


1단계(3차년도) 기술문서

(과제명) 대규모 분산 에너지 저장장치 인프라의 안전한 자율운영
및 성능 평가를 위한 지능형 SW 프레임워크 개발
(과제번호) 2021-0-00077

- 결과물명 : 가상 BMS의 SW 이상 감지 분석 보고서
- 작성일자 : 2023년 11월 09일

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

수행기관	성명/직위	확인
슈어소프트테크(주)	심정민/상무이사	

정보통신기획평가원장 귀하

목차

1. 개요
 - 1.1 BMS 및 가상 BMS의 구성
 - 1.2 가상 BMS의 수행 기능
 - 1.3 가상 BMS 소프트웨어 이상 구현
 - 1.4 가상 BMS 시스템 리소스 수집 목적
2. 가상 BMS의 에너지원 및 시스템 리소스 데이터의 수집 및 시각화
 - 2.1 가상 BMS 에너지원 및 시스템 리소스 측정/수집
 - 2.2 수집 데이터 시각화
3. DL 모델 구축
 - 3.1 DL 모델 탐색
 - 3.2 DL 모델 적용 결과
4. 결론

<그림 차례>

- 그림 1 가상 BMS 보드 구성
- 그림 2 가상 BMS에서 수집된 정상 CPU 사용량 데이터
- 그림 3 가상 BMS에서 수집된 메모리, 온도 데이터
- 그림 4 Demo 시각화 및 CPU 100% 유지 상태 이상탐지
- 그림 5 Demo 시각화 및 Deadlock 상태 이상탐지

1. 개요

1.1 BMS 및 가상 BMS의 구성

에너지저장장치(Energy Saving System; ESS)는 전력을 효율적으로 사용하게 하는 설비로, 주파수 조정, 신재생에너지 출력 안정화, 전력피크 저감 등에 활용함으로써 전력품질 향상 및 전력수급 위기에 대응이 가능한 설비이다.

에너지저장장치는 배터리(Battery), 전력변환장치(PCS; Power Conditioner System), 배터리관리시스템(BMS; Battery Management System), 전력관리장치(PMS; Power Management System)으로 구성되어 있다. 전력변환 장치는 입력되는 교류 전기를 직류로 변환하여 배터리 셀에 저장하며 방전 시에는 반대로 직류 전기를 교류로 변환하여 외부에 전력을 공급하는 역할을 한다. 배터리관리시스템은 MCU와 셀 모니터링 칩셋으로 구성 되어 있으며 각 셀에 직접 접근하면서 충/방전 제어 및 상태를 확인하는 역할을 한다. 전력관리장치는 모니터링 및 제어가 가능한 시스템으로 에너지저장장치가 충전과 방전을 수행할 조건을 설정할 수 있게 하여 보다 효율적으로 운영이 가능하도록 제어하는 요소이다.

실제 운용 중인 BMS에서는 SW 이상을 임의로 발생시키지 못하기 때문에 자체적으로 가상 BMS를 구축하였다. 실제 BMS와 유사한 임베디드 보드와 FreeRTOS를 기반으로 구성하였고, BMS에서 수행하는 여러 작업을 모사한 Task를 개발하였다. 이를 RTOS 상에서 시간, 우선순위에 따라 순차적으로 수행하는 가상 BMS로 구성하였다.

1.2. 가상 BMS의 수행 기능

자체적으로 개발한 가상 BMS는 다음과 같은 기능을 수행한다.

A. Monitoring Task

2차년도에 구성한 바와 같이 시스템 상태 및 리소스 데이터를 모니터링 하고, 생성된 데이터를 전송하는 작업을 수행한다.

B. 하위 BMS 데이터 로드 및 메모리 적재

하위 BMS로부터 데이터를 로드하고, 이 데이터를 메모리에 적재하는 과정으로 가상으로 Cell, Tray, Rack의 데이터 구조체를 생성하고, 이러한 구조체를 적재된 데이터와 함께 초기화 한다. 이 Task는 데이터를 Cell 단위로 구분하고, Tray 및 Rack의 레벨로 구조화하여 메모리에 저장하는 기능을 수행한다.

C. 하위 BMS에서 적재된 데이터에 대해 데이터 처리, 분석

전력 사용량, 온도, 전압 등과 같이 하위 BMS에서 적재된 데이터에 대해 데이터 처리, 분석 및 계산을 수행하여 상위 BMS로 데이터를 전달한다.

D. 하위 BMS 상태 감지 및 제어

하위 BMS에서 감지된 에러 또는 비정상 상태를 감지하고, 이러한 정보를 상위 BMS로 전달하게 된다. 따라서 Rack의 상태를 주기적으로 모니터링하고, 중요한 이벤트 또는 경고를 생성하고 추가로 에러 감지 시 하위 BMS의 동작을 제어하여 적절한 조치를 취하거나, 상위 BMS로 전달하여 상위 BMS에서 결정된 동작을 수행한다.

1.3 가상 BMS의 소프트웨어 이상 구현

이상 상태 유발 코드는 기본적으로 소프트웨어 이상 세 가지 상황으로 구성하였다.

첫 번째는 데드락(Deadlock) 유발 상황이다. 데드락은 주로 병렬 처리에서 발생하는 상황 중 하나로 멀티 스레드 또는 멀티프로세스 환경에서 프로세스나 스레드가 상호 작용하면서 서로를 기다리는 상태에 빠지는 현상을 의미한다. 이러한 상황에서는 아무런 진전도 없고, 프로그램이나 시스템이 무한 대기 상태에 빠질 수 있다.

일정 시간 정상코드를 동작시킨 후 Deadlock을 유발하는 코드를 동작하게 구성하였고, 모든 Task에서 아무런 동작을 실행시키지 않음으로써, 시스템이 무한 대기 상태에 빠진 상황을 모사한다.

두 번째 이상상황으로는 비정상적인 CPU 사용을 유발하는 패턴이다. BMS, RTOS의 특성상 여러 Task가 순차적으로 수행되면서 일정한 패턴의 CPU사용량을 관측 할 수 있다. CPU 사용량을 100%로 고정시키도록 구현하여 비정상적인 CPU 사용 상태를 모사한다.

세 번째 이상상황은 메모리 누수 시나리오이다. 5개 변수를 동적할당하는 함수를 2 회 호출하고, 해제하는 함수를 1회만 호출하여 지속적으로 메모리 누수가 발생하도록 구성하였다. 정상 시나리오가 일정시간동안 수행된 후 메모리 누수 시나리오가 실행되도록 구성하였다.

추가적으로 SIN 파 패턴, V 모양 패턴, 정상과 유사한 패턴의 데이터를 생성하여, 검증용 데이터에 추가하였다.

1.4 가상 BMS 시스템 리소스 수집 목적

실제 BMS의 테스트베드 환경에서 시스템 리소스 데이터 수집 결과 CPU, Stack, Heap 메모리 등의 주요 리소스에서 변화를 관찰 할 수 없었고, 다양한 분석기법이나, DL 모델을 적용하기엔 무리가 있다고 판단하였다, 또한 정상상태로만 동작하여 이상상태가 발생했을 시 리소스 데이터를 알 수 없다는 단점이 있었다.

따라서 자체적으로 가상의 BMS 시스템을 구축하고, 여러 환경과 여러 이상을 주입하여 리소스 데이터를 획득하고 이를 기반으로 학습 및 검증을 진행하여, 실제 환경에서 발생 할 수 있는 다양한 소프트웨어 이상을 선제적으로 대응 할 수 있도록 진행하였다.

2. 가상 BMS의 시스템 리소스 데이터의 수집 및 시각화

2.1 가상 BMS 시스템 리소스 측정/수집

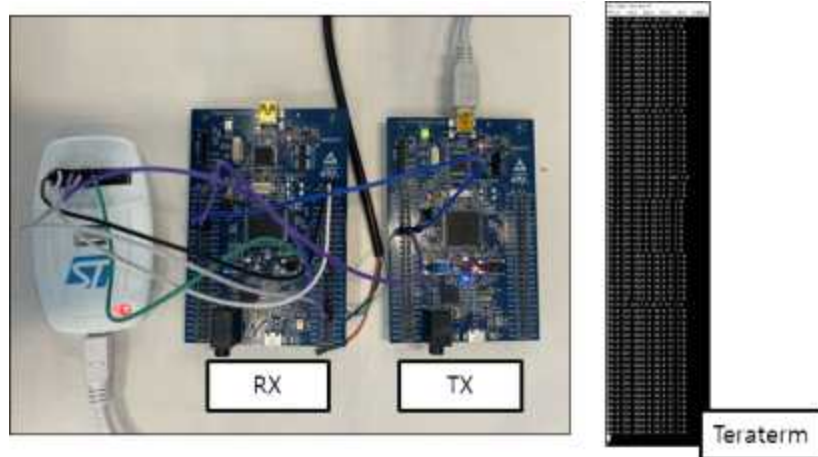


그림 1 가상 BMS 보드 및 Teraterm을 이용한 데이터 수집

앞서 구성한 가상 BMS에는 기본적으로 리소스 데이터를 모니터링하는 모듈이 내장되어있어, 이 부분을 이용하여 수집을 진행하였다. 측정되는 데이터 항목으로는 CPU 사용량, Stack 사용량, 남은 Heap 공간, 온도, 지연시간 등이 있으며, 측정되는 시스템 데이터를 Teraterm을 이용하여 파일

로 저장 후, 1차적으로 시간, 컬럼명 등의 전처리를 거친 후 CSV 파일로 NAS 서버로 업로드하여 데이터의 외부 유출을 차단하고 데이터 분석 시 용이하게 접근하여 사용할 수 있도록 처리하였다.

2.2 수집 데이터 시각화

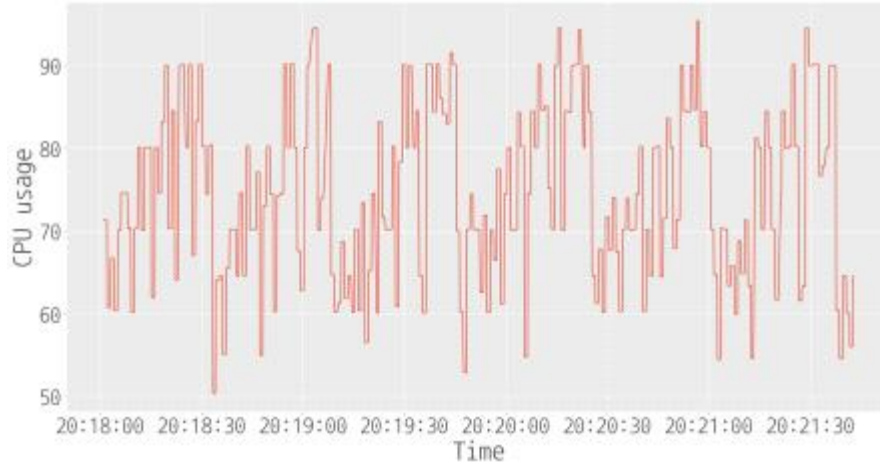


그림 2 가상 BMS에서 수집된 정상 CPU 사용량 데이터

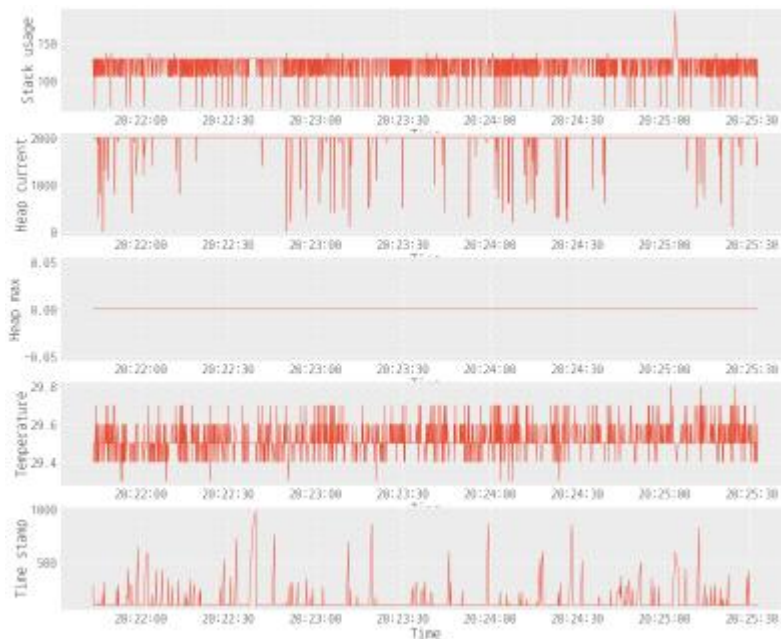


그림 3 가상 BMS에서 수집된 메모리, 온도 데이터

정상 Task가 수행되고 있을 때, 다음과 같이 일정한 패턴이 반복되는 형태로 CPU 사용량을 관찰 할 수 있다. 하지만 이외의 Stack, Heap 메모리 등은 실제 BMS 와 마찬가지로 뚜렷한 패턴이 존재하지 않는 것으로 확인되었다. 이는 임베디드 보드의 특성상 안정성을 위해 동적할당을 지양하는 디자인 패턴에 의한 것으로 확인하였다.



그림 4 소프트웨어 이상을 주입한 CPU 사용량 데이터

데드락 상태와 CPU 사용량을 100%로 유지시키는 소프트웨어 이상을 주입한 상태에서의 CPU 사용량 데이터이다. 데드락이 발생하는 경우 리소스 수집 Task까지 작동하지 않게 되므로 원본 데이터 상 모든 데이터가 수집되지 않고 비어있게 된다. 따라서 리소스 데이터가 수집 되지 않는 구간은 -1값으로 값을 채우도록 처리를 진행하였다.

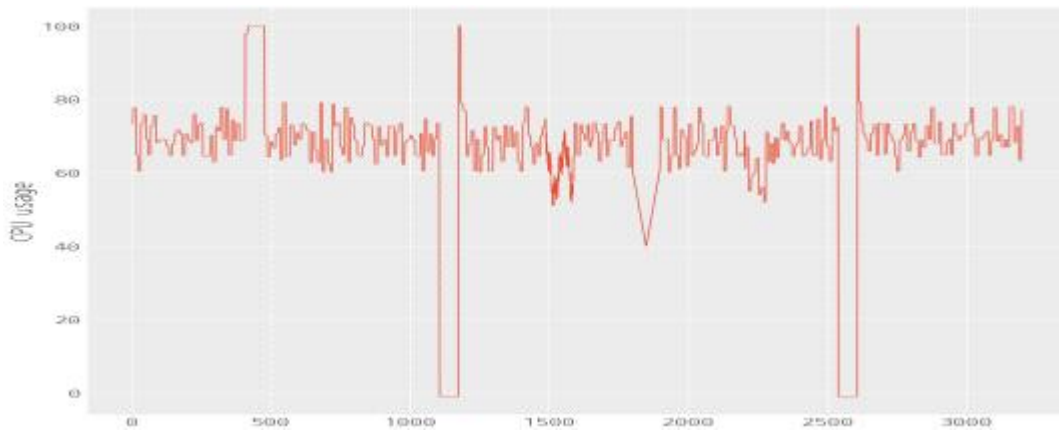


그림 5 이상 패턴을 생성한 후 주입한 CPU 사용량 데이터

CPU 이상 패턴 데이터를 생성하여 데이터에 추가한 그래프이다. 교착상태가 발생한 후 순차적으로 SIN 모양 패턴, V 모양 패턴, 정상과 유사한 패턴의 이상데이터가 추력되는 것을 확인할 수 있다.

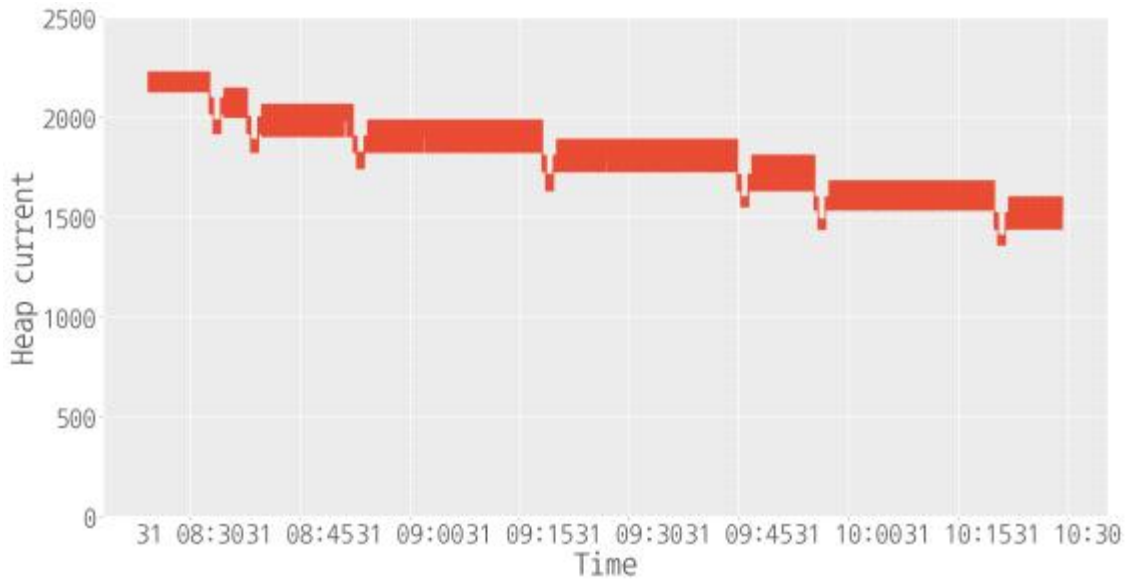


그림 6 메모리 누수 시나리오를 적용한 Heap 잔여량 데이터

마지막으로 메모리 누수 시나리오를 주입한 Heap 잔여 사용량 그래프이다. 정상시나리오가 일부 동작한 후 이후 메모리 할당, 일부만 해제를 반복하여 잔여 메모리량이 지속적으로 감소하는 것을 확인할 수 있다.

위 데이터에서 확인 가능한 것 처럼 이상 상황이 발생하는 경우 한 점으로 이상데이터가 발생하거나, 일정 구간의 이상데이터가 발생하는 두 가지가 존재한다. 따라서 이상탐지 모델을 설정할 때 각 이상 데이터를 탐지할 수 있는 모델을 탐색하여 선정하였다.

3. DL 모델 구축

3.1 DL 모델 탐색

시계열 데이터의 이상치는 크게 Point, Contextual, Collective Anomaly 3가지로 구분할 수 있다. 이 중 리소스 데이터에서 발생 가능성이 있는 이상데이터는 Point, Contextual Anomaly 두 가지 형태로 Point Anomaly 탐지를 위해 DeepAnt 모델, Contextual Anomaly 탐지를 위해 USAD 모델 두 가지를 선택하여 학습을 진행하였다.

USAD 모델의 경우 AE기반의 재구축 오차를 이용한 이상탐지 모델이다. AE 두 개를 배치하여 한 디코더는 원본과 가깝게 복원하고, 다른 디코더는 원본을 학습하여 원본과 다른 디코더를 거쳐 온 데이터를 구분하여 복원시키는 방식으로 학습을 진행한다. 최종적으로 학습에 사용되지 않은 데이터에 대해 패턴을 인식하여 이상탐지를 수행하게 된다.

DeepAnt 모델은 시계열 예측기, 이상 감지기 두 가지 모듈을 토대로 이상탐지를 진행하는 모델이다. 시계열 예측기는 Convolution Neural Network를 사용하여 Input 시계열 데이터 기반으로 이후의 값을 예측하고, 이상감지기의 경우 예측 된 값을 토대로 해당 시점이 정상인지 비정상인지 탐지를 진행하게 된다. DeepAnt 모델의 경우 Streaming Data에 최적화된 모델로서, CNN의 parameter sharing으로 준수한 성능을 가지고 있고 데이터의 양이 크지 않을 때도 동일하게 적용할 수 있다는 장점이 있다.

3.2 DL 모델 적용 결과

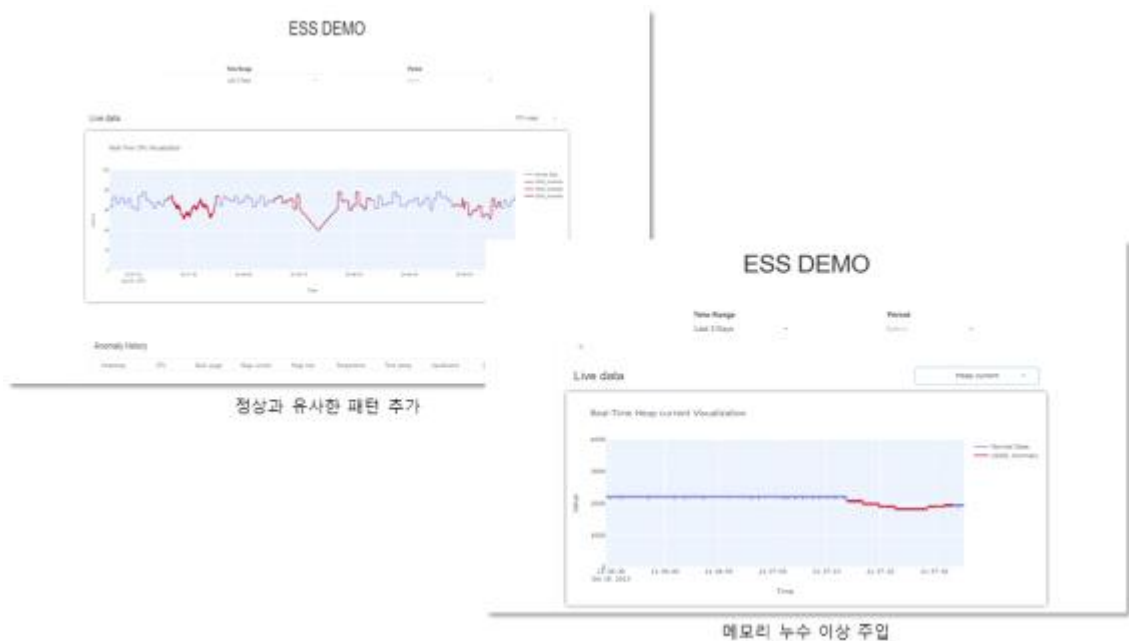


그림 7 Demo 시각화 및 CPU 100% 유지 상태 이상탐지



그림 8 Demo 시각화 및 Deadlock 상태 이상탐지

모델의 이상 탐지 예측 결과를 확인하기 위해 데모를 제작하여, 기 수집된 이상 데이터를 실제



Live 데이터처럼 흘러주는 방식으로 테스트를 진행하였다. 학습 결과에 따라 각 모델의 Threshold를 설정해주었고, DeepAnt 모델에서 탐지되는 Point와 USAD 모델에서 탐지되는 Contextual Anomaly를 각각 구분하여 표출하여 직관적으로 확인 할 수 있도록 구성하였다.

확인 결과, 현재까지 구성한 소프트웨어 이상은 정상적으로 탐지 되는 것을 확인하였고, 오탐 또한 발생하지 않았다.

4. 결론

최종적으로 가상 BMS를 구성하여, 리소스 데이터를 수집하고, 소프트웨어 이상을 구현하여 마찬가지로 데이터를 수집하였다. EDA 결과 Point, Contextual Anomaly 두 가지 형태로 이상 데이터가 발생하는 것을 확인하였고, 따라서 Deep Learning 모델 또한 두 가지를 적용하여 각각 탐지를 가능하도록 하였다. 각 모델의 학습 결과에 따라 Threshold를 지정해주고 사용자에게 알람을 전달할 수 있는 Demo를 설계하였고, API 형태로 모델 Input, Output을 전달하도록 하여 검증을 진행하였다. 실시간으로 데이터가 수집된다고 가정한 상황에서 확인해본 결과 이상데이터가 주어진 경우 실시간으로 이상탐지가 가능함을 확인하였다.

현재 이상 시나리오는 기초적인 이상 상태만 구현되어있고, CPU, Memory에 각각 모델이 학습되어 단변량의 이상탐지만 수행할 수 있다는 한계점이 존재한다. 따라서, 추가적인 소프트웨어 이상 시나리오를 생성하고, 리소스 데이터들간의 관계를 분석하여 여러 리소스 데이터를 통합한 이상탐지를 수행할 수 있는 모델로의 확대를 연구하고 있다. 마지막으로 외부 요인에 관한 데이터를 추가하여 내부 소프트웨어의 이상과 외부 환경에 의한 소프트웨어 오작동을 판단할 수 있는 모델로 발전시켜야 할 필요성을 확인하였다.