


1단계(3차년도) 기술문서

(과제명) 대규모 분산 에너지 저장장치 인프라의 안전한 자율운영
및 성능 평가를 위한 지능형 SW 프레임워크 개발

(과제번호) 2021-0-00077

- 결과물명 : ESS 테스트베드 및 안전SW 프레임워크 간 상호
연동 및 시나리오 기술서
- 작성일자 : 2023년 11월 20일

과학기술정보통신부 SW컴퓨팅산업원천기술개발사업
“1단계(3차년도) 기술문서” 로 제출합니다.

수행기관	성명/직위	확인
한국전자기술연구원	최효섭/책임연구원	

정보통신기획평가원장 귀하



사 용 권 한

본 문서에 대한 서명은 한국전자기술연구원 내부에서 본 문서에 대하여
수행 및 유지관리의 책임이 있음을 인정하는 것임.

본 문서는 작성, 검토, 승인하여 승인된 원본을 보관한다.

작성자 : 김창우, 박진원, 윤태일

일자 : 2023. 11. 20

검토자 : 김창우

일자 : 2023. 11. 21

승인자 : 최효섭

일자 : 2023. 11. 22

제 · 개정 이력

버전	변경일자	제·개정 내용	작성자
1.0	2023-11-20	최초 등록	김창우, 박진원, 윤태일

목 차

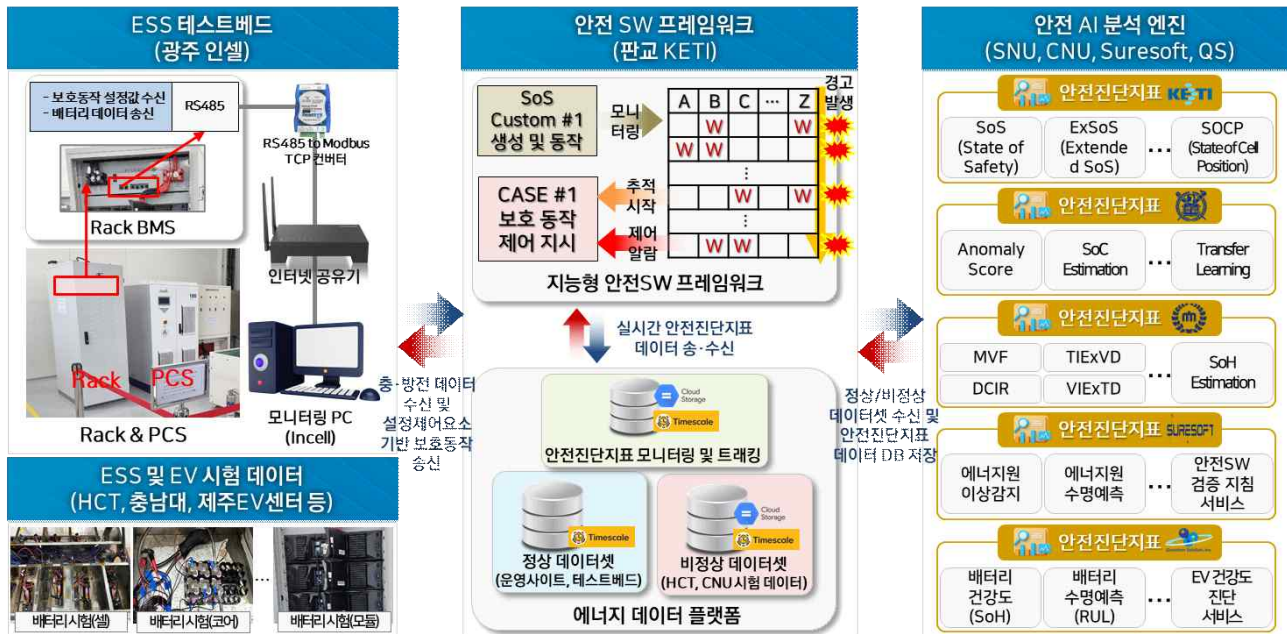
1. 개요	1
2. 맞춤형 SoS 서비스를 위한 데이터베이스 설계 및 개발	4
3. 맞춤형 SoS(State of Safety) 계산 처리 SW	5
4. 웹페이지 기반 맞춤형 SoS(State of Safety) 연동 가시화	8

1. 개요

□ 목적

- 본 기술문서는 ESS 운영사이트 및 테스트베드와 안전AI분석엔진과 연동된 안전SW 프레임워크 간 상호 기술 연동을 위한 기술 시나리오 문서이다. 각 기관에서 연구 개발한 안전AI분석엔진 기반 분석 결과를 통합한 SoS(State of Safety)로 수치화함으로써 ESS 운영 사이트 및 테스트베드 등 배터리 사용하는 인프라의 실시간 상태를 통합 수치화하는 것을 목적으로 한다.

□ 기술 개요



< 충·방전 배터리 보호동작 및 예지보전을 위한 컨소시엄 기술간 연동 구조도 >

- ESS 운영사이트 및 테스트베드와 안전AI분석엔진과 연동된 안전SW 프레임워크 간 상호 기술 연동을 위해 각 기관에서는 다음과 같이 역할을 나눠 기술하였다.

- ESS 테스트베드 :

- 1) “RS485 to Modbus TCP 컨버터”를 이용한 단일 포트 송·수신 통신 구조 BMS 개발 처리
 - * Modbus TCP - Read Input Register/Write Single Register를 활용하여 Testbed 데이터 송·수신 처리
 - * IP, Port, Device num, Protocol map
- 2) 인터넷 접속을 통한 BMS 네트워킹(Port-forwarding) 처리

- 안전AI분석엔진 :

1) 사전정보 :

- * 상한안전도(ξ) : 0.8 기본값 설정 (1일수록 안정된 값),
- * warning 초기값($h(x_{\xi})$) : 분석/예측값 기반 정수/실수값 (ex) 0.7, 4.05, 47 등),
- * 최대안전값($h(x_{100})$) : 정상범위 내에서 최댓값 (ex) 1, 4.00, 40 등)
- * #N회 이상 warning&critical 트리거 반복시 원인요소 및 제어값(V/I/T, ON/OFF) 처리 요청

2) 입력: KETI DB 접속 ⇒ ESS 운영사이트, ESS 테스트베드의 수집 데이터(DB)를 기반으로 실시간/주기적 데이터 로드

3) 출력: Timestamp, 분석/예측 결과값(정수/실수), warning 변경 요청값 (변경 요청 시, 분석/예측값(정수/실수)) ⇒ KETI DB 저장

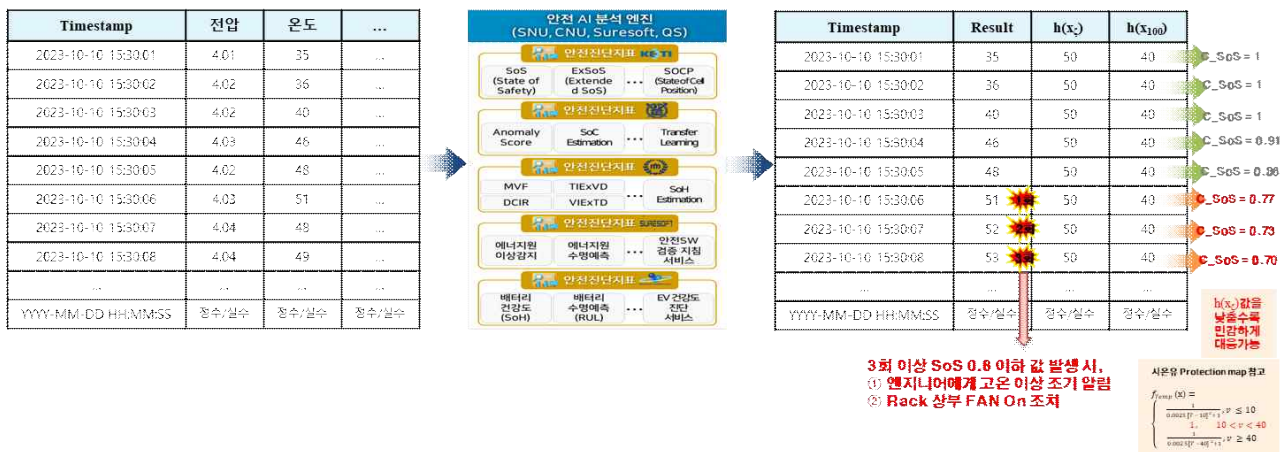
- 안전 SW 프레임워크 :

1) 데이터 수집 · 저장 :

- * Testbed BMS 데이터셋 및 이상 데이터셋의 DB 실시간/주기적 저장

2) 사용자 인터페이스 :

- * 안전AI분석엔진 요소 기반 Customized SoS 생성
- * Customized SoS 기반 안전AI분석엔진 요소 모니터링 처리
- * Customized SoS 기반 #N회 트리거 발생 시 warning&critical 알람 처리 및
- * 안전AI분석엔진 기반 원인요소 제어 데이터를 BMS에 수신 규격에 맞춰 송신



< 안전AI분석엔진 중 단일 엔진(배터리 고온 사전 조치 시나리오) 예시 >

- 공개SW 서비스 :

1) ESS 운영사이트에서 수집되는 데이터와 참여기관의 안전AI분석엔진 결과를 사용하여 배터리 사용하는 인프라의 실시간 상태를 통합 수치화

- * 전압, 전류, 온도와 같은 주요 데이터와 건강성 지표(Health Indicator), 이상 탐지, 에너지원 이상 감지, SoC, SoH 등 포함

- 2) 웹 사이트를 통해 사용자가 원하는 주요 데이터 분석 기법을 선택하여 맞춤형으로 SoS(State of Safety)를 제공하는 시스템으로 각 요소에 따른 모니터링 뿐 아니라 시스템 전체를 모니터링할 수 있는 기술을 제공

KETI SOS 서비스

분석컨테이너 선택

☐ 중남대건전성지표1
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ 중남대건전성지표2
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ 쿼텀솔루션지표1
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ 쿼텀솔루션지표2
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ DeeAnt
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ USAD
 0 1
 WARN: 0.40, SAFETY: 0.80

☒ under_voltage
 0 1
 WARN: 0.40, SAFETY: 0.80

☒ over_voltage
 0 1
 WARN: 0.40, SAFETY: 0.80

☒ over_temp
 0 1
 WARN: 0.40, SAFETY: 0.80

☒ under_temp
 0 1
 WARN: 0.40, SAFETY: 0.80

☐ anomaly_score
 0 1
 WARN: 0.40, SAFETY: 0.80

선택된 분석컨테이너

TAG	Value	Action	MAX_count	SAFETY_VALUE	WARN_THRESHOLD
over_temp	32.9	null	2	40	50
under_temp	29.2	null	2	10	0
under_voltage	4.038	null	2	3.3	3.15
over_voltage	4.057	null	2	4	4.086

Check

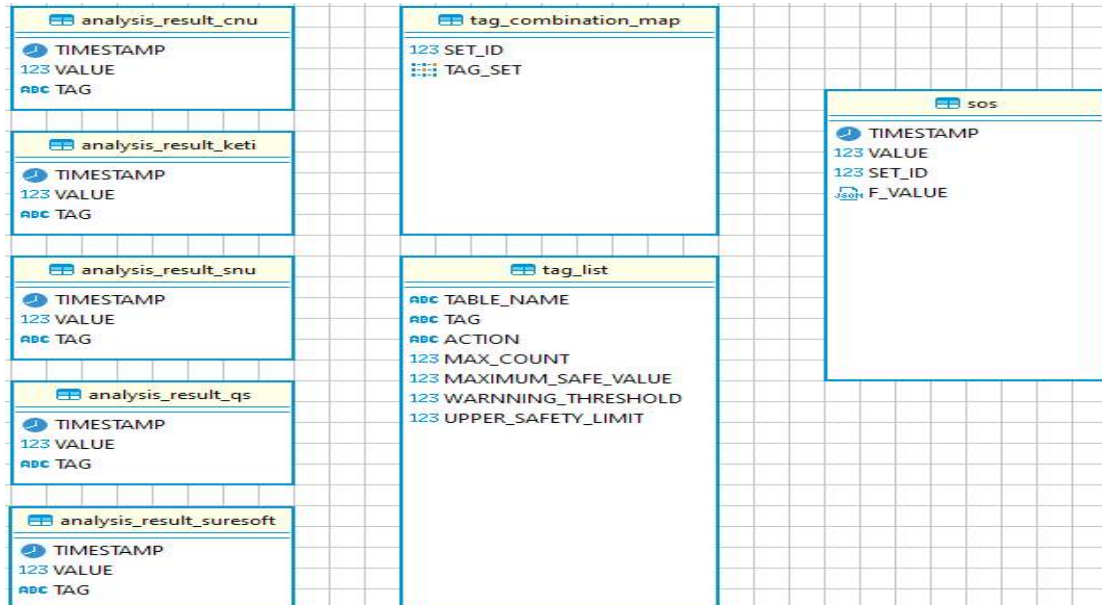
Create



< 맞춤형 SoS(State of Safety) 가시화 페이지 >

2. 맞춤형 SoS 서비스를 위한 데이터베이스 설계 및 개발

□ 맞춤형 SoS(State of Safety) 서비스를 위한 Table 설계



< 안전AI분석 SW 결과 저장을 위한 테이블 구조 >

- 각 기관별로 하나의 테이블을 갖도록 하며, 분석 종류는 TAG를 사용하여 구분하며, 각 분석 결과에 대한 값은 VALUE에 넣는다.
- tag_list 테이블은 분석 종류를 나타내는 TAG, 해당 분석을 어느 기관에서 만들었는지 알려주는 TABLE_NAME, 해당 분석이 위험 상태가 되면 어떤 행동을 취해야는지 나타내는 ACTION, 해당 분석종류의 위험의 기준이 되는 값이 어디인지를 나타내는 WARNING_THRESHOLD, 해당 분석 종류의 최대로 안전했을 때의 값을 나타내는 MAXIMUM_SAFE_VALUE, 마지막으로 WARNING_THRESHOLD의 값이 MAXIMUM_SAFE_VALUE에 비해 몇프로 지점인지를 나타내는 UPPER_SAFETY_LIMIT를 갖는다.
- tag_combination_map 테이블은 맞춤형 SoS(State of Safety) 시스템에서 사용자가 조합한 TAG를 기반으로 TAG_SET을 만들며, 해당 TAG_SET에 대한 ID를 만든다. 이때, 다른 사용자가 이전에 만들어진 TAG_SET과 같다면, 해당 TAG_SET에 대한 정보는 기록하지 않는다.
- sos 테이블은 tag_combination_map에서 조합된 것을 기반으로 각 기관의 테이블에 참조하여 최근 값을 가져온 뒤, SoS를 계산하여 저장한다. 여기서, SET_ID는 TAG의 조합이며, VALUE는 선택된 TAG들을 사용하여 계산된 통합 SOS 값이며, F_VALUE는 각 TAG의 safety 값이다.

3. 맞춤형 SoS(State of Safety) 계산 처리 SW

□ 맞춤형 SoS(State of Safety) 알고리즘

- 분석 기관의 데이터는 1분 단위로 들어오기 때문에 SoS 계산 또한 1분단위로 시행되어야 한다. 따라서, SoS 알고리즘이 주기적으로 수행할 수 있게 contab을 사용하여 설정하였다.
- 맞춤형 SoS 알고리즘은 사용자의 요구에 따라 다양한 조합으로 사용이 가능하다. 따라서 사용자가 지정한 모든 경우에 대해서 SoS 값을 나타내어야 한다. 따라서, 본 연구에서는 tag_combination_map에 기록된 모든 SET_ID에 대해서 SoS 계산하며, SET_ID에 따른 결과를 sos 테이블에 저장한다.

맞춤형 SoS 알고리즘 코드
<pre> import psycpg2 import pandas as pd import math import os import json from dotenv import load_dotenv load_dotenv() # db info DB_NAME = os.environ.get("DB_NAME") HOST = os.environ.get("HOST") PORT = os.environ.get("PORT") USER_ID = os.environ.get("USER_ID") PASSWORD = os.environ.get("PASSWORD") # table info TAG_MAP = os.environ.get("TAG_MAP") TAG_LIST = os.environ.get("TAG_LIST") SOS = os.environ.get("SOS") def dictfetchall(cursor): """ 커서에서 모든 행을 딕셔너리 리스트로 반환합니다. Args: cursor (psycpg2.extensions.cursor): 커서 객체. Returns: list: 커서의 모든 행을 포함하는 딕셔너리 리스트. """ columns = [col[0] for col in cursor.description] return [dict(zip(columns, row)) for row in cursor.fetchall()] </pre>


```

def calc_safety_normalization(value, threshold, max_value):

    exp1 = 0.25 / math.pow(threshold - max_value, 2)
    exp2 = math.pow(value - max_value, 2)
    exp3 = 1/(exp1*exp2+1)

    return exp3

def calc_safety_score(value, tag, threshold, max_value):

    if max_value - threshold < 0:
        if value < max_value:
            safety_value = 1
        else:
            safety_value = calc_safety_normalization(value, threshold, max_value)

    else:
        if value > max_value:
            safety_value = 1
        else:
            safety_value = calc_safety_normalization(value, threshold, max_value)

    return safety_value

if __name__ == "__main__":

    conn = psycopg2.connect(host=HOST,
                            dbname=DB_NAME,
                            user=USER_ID,
                            password=PASSWORD,
                            port=PORT)

    with conn.cursor() as cur:
        query_set = f"""select * from {TAG_MAP}"""
        cur.execute(query_set)
        map_dict = dictfetchall(cur)

    # map_tag 정보를 갖고 있는 Dataframe
    map_df = pd.DataFrame(map_dict)

    for i in range(len(map_df["SET_ID"])):
        set_id = map_df.loc[i, "SET_ID"]
        tag_set = map_df.loc[i, "TAG_SET"]

    with conn.cursor() as cur:
        query_set = f"""select * from {TAG_LIST}"""
        cur.execute(query_set)

```

```

map_list_dict = dictfetchall(cur)

# tag에 따른 정보(X_100, X_80 등)를 갖는 Dataframe
map_list_df = pd.DataFrame(map_list_dict)

f_safety_dict = {}
for tag_name in tag_set:
    usad_row = map_list_df[map_list_df['TAG'] == tag_name]

    maximum_safe_value = usad_row["MAXIMUM_SAFE_VALUE"].iloc[0]
    warning_threshold = usad_row["WARNING_THRESHOLD"].iloc[0]
    tag_name = usad_row['TAG'].iloc[0]
    table_name = usad_row["TABLE_NAME"].iloc[0]

    #tag에 따른 value 값 가져오기
    with conn.cursor() as cur:
        query_set = f"""SELECT * FROM {table_name} WHERE "TAG" = '{tag_name}' ORDER BY
"TIMESTAMP" DESC LIMIT 1"""
        cur.execute(query_set)
        value_dict = dictfetchall(cur)

    value_df = pd.DataFrame(value_dict)

    # timezone 변경
    value_df["TIMESTAMP"] = value_df["TIMESTAMP"].dt.tz_convert('Asia/Seoul')

    norm_value = calc_safety_score(value=value_df["VALUE"].iloc[0], tag=tag_name,
threshold=warning_threshold, max_value=maximum_safe_value)
    f_safety_dict[tag_name] = norm_value

# SET_ID에 따른 데이터 저장

sos = math.prod(f_safety_dict.values())

insert_query = """INSERT INTO sos ("TIMESTAMP",
                                "VALUE",
                                "SET_ID",
                                "F_VALUE") VALUES(%s, %s, %s, %s)"""

f_safety_json = json.dumps(f_safety_dict)
with conn.cursor() as cur:
    cur.execute(insert_query, (value_df["TIMESTAMP"].iloc[0], sos, int(set_id), f_safety_json))
    conn.commit()

```

4. 웹페이지 기반 맞춤형 SoS(State of Safety) 연동 가시화

□ 개요

- o python flask로 구축된 웹 환경에서 여러 기관으로부터 수집된 데이터를 통해 조합된 SoS를 생성하고 조회해보기 위해 기관별 분석 데이터를 선택하고 선택한 데이터를 확인하고 생성버튼을 통해 선택한 분석 데이터를 통해 SoS를 계산하고 시각화한다.

□ 분석데이터 선택 및 시각화

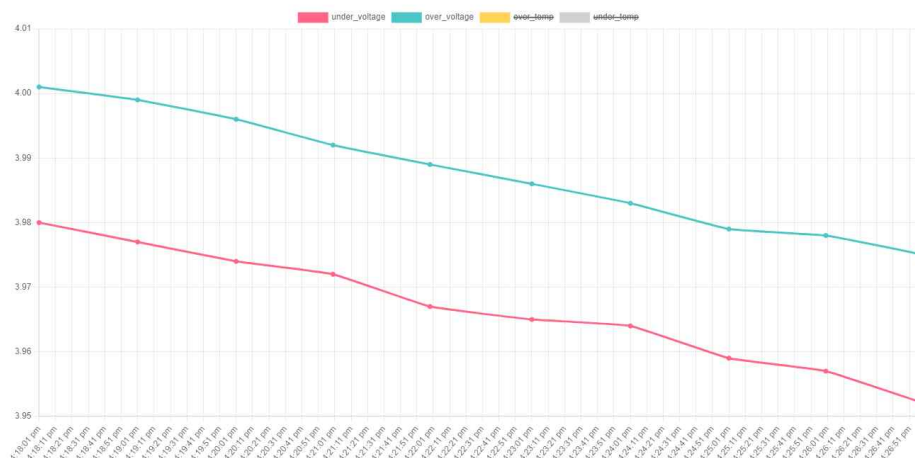
- o 연결된 데이터베이스에서 tag_list에 저장된 분석 데이터 종류를 출력하여 체크박스 형태의 목록을 보여준다. 최대 4개의 분석 데이터를 선택할 수 있으며 선택된 데이터들은 python코드에서 데이터베이스로 가장 최근에 저장된 값을 요청하여 불러와 다시 json형태로 웹쪽에 전달하여 테이블 형태로 보여준다.
- o 데이터를 선택한 후 ‘check’ 버튼을 누르면, 선택된 TAG 값이 Python을 통해 데이터베이스로 전송된다. 이후, 각 분석 테이블에서 데이터가 추출되고, 추출된 데이터는 ‘time’ 과 ‘value’ 라는 키 값을 가진 JSON 형태로 전달되며, 이 데이터는 테이블 하단에 위치한 멀티라인 차트에 가장 최근에 저장된 분석 데이터들로 표시된다.

선택된 분석컨테이너


TAG	Value	Action	MAX_count	SAFETY_VALUE	WARN_THRESHOLD
under_voltage	3.952	null	2	3.3	3.15
over_voltage	3.975	null	2	4	4.086
over_temp	32.4	null	2	40	50
under_temp	28.4	null	2	10	0

Check

Create

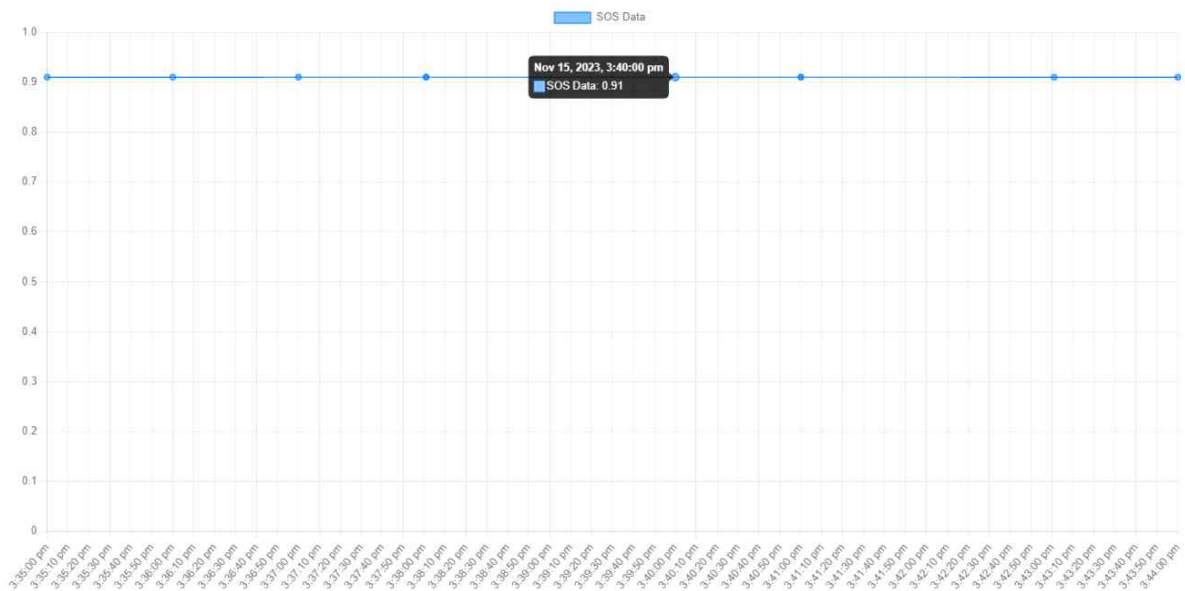


< 분석데이터 출력 예시 >

	ESS 테스트베드 및 안전SW 프레임워크 간 상호 연동 및 시나리오 기술서	
	프로젝트	대규모 분산 에너지 저장장치 인프라의 안전한 자율운영 및 성능 평가를 위한 지능형 SW 프레임워크 개발

□ 통합SoS 생성

- o Create 버튼을 누르면 선택된 분석데이터를 리스트형태로 만들어 데이터베이스에 해당 조합을 가지는 분석데이터에 대한 SET_ID를 알아내고 해당 SET_ID를 sos 가 저장되는 sos 테이블에서 조건문을 통해 최근 데이터를 가져온다.
- o 가져온 데이터는 json형태로 웹에 전달되고 자바스크립트를 통해 차트로 그려진다.



< SoS 데이터 출력 예시 >