연구소개 및 테스트베드 개발 상황 공유

경북대학교 권영우
(지능소프트웨어 시스템 연구실)

연구실 주요 연구 내용

- 재난디지털트윈 (시스템 개발)
  - 지진관측/경보시스템
  - 폭염디지털트윈
- Anomaly 탐지 (현상 관찰)
  - 재난 탐지
    - 지식 그래프 기반의 재난 상황 인식
    - 딥러닝 기반의 실시간 재난 탐지 (지진, 폭염)
  - 로그기반 시스템 이상 징후 탐지
  - 취약점/악성코드 탐지
- 분산시스템 (시스템 개선)
  - Serverless 컴퓨팅

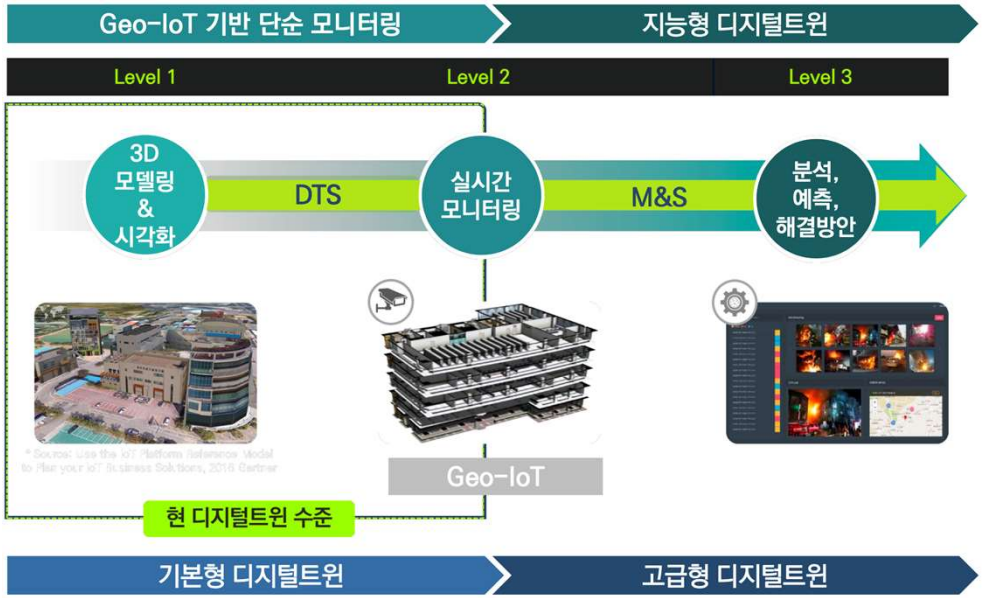
디지털 트윈을 통한 재난 관리

기존 재난관리 한계점

- 변경된 현실 세계를 반영하여 **실시간으로 후속 재난의 영향을 분석**할 수 있는 기술 부재
- **2개 이상의 동시 다발 재난**에 대한 대응 미흡

디지털 트윈 기술

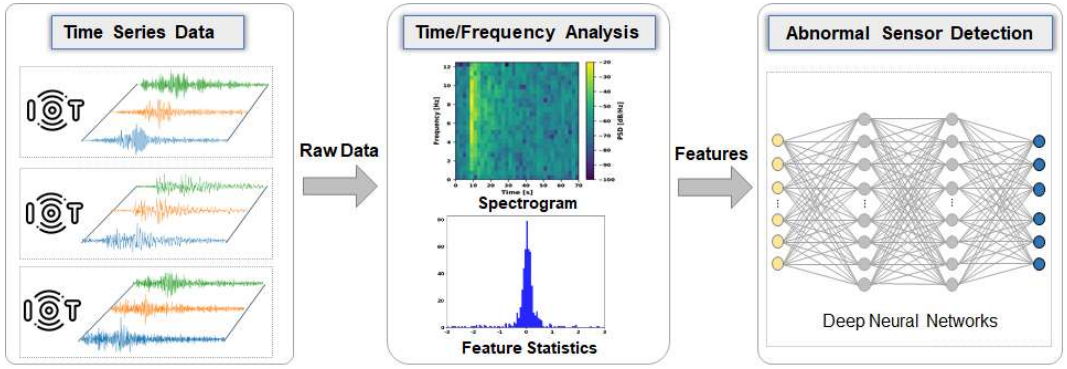
- 디지털 트윈은 현실세계에서 수집한 다양한 정보를 **가상세계에서 분석**하고 최적화 방안을 도출하여 이를 기반으로 **현실세계를 최적화**하는 지능화 융합 기술
- 현실세계와 동일한 가상세계에서의 **예측과 대응 중심의 재난 관리를 통한 도시의 재난복원력 회복**



재난 상황 인지를 위한 이상징후 감지 기술 개발

센서 기반 이상징후 감지 및 재난 탐지

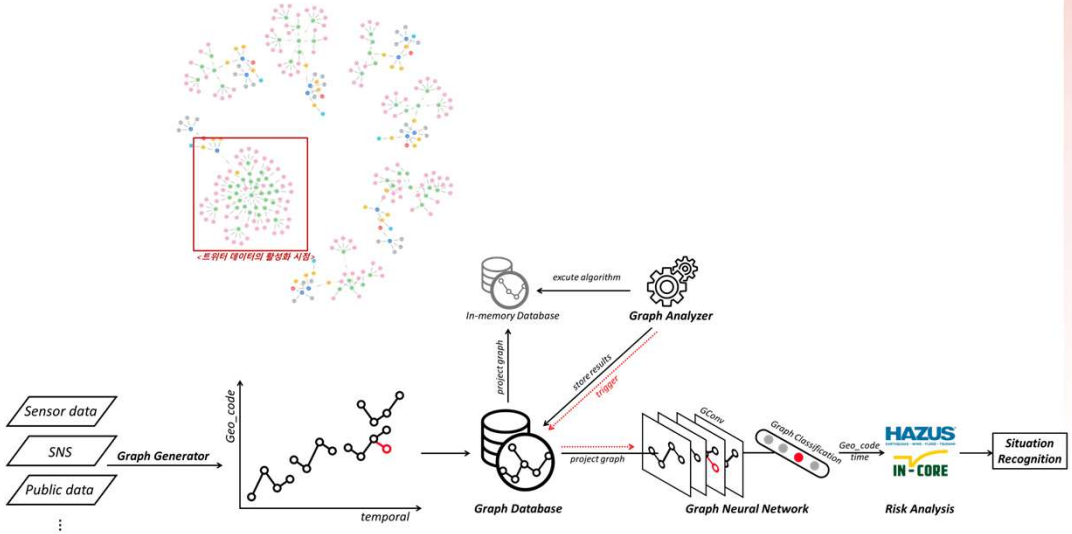
- **실시간 시계열 데이터 분석** 모듈 개발
- 시간/주파수 분석을 통한 정상 **시계열 데이터 패턴 분석** 및 이를 바탕으로 한 센서 이상징후 감지 기술 개발
- 근거리 센서들 간의 **시계열 데이터의 상관 분석**을 통한 지역적인 영향으로 인한 센서 이상징후 감지 기술 개발



〈이상징후 감지 절차〉

다양한 데이터를 활용한 인공지능 기반의 상황 인지

- IoT, SNS(트위터), 공공데이터, 뉴스 등 **다양한 데이터를 통합한 지식그래프 분석**
- **그래프 신경망**을 이용한 **유사 재난 검색 및 재난의 확산** 예측
- 지속적/장기적 리스크 분석 및 관리, 재난 대응을 위한 **의사결정**

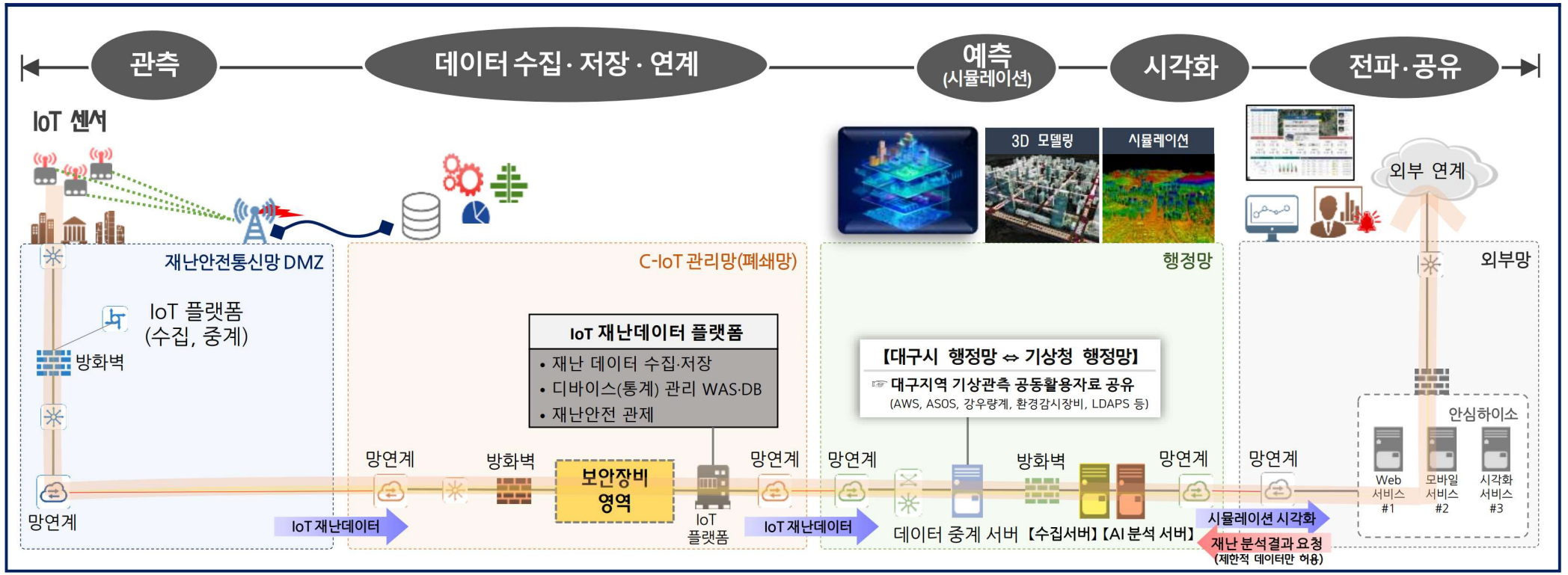


〈지식그래프 기반의 재난 탐지〉

재난 디지털트윈

대구시 재난(폭염) 디지털트윈 구축 (2023 ~ 2024)

재난디지털트윈 시스템 및 네트워크 구성 방안



지진경보시스템 테스트베드 개발 상황

KNU KYUNGPOOK NATIONAL UNIVERSITY

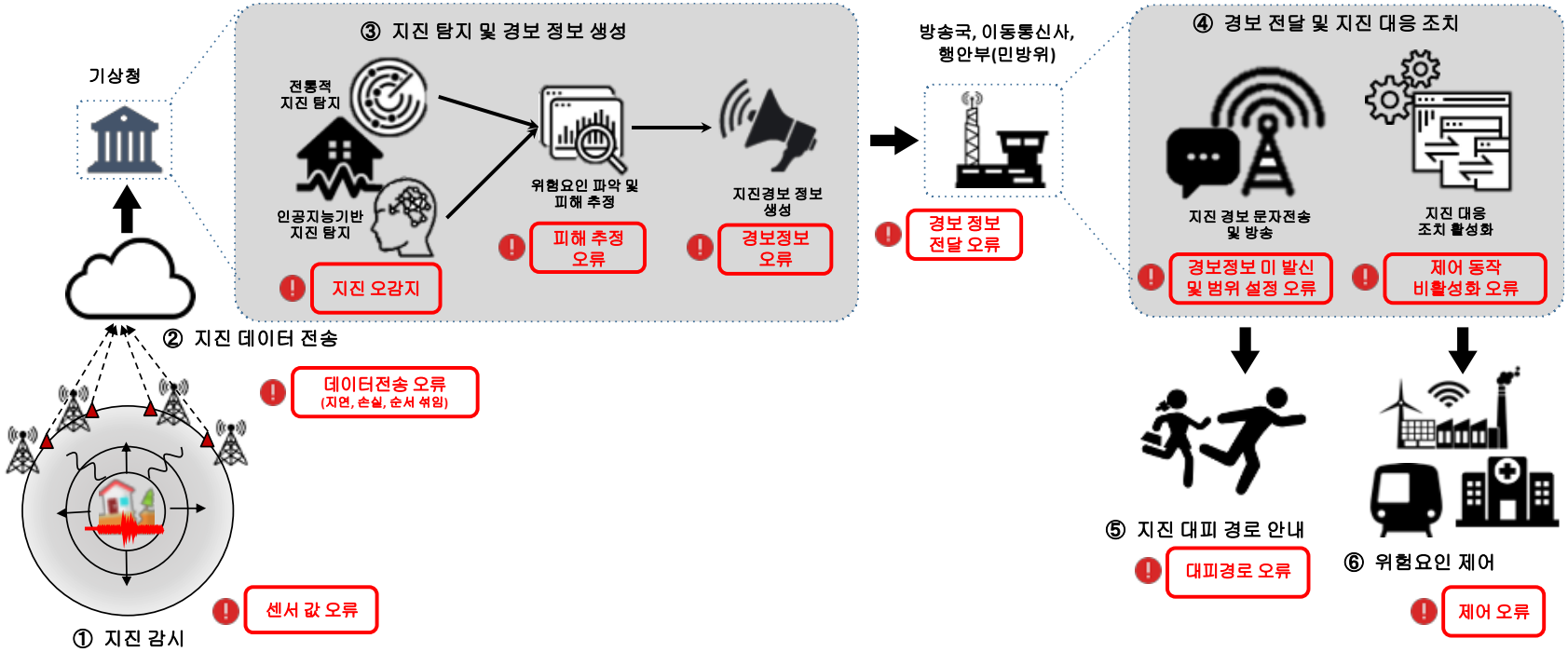
Intelligent Software Systems Lab
KYUNGPOOK NATIONAL UNIVERSITY

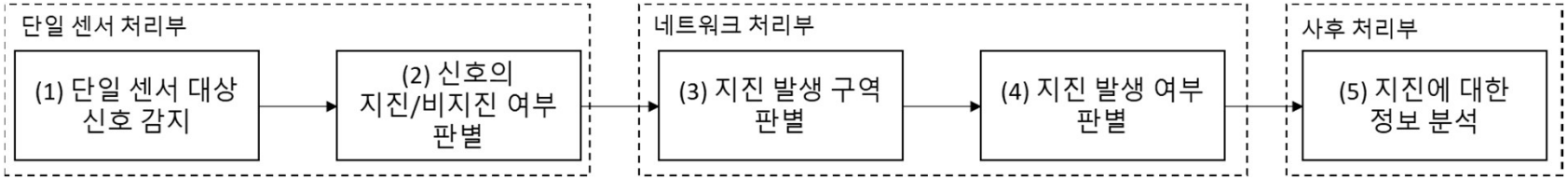
소프트웨어재난연구센터

KNU 경북대학교

CrowdQuake (a.k.a 테스트베드)

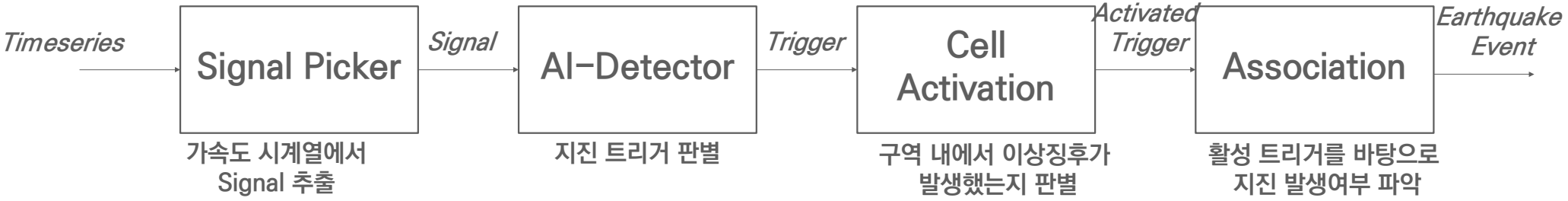
지진경보 시스템 테스트베드 개요





CrowdQuake (a.k.a 테스트베드)

실시간 지진 감지 과정



Signal Picker: 가속도 시계열에서 신호의 발생여부 확인 (현재 STA/LTA 사용중)

AI Detector: Signal Picker에서 감지한 신호를 딥러닝 모델을 사용하여 지진 가능성이 높은 트리거로 구분

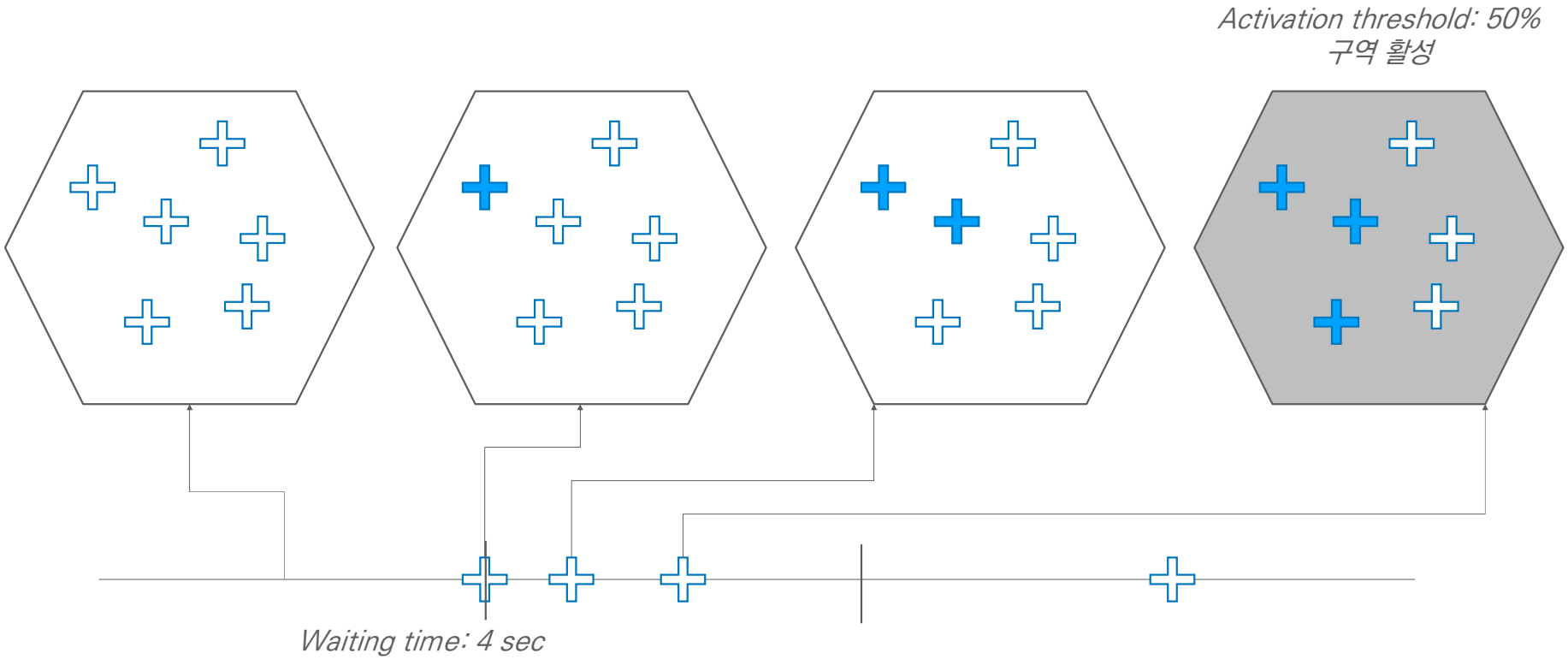
Cell Activation: 구역 내에서 이상징후가 발생했는지의 여부를 판별

- 구역 내에서 {activation_threshold}% 이상의 센서가 {waiting_time} 초 사이에 동시에 trigger 되었는지 여부

Association: 활성 트리거를 바탕으로 실제 지진의 발생여부 판별 및 발생한 지진에 대한 추적

CrowdQuake (a.k.a 테스트베드)

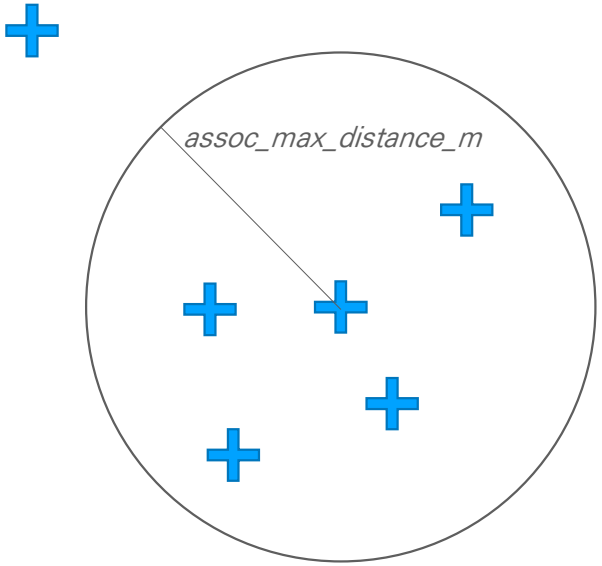
Cell Activation 과정



Waiting time: 첫 트리거 이후 후속 지진 탐지 센서를 기다리는 시간
Activation threshold: 해당 구역을 지진 발생 구역으로 판정을 하기 위한 센서의 비율 (예: 50%)

CrowdQuake (a.k.a 테스트베드)

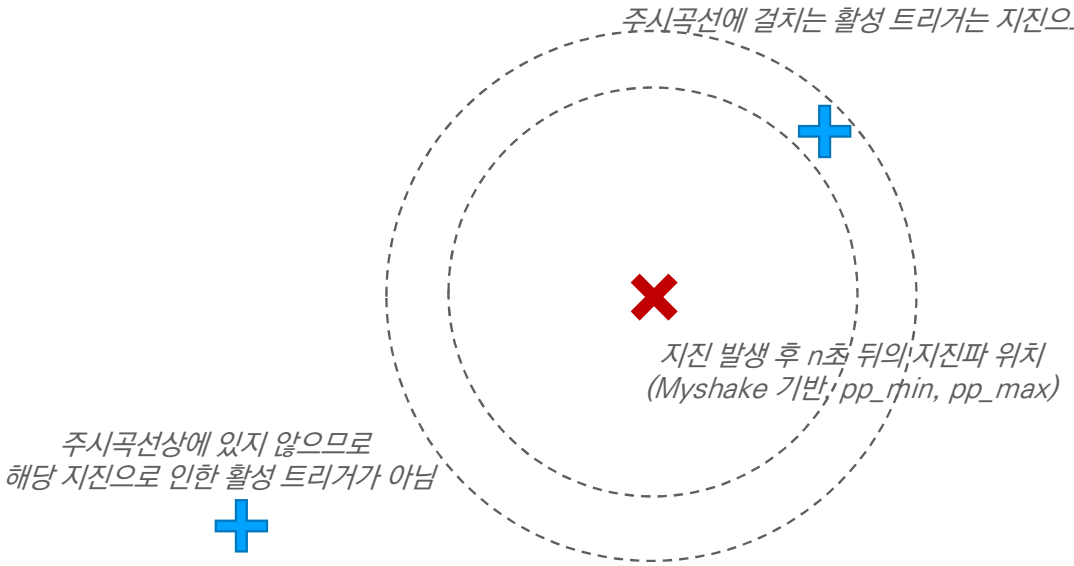
Association 과정



초기 동일 지진 이벤트 판별

특정 활성 트리거 기준
{assoc_max_distance_m}미터 내에
{assoc_max_time_msec} 밀리초 사이에
{assoc_ration}% 이상의 활성
트리거가 존재 시 동일 지진이벤트로 판별

* 지진 이벤트로 판별 후 경보 발령 가능



후기 지진 감지/지진 추적 단계

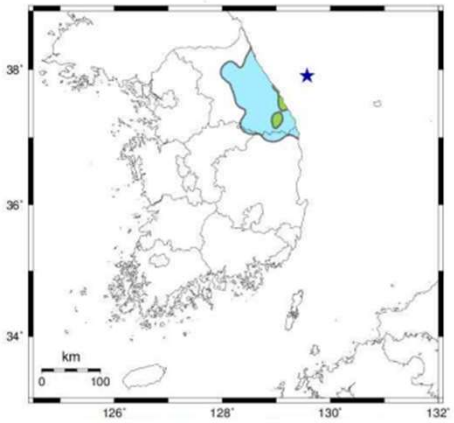
초기 지진 감지에서 감지된 각 지진마다
새로 발생한 활성 트리거가 해당 지진으로 인한
활성 트리거인지를 판별
각 지진은 {assoc_earthquake_expire_sec}초 동안
시스템에서 유지

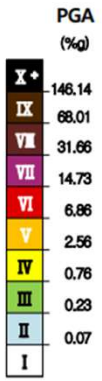
CrowdQuake (a.k.a 테스트베드)

사례: 2023년 5월 15일 동해 지진 M4.5

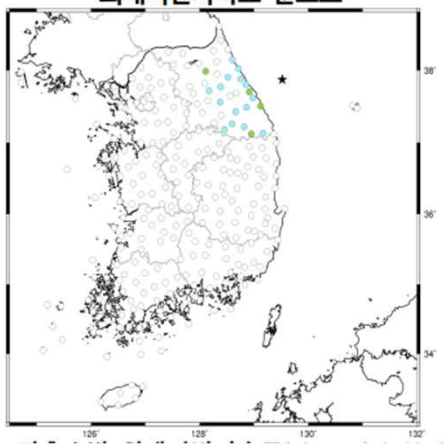
• 발생시각		2023년 5월 15일 06시 27분 37초			
• 위치(불확도)		강원 동해시 북동쪽 52km 해역 위도: 37.874° N, 경도: 129.522° E (±4.29 km)			
• 규모(불확도)		4.5 M_L (± 0.3)	깊이	31 km	
• 진도	최대계기진도	Ⅲ(강원, 경북), Ⅱ(충북)			
	최대지반가속도	관측소	동해(TOHA)	PGA(%g)	0.395

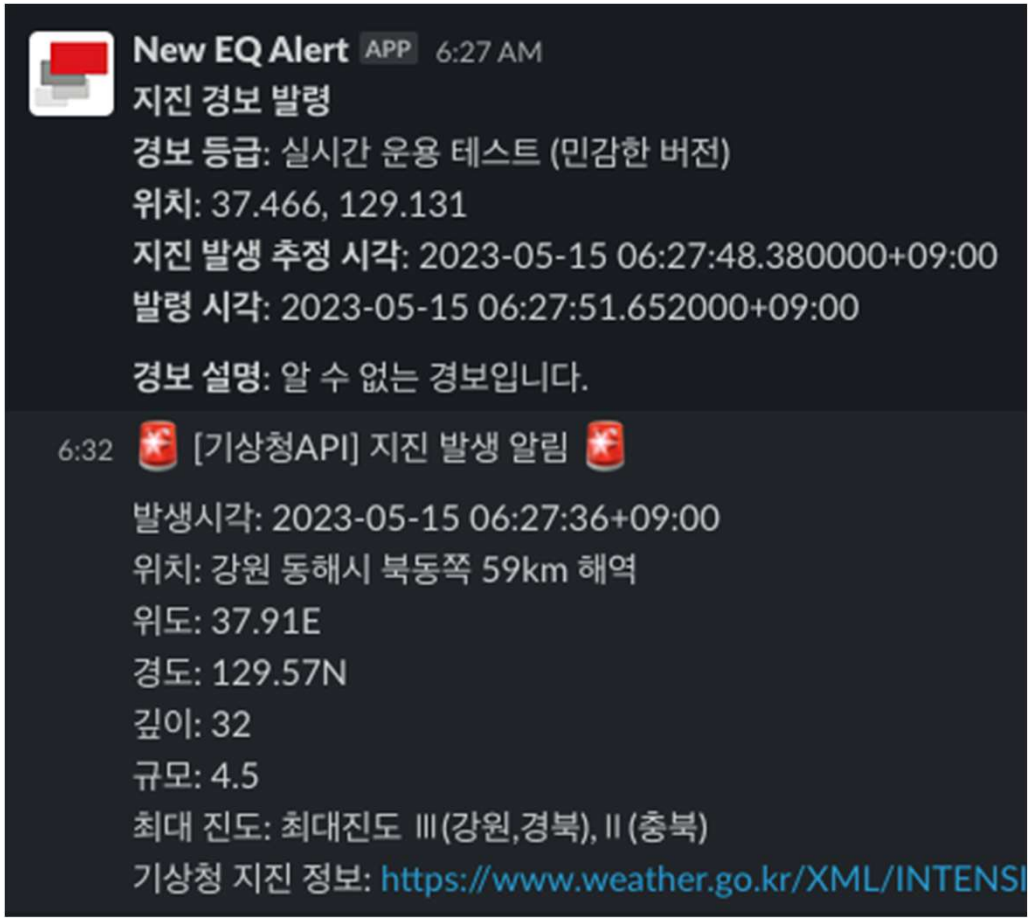
진도분포도





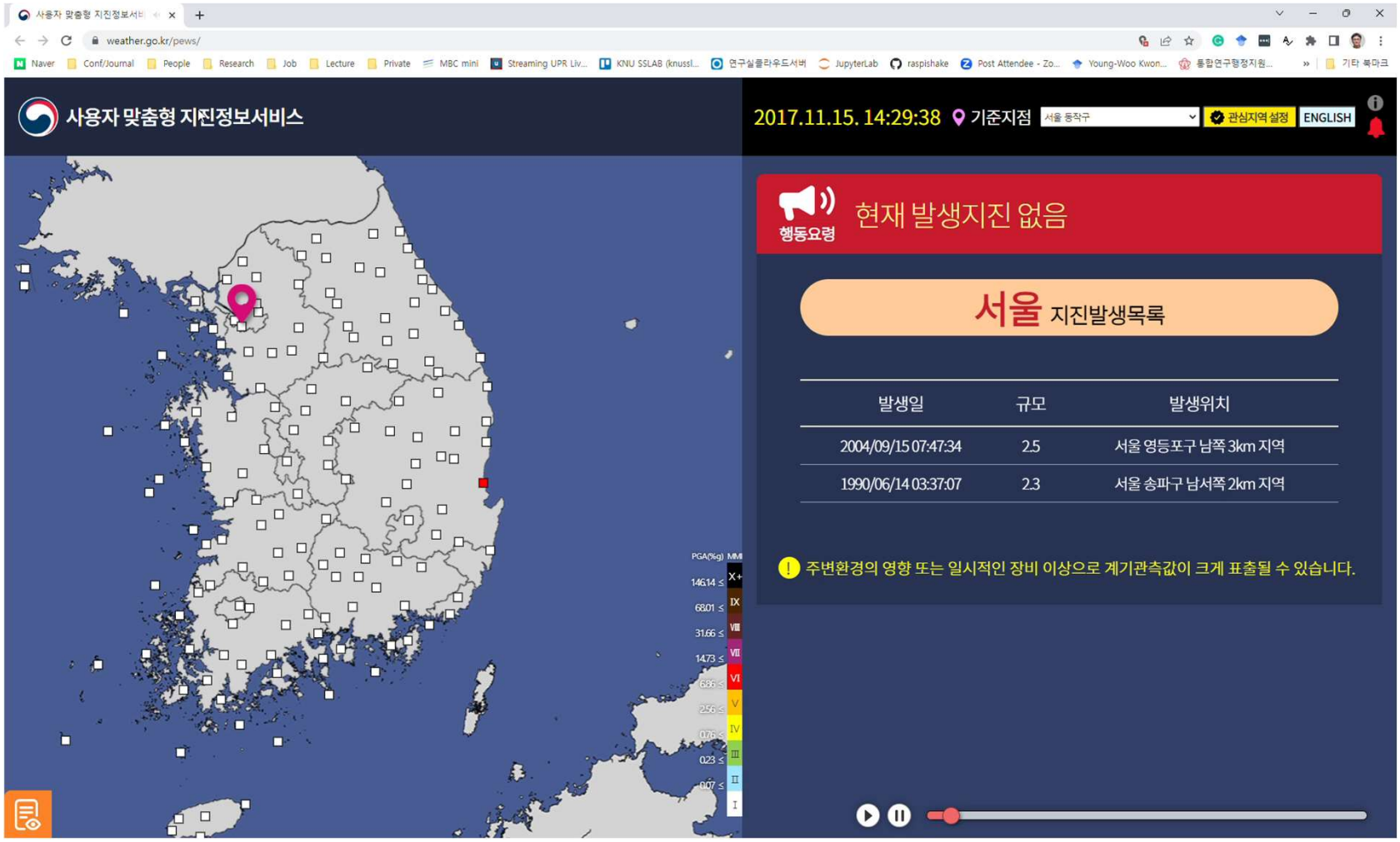
최대지반가속도 분포도





CrowdQuake (a.k.a 테스트베드)

경보 기능 개발



실시간 시스템 이상징후 탐지
(로그 기반 이상징후 탐지)

KNU KYUNGPOOK NATIONAL UNIVERSITY

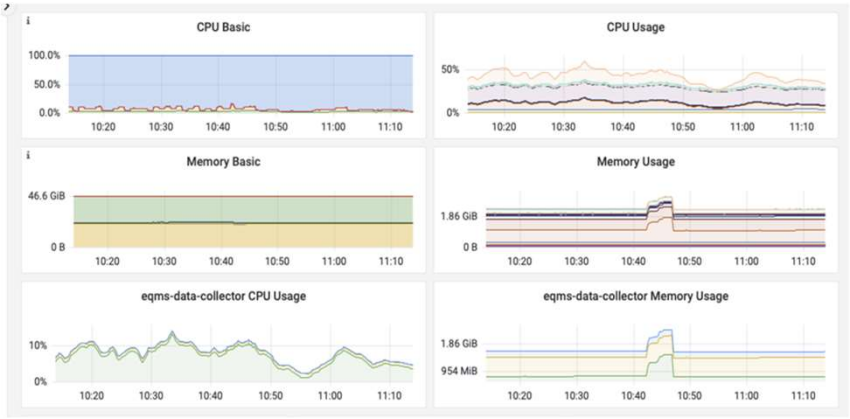
Intelligent Software Systems Lab
KYUNGPOOK NATIONAL UNIVERSITY

소프트웨어재난연구센터

KNU 경북대학교

실시간 시스템 이상징후 탐지

- 시스템 이상징후
  - 비정상 CPU, 메모리, 네트워크 사용
  - 비정상 동작, 예외 발생, 시스템 크래쉬 등
- 로그 예




```
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 714        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 / a1c602042212050143110000 / bytes: 71
datacollector_1    | Message forwarded of size 713        4
datacollector_1    | Message forwarded of size 713
datacollector_1    | Message forwarded of size 713        ------------------------------------------------------------
datacollector_1    | Message forwarded of size 713        01227501104,1670202374000,1670169975125,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202376000,1670169977084,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202379000,1670169980119,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202380000,1670169981100,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202384000,1670169985101,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202385000,1670169986104,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202387000,1670169988114,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202389000,1670169990086,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202394000,1670169995093,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202397000,1670169998131,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202411000,1670170012097,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202414000,1670170015126,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202421000,1670170022097,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202424000,1670170025126,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202425000,1670170026095,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202447000,1670170048120,9
datacollector_1    | Message forwarded of size 714        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
```

실시간 시스템 이상징후 탐지

- 지난 연구
  - 지진경보시스템의 오류 찾기 (홍신 교수님)
  - 코드의 오류가 시스템에 미치는 영향이 있는가?
    - 영향을 미친다면 언제 증상이 나오는가?
    - 시스템에서의 증상(이상징후)이란 무엇이고, 이를 탐지할 수 있는 방법은 무엇인가?
      - 이상징후: 비정상 동작, 크래쉬, 기능 요구사항 미충족 등
      - 탐지 방법: 메트릭, 로그 등
    - 코드의 오류가 로그에 나타나는가?
      - 코드에서 오류가 있더라도 로그에는 나타나지 않는 경우가 많음 (앞서 찾은 오류가 로그에 나타나지 않음!)
      - 로그에는 오류에 대한 정보가 담겨있지 않는 경우가 많음 (개발자는 오류 상황을 모름 -> 필요한 정보가 로그에 없음)
    - 그렇다면, 로그를 추가해보자!

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

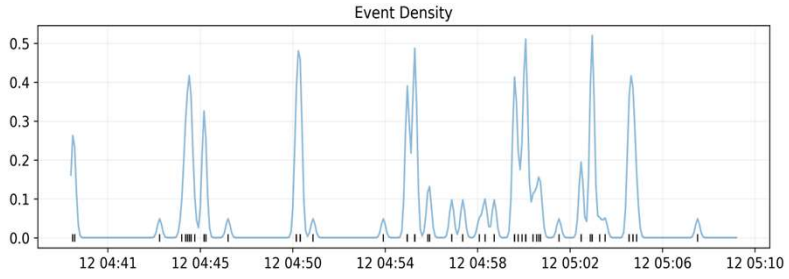


Fig. Processed Event stream using Kernel Density Estimation

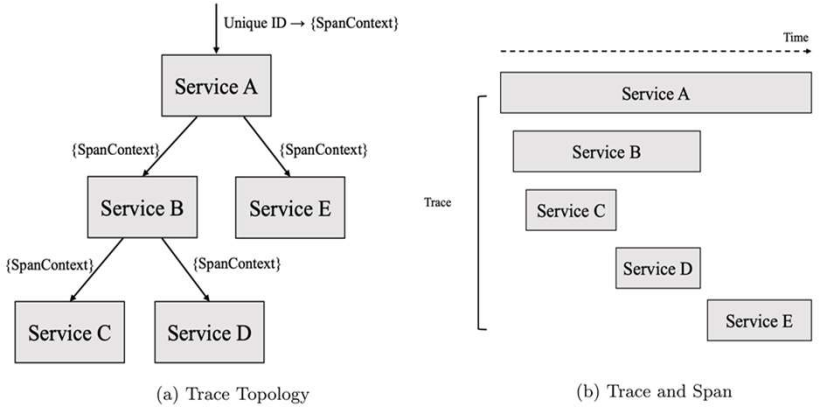


Fig. 2 Traces and Spans in Distributed Tracing

```
{'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())}
```

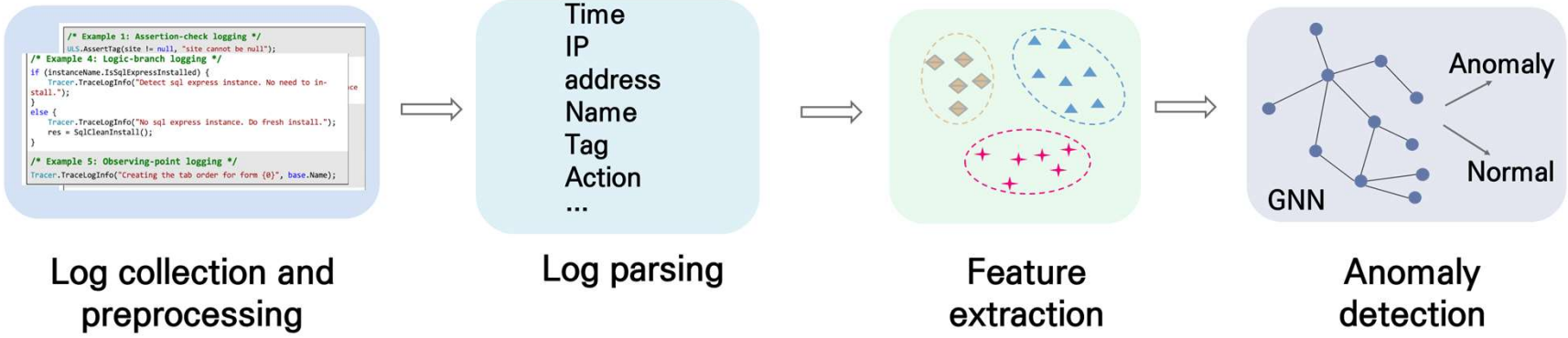
로그 기반의 Anomaly 탐지

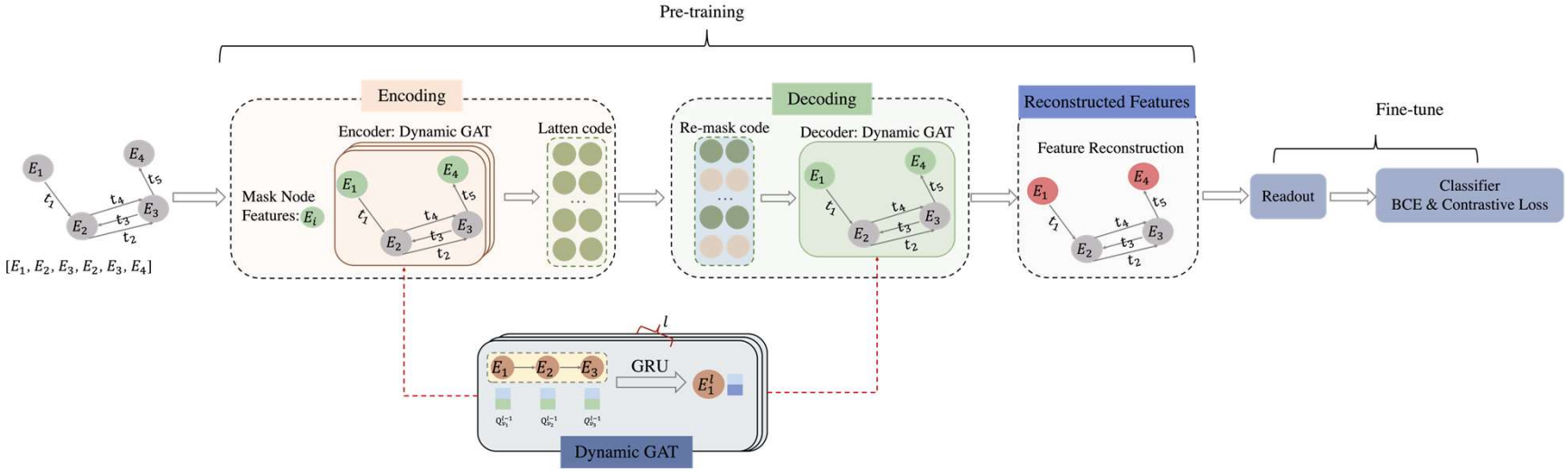
- Anomaly의 종류
  - Execution path anomaly
  - Parameter value anomaly
  - Performance anomaly
- 로그 데이터의 그래프화

Figure 2: The example of raw log parsing and graph construction.

로그 Anomaly 탐지

- 그래프 기반의 Anomaly 탐지 개요





그래프 기반의 로그 Anomaly 탐지

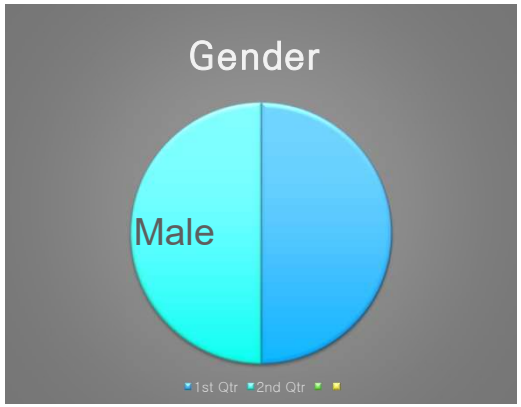
- **실험 결과**
  - **데이터셋**

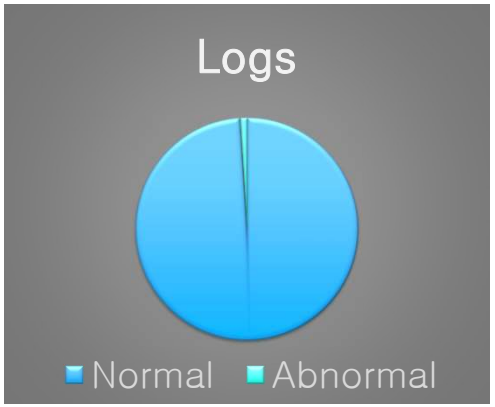
Dataset	#log Events	#Duration	#of Logs	#of Anomalies	#Anomaly Rate
HDFS	47	38.7 hours	1175629	16838	1.43%
BGL	377	7 months	4747963	348460	7.34%
Thunderbird	1758	244 days	2000000	6590	0.33%

Model	Datasets								
	HDFS			BGL			Thunderbird		
	Precision	Recall	F1_score	Precision	Recall	F1_score	Precision	Recall	F1_score
PCA	0.8583	0.6547	0.7766	0.2647	0.6337	0.3439	0.5337	0.4167	0.4973
IM	0.8185	0.7561	0.7923	0.2876	0.4734	0.4219	0.3367	0.4132	0.5431
DeepLog	0.9124	0.7841	0.8625	0.7452	0.9119	0.7612	0.4554	0.3786	0.6215
LogRobust	0.9335	0.9578	0.9623	0.9115	**0.9217**	0.8815	0.8118	0.8726	0.7967
DeepTraLog	0.8577	0.9126	0.8365	0.7742	0.8337	0.6953	0.6367	0.7331	0.6126
Pre-LogMGAE	**0.9798**	**0.9917**	**0.9815**	**0.9543**	0.8736	**0.9048**	**0.8367**	**0.9935**	**0.9042**

로그 기반 이상징후 탐지 연구의 한계 (혹은 Challenges)

- 로그 Anomaly 탐지 != 시스템 이상징후 탐지
  - (즉, 로그의 이상 패턴과 시스템의 이상 징후와는 다름)
- 데이터 불균형





- Unseen (처음 보는, 학습이 되지 않은) 데이터

감사합니다