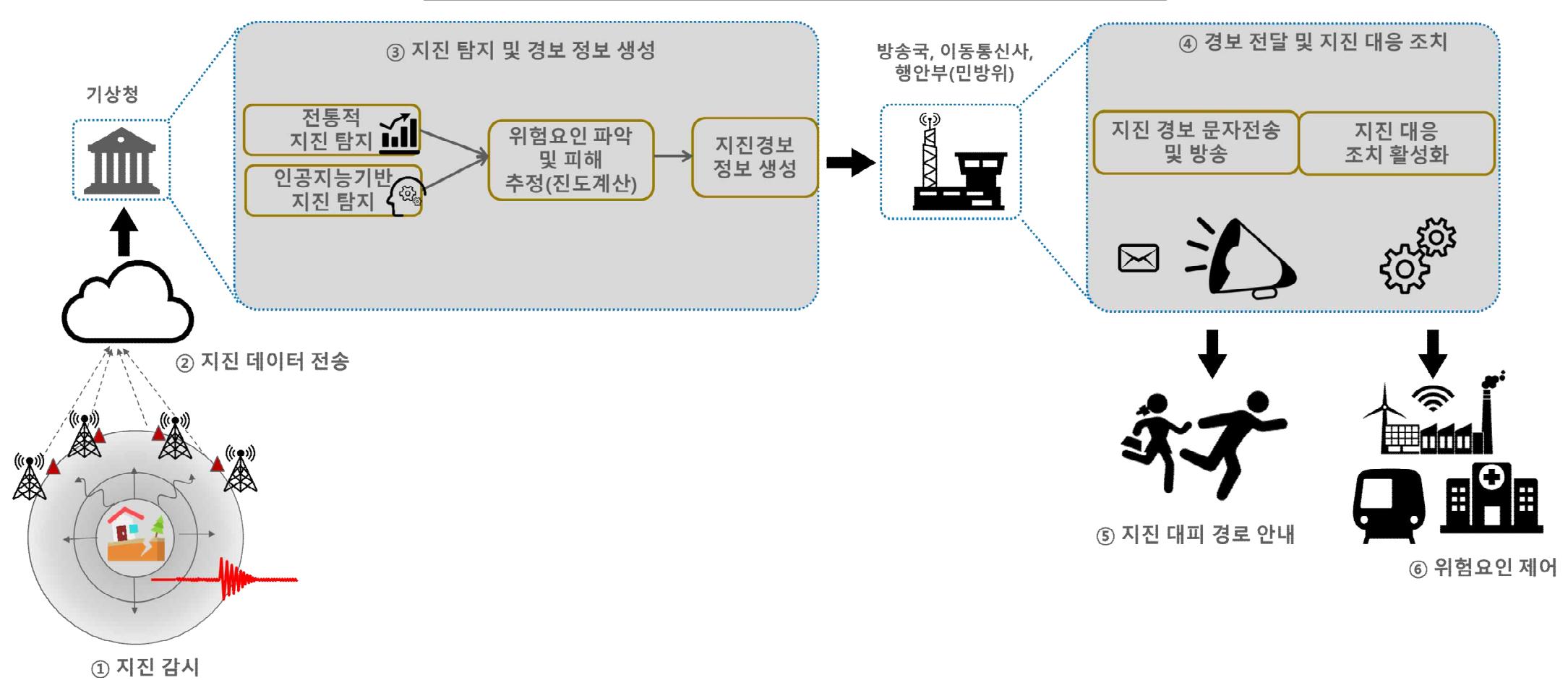


# 지진조기경보란?

## ▶ Earthquake Early Warning



# 지진조기경보 및 대응 시나리오



# 지진 감시 장치의 운영 환경

- 다양한 설치 환경에서 발생하는 외부 영향

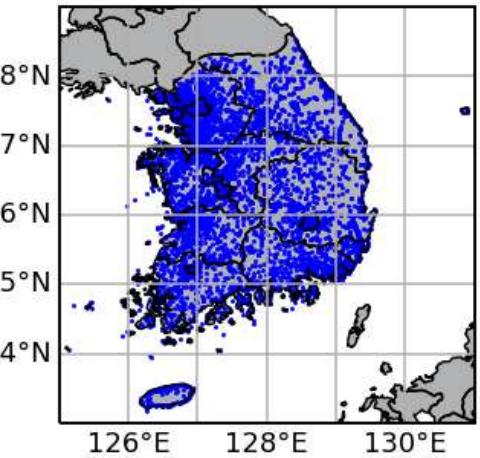
- 생활 환경 잡음



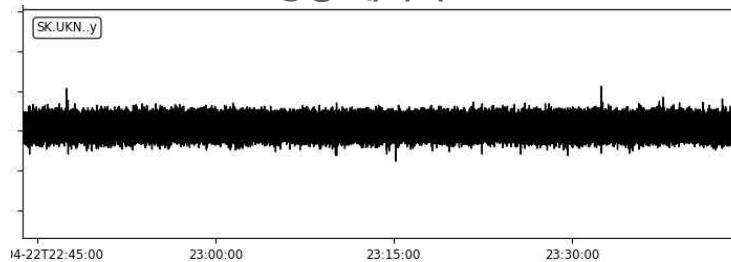
- 건물 고유의 진동



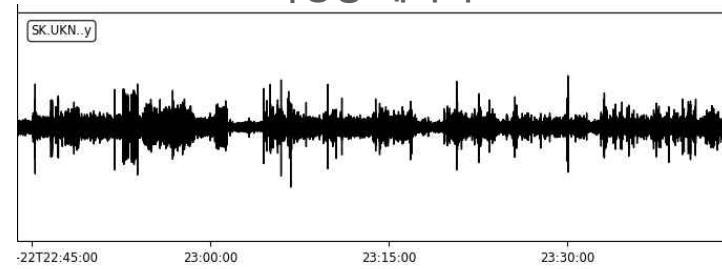
- 날씨로 인한 영향 (태풍)



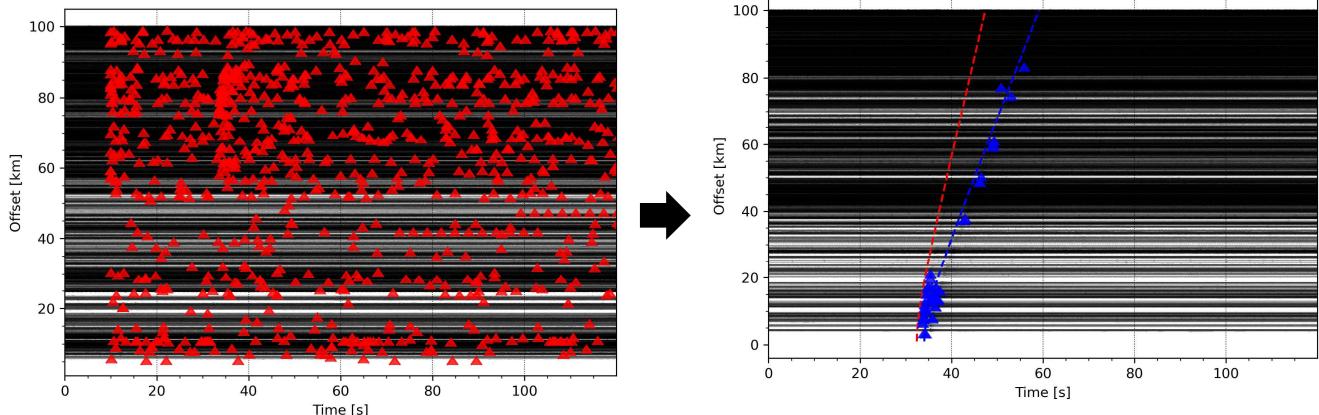
정상 데이터



비정상 데이터

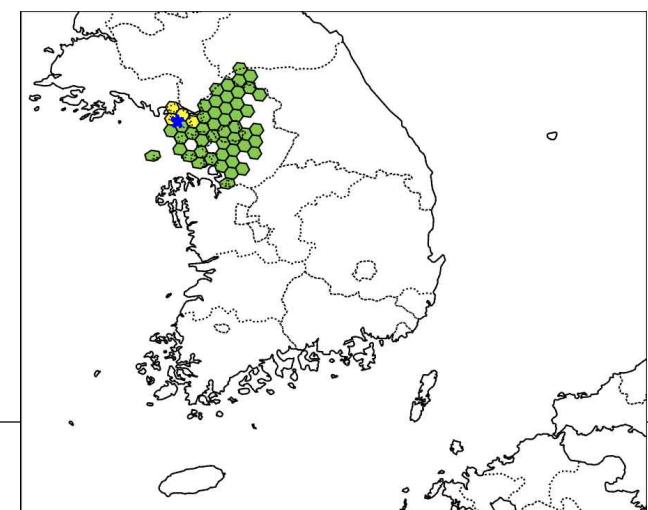
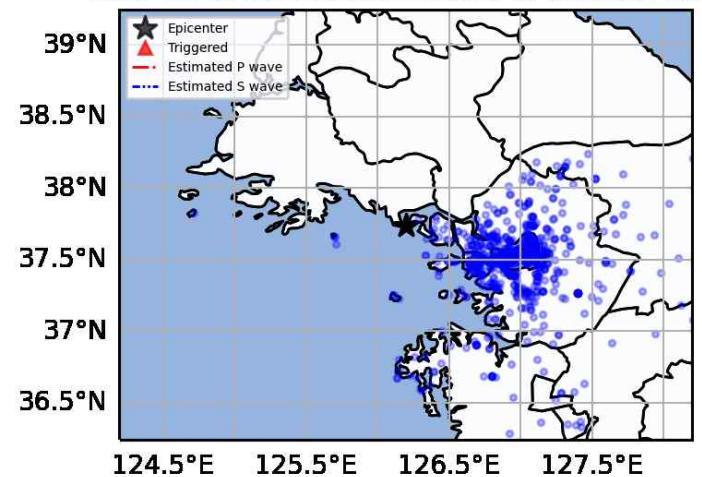


# 지진 (조기)경보 시스템의 목표

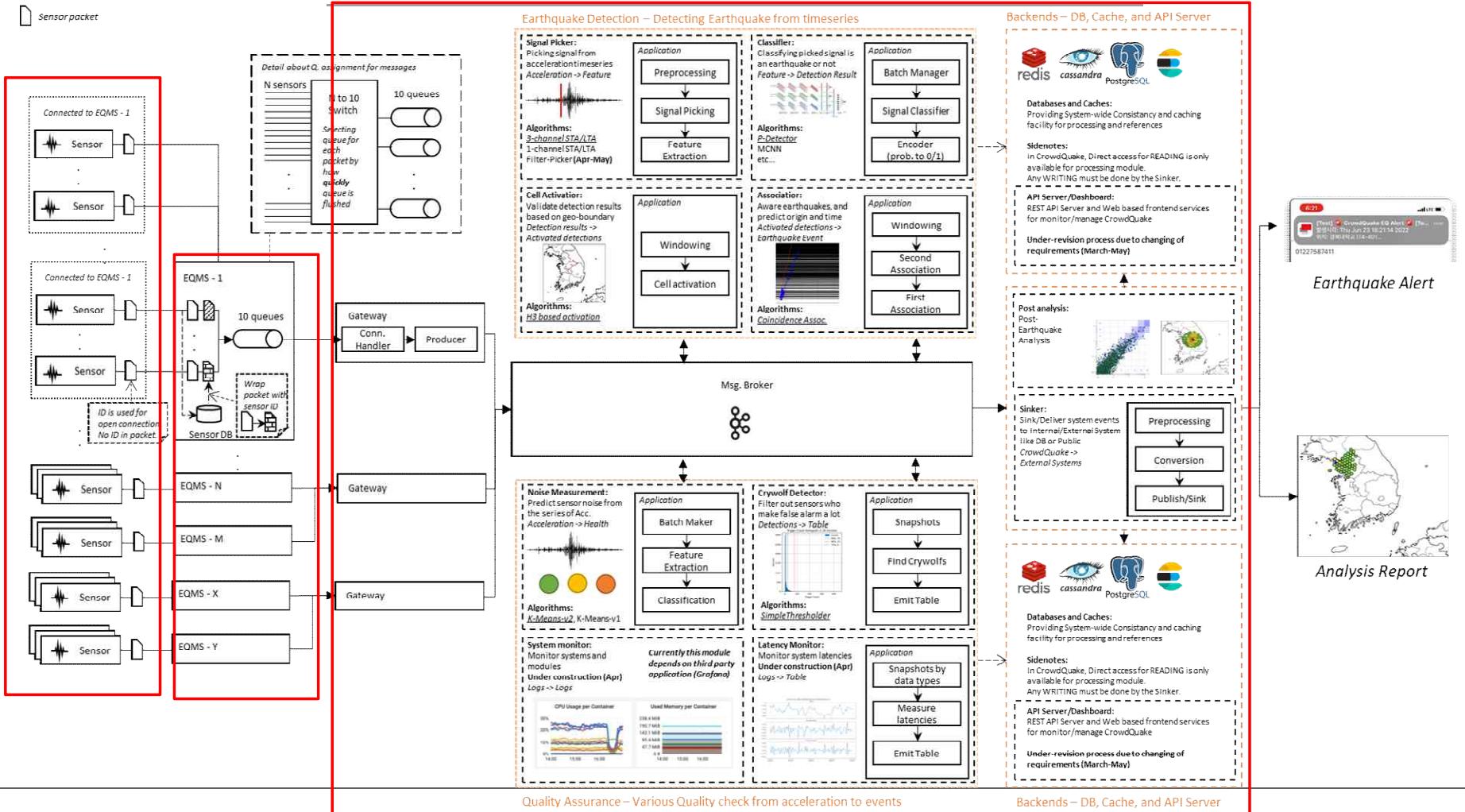


트리거 이벤트: 일정 크기 이상의 진동이 발생한 경우

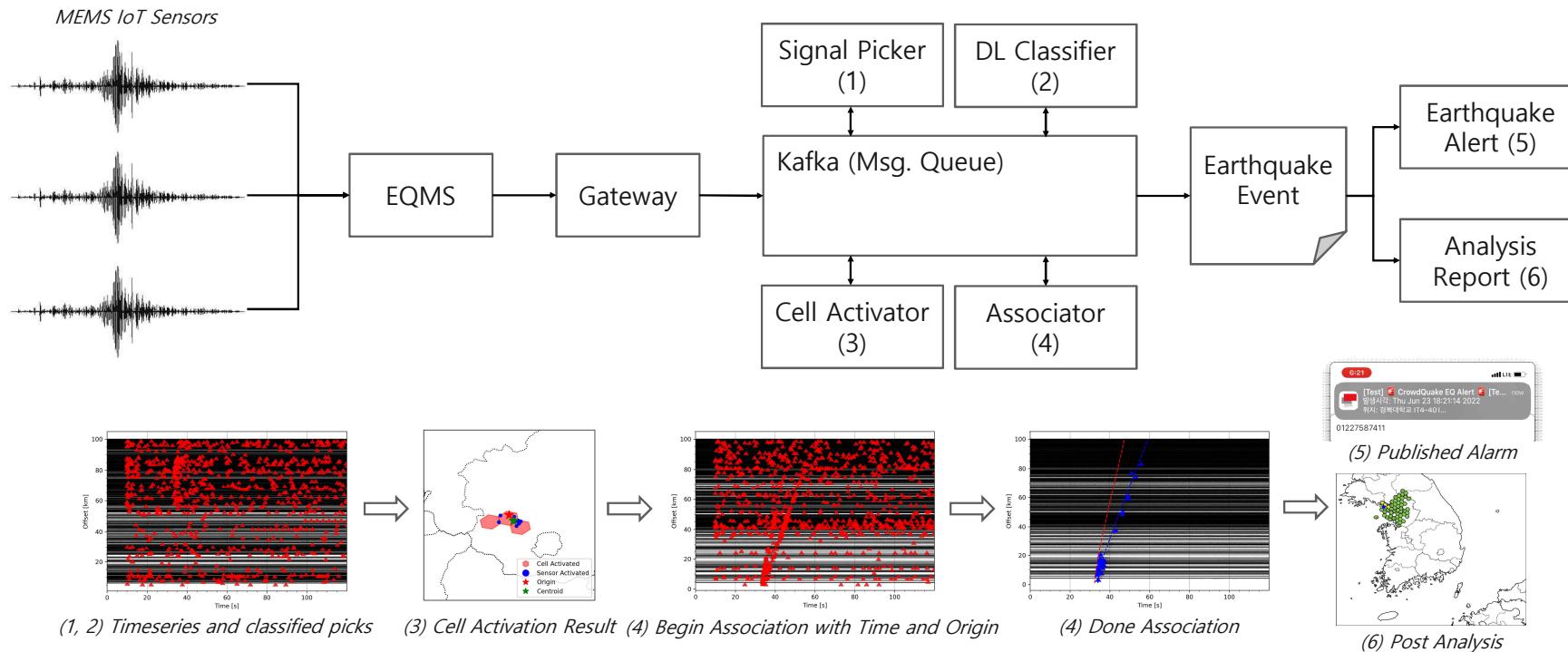
2023-01-08T16:28:15.000000Z, 00 seconds later



# 지진경보시스템(CrowdQuake) 및 테스트베드 소개



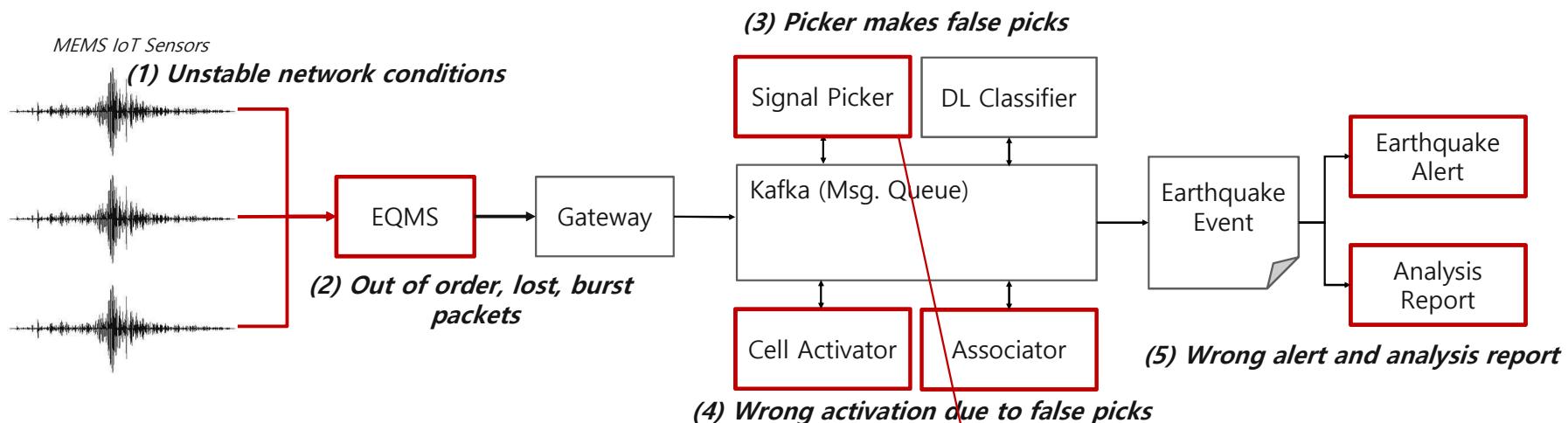
# 지진경보시스템 테스트베드 소개



# 지진경보시스템 테스트베드의 개발환경

- **지진 감시 센서**
  - 서로 다른 두 업체에서 HW/SW 별도 제작
  - C++으로 개발되었으며, 센서와 EQMS간의 데이터 전송 규약을 SKT에서 작성
- **EQMS (지진 데이터 수집 서버)**
  - SKT의 외주개발회사와 SKT 전자기술원에서 개발
  - Java로 개발되었으며, 8,000개의 센서로부터 SKT에 위치한 서버에서 수집 후 CrowdQuake로 전송
- **CrowdQuake (지진 데이터 분석 서버)**
  - 경북대에서 2018년부터 버전 1 ~ 3단계를 거쳐오며 개발
  - Python으로 개발되었으며 센서의 개수가 늘어남에 따라 (300개, 3,000개, 8,000개) 시스템 변화
  - 제 3의 라이브러리 및 프레임워크 활용이 많음

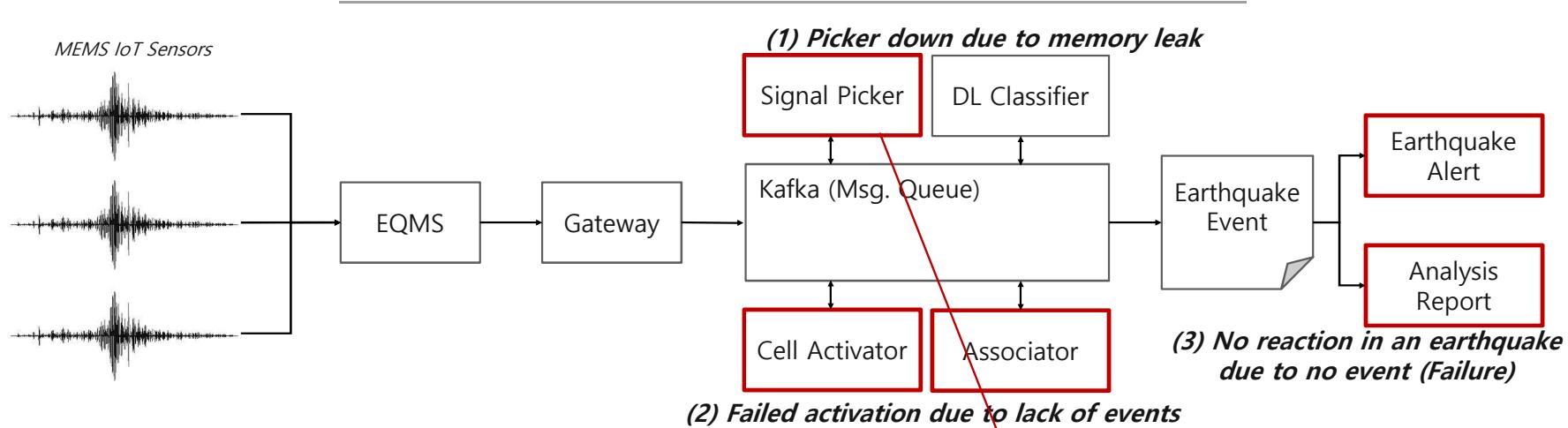
# 지진경보시스템에서 발생 가능한 오류



```
def _update_sta_lta(self, data) ->
List[TriggerResult]:
    results: List[TriggerResult] = []
    for i in range(data.shape[0]):
        sq = data[i] ** 2
        self._sta = self._csta * sq + self._icsta *
    self._sta
        self._lta = self._clta * sq + self._iclta *
    self._lta
```

잘못된 데이터 순서로 인한 트리거 값 계산 오류

# 지진경보시스템에서 발생 가능한 오류



```
def __call__(self, timestamp_msec: int, data: Any) -> List[Tuple[int, Any]]:
```

# Failure condition, if we are too far behind, we will drop the data

```
if timestamp_msec < self._expected_time_msec:
```

```
    return []
```

```
....
```

```
    if timestamp_msec > self._expected_time_msec +  
        self._hard_reset_msec:  
        self.reset(timestamp_msec)
```

```
....
```

```
# If data comes in order and there is no gap, return sorted data
```

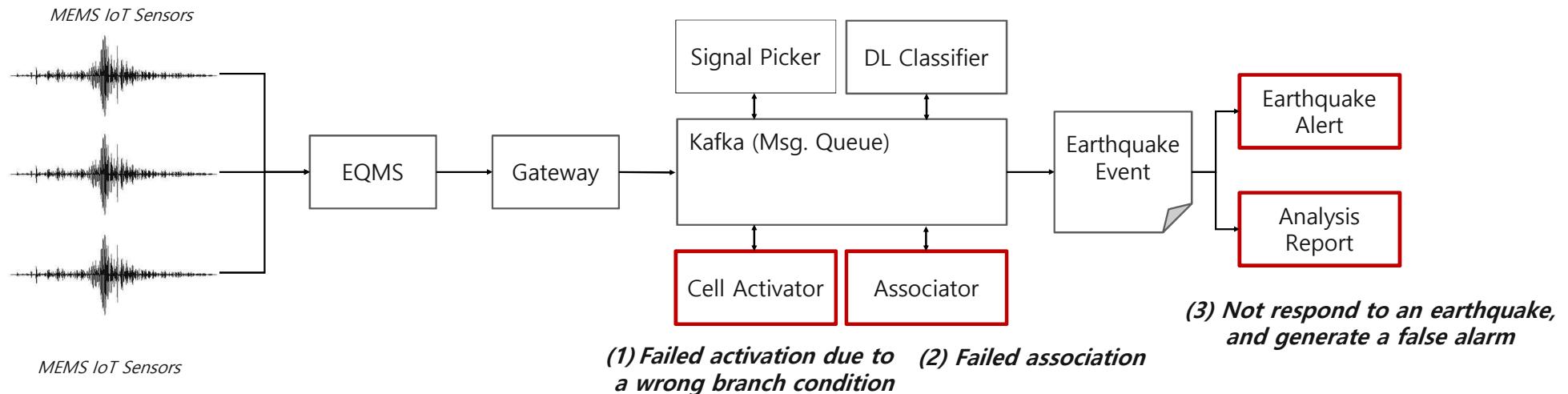
```
if self._expected_time_msec - self._zitter_msec <= W
```

```
    timestamp_msec <= W
```

```
    self._expected_time_msec + self._zitter_msec:
```

데이터 유효성 검사 부재로 인한 메모리 누수 발생

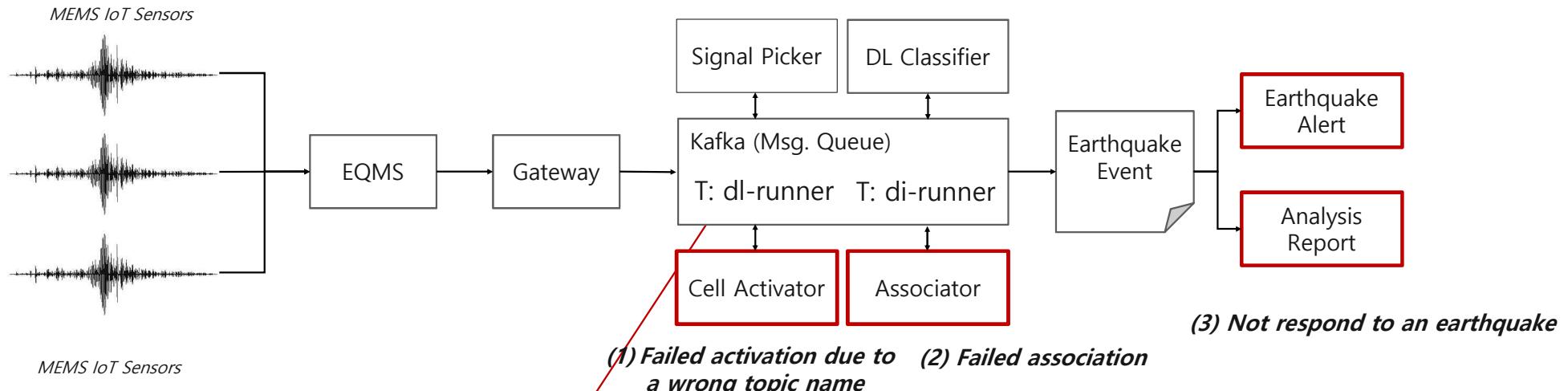
# 지진경보시스템에서 발생 가능한 오류



```
# Cell activation check
for hex_id, tr_count in grid_stat.items():
    num = sensors_in_grid(hex_id)
    ratio = tr_count / sensors_in_grid
    if num == 1:
        continue
    if num <= 2:
        activated.append(hex_id)
    elif ratio <= 0.25: # Intended >= 0.25
        activated.append(hex_id)
return activated
```

조건문 실수로 인한 지진 이벤트 미생성 오류 발생

# 지진경보시스템에서 발생 가능한 오류



```
TOPIC_CONSUME="di-runner"  
KAFKA_BOOTSTRAP=WRONG-IP
```

설정 오류로 인한 메시지 미수신

# 지진경보시스템에서의 오류 발견 연구 소개

한동대학교 홍신 교수님

# 분산 시스템에서의 이상징후 탐지

# 로그 기반 이상징후 탐지

- 로그 기반 이상징후 탐지의 문제

- 단순한 로그의 반복
- 다양한 형식의 로그
- 분석을 위하여 필요한 정보 부재 및 충분한 양의 로그 부족

- ✓ Timestamp, 함수 호출 스택, 예외 처리 내용 등
- ✓ 디버깅 목적의 변수 값 확인용 로그가 다수임
- ✓ 로그 출력 전 시스템 장애 발생하여 장애 로그가 없는 경우

- 로그와 오류 코드와의 거리 차이로 인하여 이상징후 원인 분석의 어려움

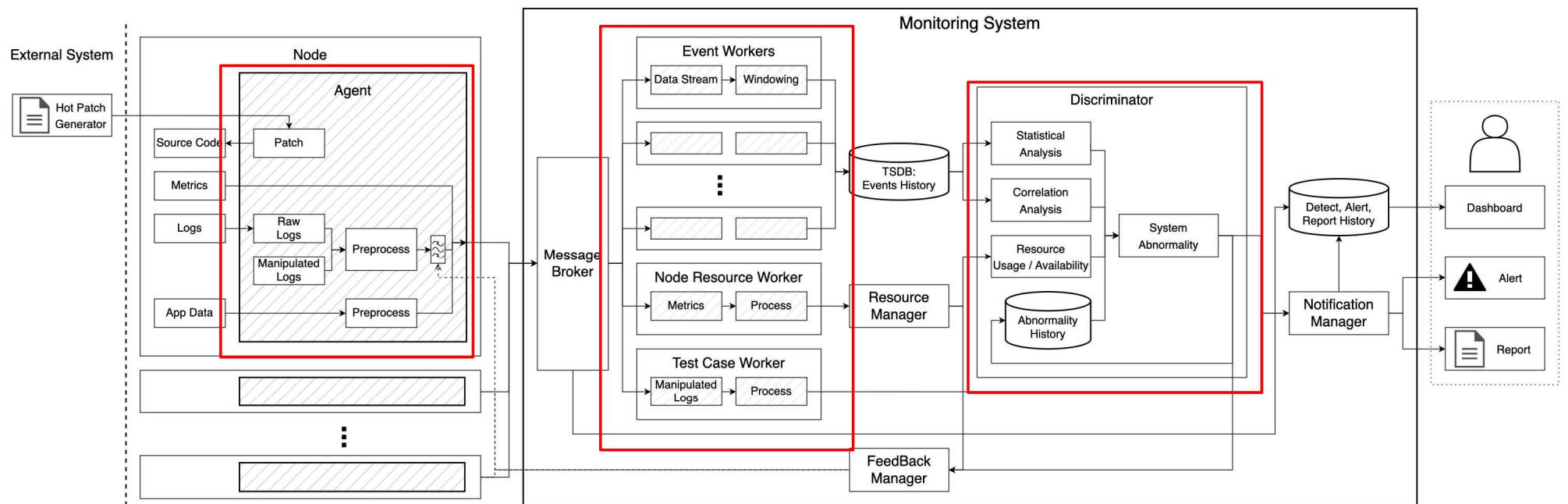
```
datacollector_1 | Message forwarded of size 713 :43:09.039 / 01227501105 / a1c602042212050143090000 / bytes: 7  
datacollector_1 | Message forwarded of size 713 14  
datacollector_1 | Message forwarded of size 713 eqms_sensor_sensor.1.ad8aqawa8dgz@hdc-node1 | 2022-12-05 01  
datacollector_1 | Message forwarded of size 713 :43:10.039 / 01227501105 / a1c502042212050143100000 / bytes: 7  
datacollector_1 | Message forwarded of size 713 13  
datacollector_1 | Message forwarded of size 713 eqms_sensor_sensor.1.ad8aqawa8dgz@hdc-node1 | 2022-12-05 01  
datacollector_1 | Message forwarded of size 713 :43:11.04 / 01227501105 / a1c02042212050143110000 / bytes: 71  
datacollector_1 | Message forwarded of size 713 4  
  
01227501104, 1670202374000, 1670169975125, 9  
01227501104, 1670202376000, 1670169977084, 9  
01227501104, 1670202379000, 1670169980119, 9  
01227501104, 1670202380000, 1670169981100, 9  
01227501104, 1670202384000, 1670169985101, 9  
01227501104, 1670202385000, 1670169986104, 9  
01227501104, 1670202387000, 1670169988114, 9  
01227501104, 1670202389000, 1670169990086, 9  
01227501104, 1670202394000, 1670169995093, 9  
01227501104, 1670202397000, 1670169998131, 9  
01227501104, 1670202411000, 1670170012097, 9  
01227501104, 1670202414000, 1670170015126, 9  
01227501104, 1670202421000, 1670170022097, 9  
01227501104, 1670202424000, 1670170025126, 9  
01227501104, 1670202425000, 1670170026095, 9  
01227501104, 1670202447000, 1670170048120, 9  
01227501104, 1670202460000, 1670170061123, 9
```

```
s=Hamburger,5_Cola,2_Chicken,10.5  
s=Rice,1.2_Chicken Soup,2.5  
s=Big Burger,1.2_Bone Soup,2.5  
s=Big Burger,1.2_Bone Soup,2.5  
s=Big Mac,2.2_McChicken,1.5  
s=Big Burger,1.2_Bone Soup,2.5  
s=French Fries,1.2_Vegetable Seafood Soup,2.5  
s=Hot Chocolate,1.2_Pineapple Pie,2.5  
s=Karubi Beef,1.2_Low Fat Yogurt Blackberry,2.5  
s=Pork Chop with rice,9.5_Egg Soup,3.2  
s=Beef with rice,9.5_Soup,3.7_Pork pickled mustard green noodles,10  
s=Seafood noodles,9.5_Glutinous rice,0.9_Dumplings,5.5  
s=Spring rolls,1.5_Vegetable soup,0.8_Rice and vegetable roll,1  
s=Spicy hot noodles,5_Soup,3.7_Oily bean curd,2  
[SSO Service][Init Account] Before:4d2a46c7-71cb-4cf1-b5bb-b68406d9da6f  
[SSO Service][Init Account] After:4d2a46c7-71cb-4cf1-b5bb-b68406d9da6f  
[SSO Service][Init Account] Before:1d1a11c1-11cb-1cf1-b1bb-b11111d1da1f  
[SSO Service][Init Account] After:1d1a11c1-11cb-1cf1-b1bb-b11111d1da1f  
[Consign price service] [Init data operation]  
[Consign Price Service][Create New Price Config]  
[Route Service] Create And Modify Start:shanghai End:taiyuan  
[Route Service] Create And Modify Start:nanjing End:beijing  
[Route Service] Create And Modify Start:taiyuan End:shanghai  
[Route Service] Create And Modify Start:shanghai End:taiyuan
```

# 로그, 메트릭, 데이터 기반의 시스템 이상징후 탐지



# 분산시스템에서의 이상징후 탐지



- Agent: 로그 및 메트릭 정보의 수집 및 전송, 로그 출력 양 조절, (긴급 패치 적용)
- Worker: 이벤트 생성을 위한 스트림 데이터 처리
- Event Analyzer: 이상징후 분석

# 로그 삽입을 통한 시스템 상태 정보 수집

## • 로그 삽입

- 소스코드 수정 없이 로그 생성 코드 추가
  - ✓ Java: Bytecode Instrumentation (BCI)
  - ✓ Python: Meta-Object protocol + Dynamic Typing
- 함수 시작/종료 지점, 매개변수, 인수, 반환값, 실행시간, print 문 등

## • 피드백 기반의 로그의 양 조절

- 함수의 중요 수준, 시스템의 자원 사용량, 이상징후의 심각도에 따라 출력되는 로그의 양 조절
- 시스템의 이상 정도를 점수로 계산
  - ✓ 이벤트 밀도: 일정 시간 동안 시스템의 상태 변화량

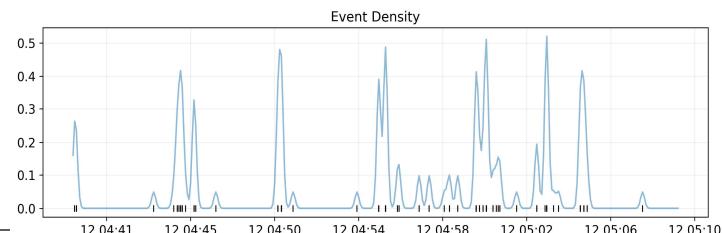


Fig. Processed Event stream using Kernel Density Estimation

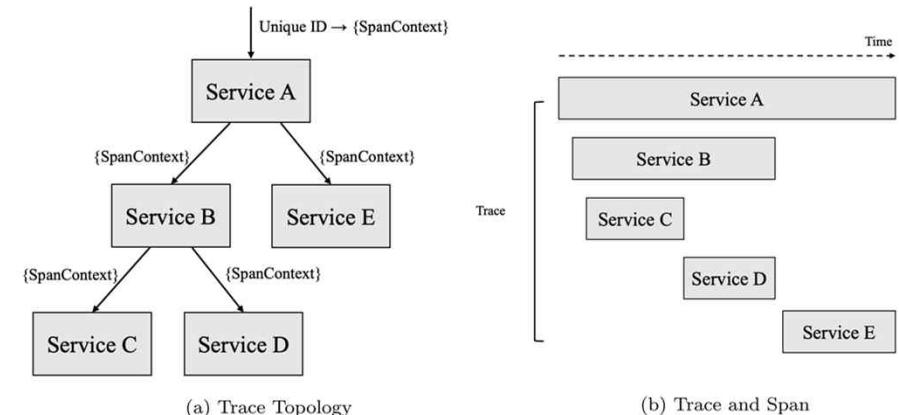


Fig. 2 Traces and Spans in Distributed Tracing

```
{'context': {'span_id': -8820367226001093835,
             'thread_id': '53',
             'trace_id': 2766762858404346241},
  'elapsed_time': datetime.timedelta(microseconds=139),
  'end_time': datetime.datetime(2022, 12, 13, 6, 9, 46, 494053, tzinfo=tzutc()),
  'events': [{"attributes": ['@parameter0: io.netty.channel.Channel [id: '
                           '0xb62100a7, L:/192.168.64.2:28000 - '
                           'R:/155.230.118.234:44812'],
              'name': 'Method Start',
              'timestamp': datetime.datetime(2022, 12, 13, 6, 9, 46, 493914,
                                             tzinfo=tzutc())},
             {"attributes": ['java.lang.Integer -1239351129'],
              'name': 'Method end',
              'timestamp': datetime.datetime(2022, 12, 13, 6, 9, 46, 494053,
                                             tzinfo=tzutc())}],
             'name': '<com.finedigital.bean.SensorEventHandler: java.lang.Integer '
                     '>getChannelId(io.netty.channel.Channel)> ',
             'parent_id': 3969049755867096819,
             'start_time': datetime.datetime(2022, 12, 13, 6, 9, 46, 493914, tzinfo=tzutc())}}
```

# 지진 경보 시스템에서의 오류 로그 수집

- 시스템에 존재하는 오류에 대한 로그 수집
- Fault Injection 통한 로그 수집
- <http://155.230.118.234:18000/>에서 확인 가능

## CrowdQuake, EQMS 버그 리포트 목록

CrowdQuake, EQMS에서 발생 가능한 버그, 문제점에 대해 설명.

순번	예상 원인	예상 원인 위치	예상 원인 유형	영향을 받는 서비스
1	EQMS 패킷에서 센서 데이터 길이가 0인 경우	EQMS	Code	CrowdQuake Gateway
2	EQMS 자원 부족으로 인해 센서 데이터가 CrowdQuake로 포워딩 되지 못하고 버퍼에 쌓이는 문제	EQMS	HW, Configuration	CrowdQuake Gateway
3	Gateway에서 패킷 메시지 타입 체크 버그	CrowdQuake Gateway	Code	
4	Batch-ML에서 버퍼 길이보다 더 큰 메시지가 들어오는 경우	CrowdQuake Batch-ML	Code, Configuration	CrowdQuake Batch-ML
5	Batch-ML에서 거의 가득 찬 버퍼에 남은 공간보다 큰 메시지가 들어오는 경우	CrowdQuake Batch-ML	Code, Configuration	CrowdQuake Batch-ML
6	Batch-ML Kmeans에서 Floating-point 에러	CrowdQuake Batch-ML		CrowdQuake Batch-ML
7	Kafka 디스크 공간 부족과, 연관된 서비스의 버퍼 문제	Kafka	HW, Configuration	All CrowdQuake Components
8	메모리 누수 문제			

# 지진 경보 시스템에서의 오류 로그 수집

- 센서 데이터 길이가 0으로 된 패킷

- 패킷 헤더와 데이터에 대한 검증이 없으므로 CrowdQuake의 Gateway로 전송된 패킷으로 인하여 Gateway에 장애가 발생할 수 있으며, 장애가 발생된 Gateway는 재시작되고 다른 Gateway로 재전송 시도.

문제 발생 예상 위치

- eqms-data-collector/src/main/java/com/finedigital/bean/DataToKnuSender.java

```
1 public void putData(String mid, byte[] bytes) {
2     if (senderRunning && mid.length() >= 10) {
3         ByteBuffer buffer = ByteBuffer.allocate(bytes.length + 8);
4         buffer.put((byte) 0xF1);
5         buffer.putInt(NettyServer.intToByte(Integer.parseUnsignedInt(mid)));
6         buffer.put(NettyServer.shortToByte((short) bytes.length)); // 예상
7         buffer.put(bytes); // 예상 위치
8         buffer.put((byte) 0xFD);
9         try {
10             bqSendDataes.out(buffer);
11             if (bqSer • crowdquake-gateway/crowdquake/core/eqms.py
```

```
1     try:
2         key = self._current_packet_usim
3
4         # From here, we can fail-over when packet cannot parsed,
5         # by discard current packet.
6         packet = packet[:-1]
7         if packet[0] != self.CONTENT_START_BYTE or packet[-1] != self.CONTENT
8             self._failed += 1
9             self._ctx.log_failure(err="EQMS_MSG_PARSE_FAILURE", reason="Message
10            continue
11
```

```
18_gateway-0_1"} 2022-12-27 08:40:23.444 | ERROR    | crowdquake.gateway:r0_
18_gateway-0_1"} protocol: <crowdquake.core.eqms.EQMSProtocol object at 0x7f50
18_gateway-0_1"} transport: <TCPTransport closed=False reading=False 0x55b208-
18_gateway-0_1"} Traceback (most recent call last):
18_gateway-0_1"}     18_gateway-0_1"} File "app.py", line 57, in <module>
18_gateway-0_1"} gateway.run_forever()
18_gateway-0_1"} |           | <function EQMSGateway.run_forever at 0x7f5656780af8>
18_gateway-0_1"} |           | <crowdquake.gateway.EQMSGateway object at 0x7f5656bd37c0>
18_gateway-0_1"} |           |
18_gateway-0_1"}     18_gateway-0_1"} File "/app/crowdquake/gateway.py", line 212, in run_forever
18_gateway-0_1"} self._loop.run_forever()
18_gateway-0_1"} |           | <method 'run_forever' of 'uvloop.loop.Loop' objec
18_gateway-0_1"} |           | <uvloop.Loop running=True closed=False debug=False>
18_gateway-0_1"} |           | <crowdquake.gateway.EQMSGateway object at 0x7f5656bd37c0>
18_gateway-0_1"} |           |
18_gateway-0_1"}     18_gateway-0_1"} > File "uvloop/handles/stream.pyx", line 829, in uvloop.loop
18_gateway-0_1"} run_in_context1(
18_gateway-0_1"}     18_gateway-0_1"} File "uvloop/loop.pyx", line 107, in uvloop.loop.run_in_context
18_gateway-0_1"} return context.run(method, arg)
18_gateway-0_1"}     18_gateway-0_1"} File "/app/crowdquake/core/eqms.py", line 191, in data_receiv
18_gateway-0_1"} if packet[0] != self.CONTENT_START_BYTE or packet[-1] != self.CONTENT
18_gateway-0_1"} |           | <memory at 0x7f5678febf40>
18_gateway-0_1"} |           | <crowdquake.core.eqms.EQMSProtocol object at 0x7f565678febf40>
18_gateway-0_1"} |           | <memory at 0x7f5678febf40>
18_gateway-0_1"} |           |
18_gateway-0_1"}     18_gateway-0_1"} IndexError: index out of bounds on dimension 1
18_gateway-0_1"} 2022-12-27 08:40:23.450 | WARNING   | crowdquake.core.eqms:co
18_gateway-0_1"} 2022-12-27 08:40:23.450 | INFO      | crowdquake.core.eqms:co
```

# 지진 경보 시스템에서의 오류 로그와 메트릭 수집

- 자원 부족으로 인한 데이터 처리 및 전송 지연

- 센서 데이터는 버퍼에 임시 저장된 이후 CrowdQuake로 전송되지만 자원 부족으로 인하여 데이터 처리 및 전송 지연이 발생할 경우 데이터 유실이 발생함

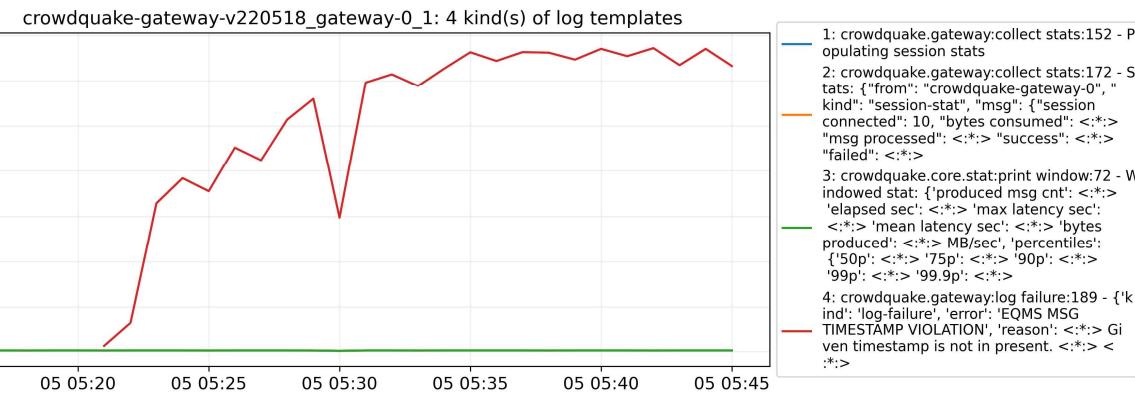
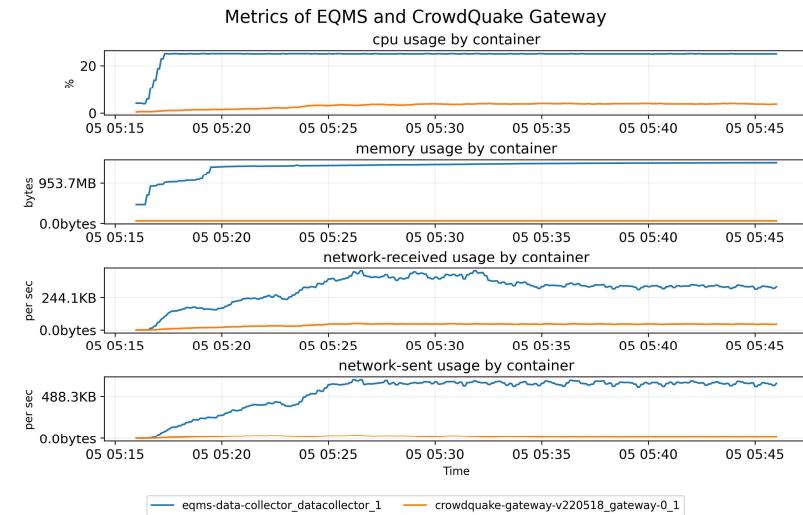
- eqms-data-collector/src/main/java/com/finedigital/bean/DataToKnuSender.java

- BlockingQueue를 사용. 여러 스레드가 동시에 접근하려 하면 문제 발생 가능.

```
1 // ...
2 import java.util.concurrent.BlockingQueue;
3 import java.util.concurrent.LinkedBlockingQueue;
4 // ...
5 @Component
6 public class DataToKnuSender {
7     // ...
8     BlockingQueue<ByteBuffer> bqSendDatas = new LinkedBlockingQueue<>();
9     // ...
```

```
31         long serverIdx = System.currentTimeMillis();
32         ByteBuffer buffer = bqSendDatas.take();
33         outStrm.write(buffer.array());
34         outStrm.flush();
35         LogKnuSent(serverPort + serverIdx, buffer.array())
```

```
▼ 2022-12-30 18:55:39 2022-12-30 09:55:39.070 | WARNING | crowdquake.gateway:log_failure:189 - {'kind': 'log-failure', 'error': 'EQMS_MSG_TIMESTAMP_VIOLATION', 'reason': '01232973321: Given timestamp is not in present. 167239383900 0 1672394139070'}
```



## 향후 연구

- 로그, 메트릭, 데이터 기반의 시스템 이상징후 탐지 연구
  - 로그 출력양 조절을 통한 시스템 상태 정보 수집
  - 이상징후 원인 추적 연구
- 런타임 시스템 상태 복구를 통한 장애 대응
  - 핫패치의 런타임 적용을 위한 시스템 실행 상태 저장 및 복구 기술 개발

# 감사합니다

