博士學位論文

분산 에지 컴퓨팅 환경에서 스웜학습 기반 예측 최적화 연구

濟州大學校 大學院

컴퓨터공학과

서 영 욱

2021年 12月

분산 에지 컴퓨팅 환경에서 스웜학습 기반 예측 최적화 연구

指導敎授 김 도 현

서 영 욱

이 論文을 工學 博士學位 論文으로 提出함

2021年 12月

서영욱의 工學 博士學位 論文을 認准함

審査委員長 ___김수균_____

委 員 ___최민석_____

委 員 ___한정훈_____

委 員 ___이경희_____

委 員 ___김도현_____

濟州大學校 大學院

2021年 12月

A Study on Predictive Optimization Based on Swarm Learning in a Distributed Edge Computing Environment

Rongxu Xu
(Supervised by professor Do-Hyeun Kim)

A thesis submitted in partial fulfillment of the requirement for the degree of Doctoral of Computer Engineering

2021. 12.

This thesis has been examined and approved.

.....
Thesis director, Soo Kyun Kim, Professor, Department of Computer Engineering

.....
Minseok Choi, Assistant Professor, Department of Telecommunication Engineering

.....
Jung-Hoon Han, Assistant Professor, Department of Telecommunication Engineering

.....
Kyounghee Daniel Lee, Professor, Department of Computer Engineering

.....
Do-Hyeun Kim, Professor, Department of Computer Engineering

December 2021
Date

Department of Computer Engineering
GRADUATE SCHOOL
JEJU NATIONAL UNIVERSITY

목 차

그림목차

표목차

국문초록

분산 에지 컴퓨팅 환경에서 스웜학습 기반 예측 최적화 연구

컴퓨터공학과 서영욱

지도교수 김도현

사물인터넷 (Internet of Things, IoT)을 통하여 사람 또는 사람들이 살고 있는 환경의 정보들이 수집되고 분석되어 사람의 삶의 질을 향상하는 서비스들이 개발되고 있다. 사물인터넷 디바이스는 컴퓨팅, 네트워크 혹은 전원 등의 제한적인 자원으로 장착된 센서, 구동체, 모바일 기기 등을 예로 들 수 있다. 그 외에도 제조사 별로 다양한 타입의 플랫폼, 통신 프로토콜을 지원하는 디바이스를 제공하여 사물인터넷 디바이스의 다양성에 공헌하고 있다.

기계학습 기술을 사물인터넷과 결합하여 데이터를 분석하고 사용자의 수요에 맞는 서비스를 제공하는 연구가 많은 관심을 받고 있다. 기계학습은 사물인터넷 디바이스를 통하여 수집한 데이터를 기반으로 사용자의 행동, 수요, 또는 상태를 인식하는 모델을 훈련하여 사물인터넷에 지능을 부여하는 역할을 한다. 기계학습의 학습작업은 컴퓨팅 자원에 대한 요구가 높을 뿐만 아니라 데이터도 많이 필요하다. 하지만 사물인터넷으로 수집된 데이터는 사용자의 민감한 개인 정보도 포함할 수 있어 프라이버시에 대한 요구도 제기되고 있다.

클라우드 컴퓨팅은 유연한 데이터 저장 공간과 컴퓨팅 자원을 제공하여 사물인터넷의 컴퓨팅 자원을 지원하는 해결책으로 사용되고 있다. 하지만 클라우드 컴퓨팅을 이용하여 모델을 생성할 경우 데이터를 에지로부터 클라우드로 전송할 때, 중앙집중적인 클라우드 컴퓨팅은 민감한 개인 정보에 대한 안전성을 보장하기 어렵다. 그리고 거리가 먼 데이터 센터의 네트워크 지연 현상도 문제로 제기되고 있다.

EI (Edge Intelligence)는 에지 컴퓨팅과 AI (Artificial Intelligent)의 통합을 가리킨

다. EI는 클라우드에 의존하지 않고 에지 네트워크의 자원을 최대한 활용하여 AI 통찰력을 얻는다. EI는 업계와 학계로부터 많은 관심을 받고 있다. 많은 기업들이 EI의 연구개발에 공헌하고 있다. EI를 이용하여 실시간 비디오 분석, 정밀 농업, 스마트 홈 등 많은 방면에서 AI 애플리케이션이 개발되고 있다.

개인 정보 데이터의 안정성을 보장하기 위하여 에지 컴퓨팅에서 연합학습 전략을 통하여 모델을 훈련한다. 하지만 로컬 에지 노드에서 훈련한 모델을 중앙서버에서 가중치 평균의 방식으로 업데이트하여 글로벌 모델을 학습하는 것은 중앙서버의 컴퓨팅 자원에 대한 요구가 높고, 에지 컴퓨팅 네트워크에 있는 모든 데이터의 지식을 학습할 수 없다는 단점이 있다.

본 논문에서 에지 컴퓨팅 환경에 개선된 스웜학습 (Swarm Learning, SL) 전략을 제시하고 PSO (Particle Swarm Optimization)기반 PMV (Predicted mean Vote) 최적화 기법을 제안한다. 글로벌 모델 업데이트 방식은 에지 컴퓨팅 네트워크에 있는 모든 노드들의 데이터를 이용하여 하나의 모델을 훈련하는 메커니즘이다. 중앙서버에서의 모델 업데이트 기능이 없으므로 컴퓨팅에 대한 요구가 줄어들고 에지 컴퓨팅 네트워크의 모든 노드들의 데이터를 활용하여 지능적인 모델을 생성한다. 데이터를 전달하는 대신 모델을 전달하여 데이터의 안정성도 보장한다.

에지 컴퓨팅에 인공지능을 통합하기 위하여 기계학습 기반의 최적화 기법을 통하여 사용자의 수요에 맞는 서비스를 능동적으로 제공한다. 에지 컴퓨팅을 사물인터넷과 통합하여 에지 디바이스의 컴퓨팅 자원을 충족함과 동시에 저장 공간도 확장한다. 그리고 에지 컴퓨팅 서비스를 직관적으로 관리할 수 있는 그래픽 유저 인터페이스 기반의 웹 앱을 제공하여 관리기능을 용이하게 한다.

먼저 분산 디지털 트윈 에지 컴퓨팅 환경을 구현한다. 웹 페이지를 통하여 에지 컴퓨팅 네트워크에서 실행하고 있는 에지 게이트웨이 및 사물인터넷 디바이스의 실행상태를 실시간으로 모니터링할 수 있도록 개발한다.

다음으로 개선한 스웜 학습을 통하여 PMV 모델을 훈련한다. 에지 컴퓨팅 네트워크에 등록한 기기들과 각각의 기기에서 수집한 로컬 데이터를 이용하여 PMV 예측 모델 훈련 전략을 제시한다.

그리고 스마트 홈의 PMV환경을 최적화 상태로 유지하기 위하여 PSO 알고리즘을 적용한다. 스마트 홈의 환경데이터를 수집하여 최적화 알고리즘에 적용하여 적절한

스마트 홈 환경을 유지할 수 있도록 한다.

마지막으로 개선한 스웜학습 전략의 성능을 측정하기 위하여 기존 연합/스웜 학습 전략과 함께 반복 회수를 각각 1, 10, 100으로 설정하여 모델을 훈련하는 실험을 진행하였다. 기존 연합/스웜 학습 전략을 통하여 훈련한 모델의 성능이 0.3 좌우가 최고 성능이지만 본고에서 개선한 스웜학습 전략으로 훈련한 모델은 0.2 좌우의 성능을 보여주고 있다. 스마트 홈 에뮬레이터 환경에서 PSO 최적화 알고리즘의 성능도 측정하였다. 최적화 기법을 적용하였을 때 스마트 홈 에뮬레이터의 열 쾌적 레벨은 -0.6에서 -0.8 사이를 유지하고 있는 반면 적용하지 않았을 때는 -1.2와 -1.4사이의 결과값을 보여주고 있다. 최적화 기법을 적용하였을 때 실내 온도와 습도가 열 쾌적 레벨을 최적화 기법을 적용하지 않았을 때보다 따뜻하게 유지하고 있음을 알 수 있다.

영문초록(Abstract)

A Study on Predictive Optimization based on Swarm Learning in a Distributed Edge Computing Environment

Rongxu Xu

Department of Computer Engineering, Jeju National University, South Korea

Today, contextual information about a person or their surroundings is collected and analyzed with the help of technologies such as the Internet of Things (IoT) to improve their lifestyle. This information is collected through IoT devices such as sensors, actuators, and other embedded devices. Nevertheless, IoT devices have limited computing, network, or power resources. Moreover, each manufacturer contributes to the diversity of IoT devices by manufacturing devices that support various types of platforms and communication protocols. Therefore, convergence research that combines machine learning with the Internet of Things to analyze data and provide services to meet users' needs and enhance the quality of experience is attracting a lot of attention.

Machine learning plays a role in employing intelligence in the IoT by training a model that recognizes the user's state, behavior, and demand using the contextual data collected through IoT devices. However, the challenge of machine learning in IoT realization is its need for a lot of data for model training and high computing resources. In addition, the realization of IoT has another challenge of user data privacy during contextual data collection.

The problem of scarce computing resources is solved using cloud computing and

its complementary technologies by making IoT flexible in terms of data storage space and computing resources. The prepared model is based on cloud computing, and data is transmitted from the edge to the cloud for predictive analysis. However, the centralized cloud computing architecture does not guarantee the safety of sensitive personal information and data privacy. In addition, cloud computing-based IoT infrastructure will lead to a rise in network latency due to being located in a distant data center. Recently, Edge Intelligence (EI) got a lot of attention from industry and academia researchers to perform intelligent data insights by employing most of the edge network resources without relying on the cloud. Using EI, machine learning, and AI, predictive mechanisms, applications have been developed in many fields such as real-time video analysis, precision agriculture, and smart home.

The model is trained through the federated learning method in edge computing to ensure the safety of personal information data. However, learning the global model by updating the model trained on the local edge node using the weighted average method requires numerous computational resources on the central server. Moreover, it cannot learn the knowledge of all the data in the edge computing network.

In this thesis, we propose an improved swarm learning (SL) method and particle swarm optimization (PSO) based on predicted mean vote (PMV) optimization techniques for edge computing environments. The global model update method trains a single model using data from all nodes in the edge computing network. Since there is no model update function in the central server, bandwidth and computing power demand are reduced. Thus a more intelligent model is created using data from all nodes in the edge computing network. In addition, instead of passing data, we pass the model to ensure the safety of the data.

The optimization technique based on machine learning actively provides the service that meets the user's demand. Furthermore, integrating edge computing into the IoT satisfies the computing resources of edge devices and expands storage

space. As proof of concept, we developed a graphical user interface (GUI)-based web application that can intuitively manage edge computing services to facilitate and control functionalities of the edge environment.

First, we implemented a distributed digital twin edge computing environment for performance comparison. Hence, the execution status of edge gateways and IoT devices running in the edge computing network can be monitored in real-time through the web application.

Next, the PMV model is trained through swarm learning using devices registered in the edge computing network and local data collected from each device. The smart home environment contextual data is collected and applied to the optimization algorithm to maintain an appropriate smart home environment. As an optimization solver, the PSO algorithm is utilized to maintain the PMV environment of the smart home in an optimized state.

Finally, for comparative performance analysis, the swarm learning parameters were optimized using different experimental setups, such as training the model by setting the number of rounds to 1, 10, and 100. The model trained through the existing federated/swarm learning strategy is the best in terms of performance of 0.3, but the model trained with the swarm learning approach improved the performance to 0.2. Furthermore, the performance of the PSO optimization algorithm in a smart home-based simulation environment was also measured using the developed emulator. Moreover, the optimization technique also impacts the thermal comfort level of the smart home emulator by setting it between -0.6 and -0.8; without optimization, the thermal comfort level result is between -1.2 and -1.4. The optimization technique adaptation results show that indoor temperatures and humidity maintain a higher thermal comfort level, which is not the case without adaptation.

약어표

IoT	Internet of Things
M2M	Machine to Machine
D2D	Device to Device
OCF	Open Connectivity Foundation
CoAP	Constrained Application Protocol
HTTP	Hyper Text Transfer Protocol
DNN	Deep Neural Network
ANN	Artificial Neural Network
ML	Machine Learning
AI	Artificial Intelligence
DL	Deep Learning
SL	Swarm Learning
ECS	Edge Computing Supervisor
EGMS	Edge Gateway Management Supporter
TMA	Task Management Agent

I. 서론

우리는 인공지능 (Artificial Intelligence, AI) 기술이 고속으로 발전하고 있는 시대에 살고 있다. 알고리즘, 컴퓨팅 능력 및 빅데이터의 발전에 힘입어 딥러닝 (Deep Learning)[1]은 컴퓨터 비전, 음성 인식 및 자연어 처리에서 체스(예: 알파고) 및 로봇 공학 [2]에 이르기까지 광범위한 분야에서 주목받는 발전을 이루었다. 이러한 획기적인 발전으로 인해, 지능형 개인 비서, 개인 맞춤형 쇼핑 추천, 비디오 감시, 스마트 가전제품 등의 일련의 지능형 애플리케이션들이 엄청난 인기를 얻고 있다. 이러한 지능형 애플리케이션들이 사람들의 생활 방식을 풍부하게 하고, 생산성을 향상시키며, 사회적 효율성을 높인다는 것은 널리 인정받고 있다. AI 개발을 촉진하는 핵심인 빅데이터는 데이터 소스가 최근 클라우드 데이터 센터에서 모바일 기기 및 사물인터넷 기기 등 점점 더 널리 보급되는 엔드 디바이스로 급격하게 변화하고 있다. 기존에는 온라인 쇼핑 기록, 소셜 미디어 콘텐츠, 비즈니스 정보학 등의 빅데이터가 주로 클라우드 데이터 센터에서 생성되어 저장되고 있다. 그러나 모바일 컴퓨팅과 사물인터넷이 확산되면서 이제는 그 흐름이 역전되고 있다. 특히 Cisco는 2021년까지 네트워크 에지에 있는 모든 사람, 기계 및 사물에 의해 관련 데이터가 약 850 ZB 생성될 것으로 예측한다 [3]. 이와는 대조적으로, 전 세계 데이터 센터 트래픽은 2021년까지 20.6 ZB에 도달할 것이다.

딥러닝의 높은 정확도는 딥러닝의 훈련 및 추론 모든 단계에서 높은 컴퓨팅과 많은 메모리를 요구한다. 딥러닝 모델을 훈련하는 것은 반복적으로 개선하는 수백만 개의 매개 변수로 인해 공간과 계산 비용이 많이 든다. 추론 또한 입력 데이터의 잠재적으로 높은 차원(예: 고해상도 이미지)과 입력 데이터에 대해 실행해야 하는 수백만 개의 계산으로 인해 계산 비용이 많이 든다. 높은 정확도와 높은 자원 소모가 딥러닝의 특성이다 [4].

딥러닝의 컴퓨팅 요구사항을 충족하기 위한 일반적인 접근 방식은 클라우드 컴퓨팅을 활용하는 것이다 [5]. 클라우드 자원을 사용하기 위해서, 데이터를 네트워크의 에지에서 클라우드로 이동시켜야 한다. 데이터를 소스에서 클라우드로 이동시키

는 방법에는 다음과 같은 문제가 있다. 1) 지연 현상: 시간이 중요한 응용프로그램은 실시간 추론이 중요하다. 예를 들어 자율주행차는 카메라 프레임을 신속하게 처리해 장애물을 감지하고 피하는 행동을 실행하고, 음성 기반 보조 애플리케이션은 사용자의 쿼리를 빠르게 구문 분석・이해하고 응답을 반환해야 한다. 그러나 추론 또는 훈련을 위해 데이터를 클라우드로 보내면 네트워크에서 추가 대기열 및 전파 지연이 발생할 수 있으며 실시간 대화형 애플리케이션에 필요한 엄격한 종단 간 낮은 지연 시간 요구사항을 충족할 수 없다. 예를 들어, 실제 실험에 따르면 카메라 프레임을 AWS 서버로 오프로드하고 컴퓨터 비전 작업을 실행하는 데 200ms 이상의 시간이 소요된다[6]. 2) 확장성: 데이터를 센서로부터 클라우드로 보내면 연결된 디바이스 수가 증가함에 따라 클라우드에 대한 네트워크 액세스 병목 현상이 생길 수 있어 확장성 문제가 발생한다. 모든 데이터를 클라우드에 업로드하는 것은 네트워크 리소스 활용도 측면에서도 비효율적이며, 특히 딥러닝이 모든 데이터를 필요로 하지 않을 때 더욱 문제이다. 3) 개인 정보: 데이터를 클라우드로 보내면 데이터에 포함된 사용자의 개인 정보 보호 문제가 발생할 수 있다. 사용자는 중요한 정보(예: 얼굴 또는 음성)를 클라우드에 업로드할 때, 클라우드 또는 애플리케이션이 데이터를 어떻게 사용하는지 걱정할 수 밖에 없다. 예를 들어, 최근 뉴욕의 스마트 시티에 카메라와 기타 센서를 배치한 것을 들 수 있는데, 이는 개인정보 보호 감시자들의 심각한 우려를 낳았다 [7].

에지 컴퓨팅은 위에서 설명한 지연 시간, 확장성 및 개인 정보 보호 문제를 해결할 수 있는 실행 가능한 솔루션이다. 에지 컴퓨팅에서 컴퓨팅 자원은 최종 장치와 가까운 곳의 에지 네트워크에서 컴퓨팅 능력을 제공한다 [8]. 예를 들어 에지 계산 노드는 셀룰러 기지국, IoT 게이트웨이 또는 캠퍼스 네트워크와 함께 배치될 수 있다.

에지 컴퓨팅 자원이 엔드 디바이스의 가까운 위치에 배치되기 때문에 대기 시간을 줄여 실시간 서비스를 지원할 수 있게 한다. 확장성 문제를 해결하기 위해 에지 컴퓨팅은 엔드 디바이스, 에지 컴퓨팅 노드 및 클라우드 데이터 센터의 계층적 아키텍처를 지원한다. 컴퓨팅 리소스를 제공하고 클라이언트 수에 따라 확장할 수 있으며 중앙 위치에서 네트워크 병목 현상을 방지할 수 있다. 개인 정보 보호 문제를 해결하기 위해 에지 컴퓨팅을 사용하면 데이터를 소스에 가깝게 분석할 수 있으므로

공용 인터넷의 횡단을 방지하고 개인 정보 보호 및 보안 공격에 대한 노출을 줄일 수 있다. 에지 컴퓨팅은 위에서 설명한 지연 시간, 확장성 및 개인 정보 보호 이점을 제공할 수 있지만, 에지에서의 딥러닝을 실현하는 데 몇 가지 주요 과제가 남아 있다 [9]. 첫 번째 주요 과제는 제한된 에지 컴퓨팅 리소스에 딥러닝의 높은 리소스 요구사항을 수용하는 것이다. 두 번째 과제는 에지 장치의 다양한 처리 능력과 동적 네트워크 조건에서의 연결문제이다. 마지막으로 에지 컴퓨팅이 네트워크 에지에서 로컬 데이터를 유지하여 자연스럽게 프라이버시를 개선하더라도, 일부 데이터는 여전히 에지 장치 간에 교환해야 하는 경우가 많기 때문에 개인 정보 보호는 여전히 도전과제로 남아 있다.

스마트폰, 사물인터넷 센서 등 엔드 디바이스는 딥러닝을 활용해 실시간 분석을 실행하고 딥러닝 모델 훈련에 활용하는 데이터를 생성한다. 그러나 딥러닝 추론 및 훈련이 빠르게 실행되려면 상당한 계산 자원이 필요하다. 컴퓨팅 노드들이 엔드 장치에 가까이 배치된 에지 컴퓨팅은 에지 장치에 대한 딥러닝의 높은 계산 및 짧은 지연 시간 요구사항을 충족할 수 있는 실용적인 방법이며 개인 정보 보호, 대역폭 효율성 및 확장성 측면에서도 추가적인 이점을 제공한다 [10].

에지 컴퓨팅과 AI의 결합은 새로운 연구 영역, 즉 “EI“ 또는 “edge AI“ [11], [12]를 탄생시켰다. EI는 클라우드에 의존하는 대신 광범위한 에지 리소스를 최대한 활용하여 AI 통찰력을 얻는다. 특히 EI는 업계와 학계로부터 많은 관심을 받고 있다. 구글, 마이크로소프트, 인텔, IBM을 포함한 주요 기업들은 AI를 에지로 통합하는 데 있어, 에지 컴퓨팅의 장점을 입증하기 위한 시범 프로젝트를 내놓았다. 이러한 노력은 실시간 비디오 분석 [13], 인지 지원 [14]에서 정밀 농업, 스마트 홈 [15] 및 산업에서의 사물인터넷 [16]에 이르기까지 광범위한 AI 애플리케이션을 활성화시켰다.

인공지능은 인간이 하는 것처럼 업무를 수행할 수 있는 지능형 기계를 만드는 접근법이다. 기계학습은 AI의 목표를 달성하기 위한 효과적인 방법이다. 의사결정 트리, K-평균 클러스터링 및 베이지안 네트워크 등 많은 ML 방법론이 실제 세계에서 얻은 데이터를 기반으로 분류 및 예측을 하도록 개발되고 있다. 기존 ML 방법 중 인공 신경망 (Artificial Nerual Network, ANN)[17]을 활용하여 데이터의 심층 표현을 학습함으로써, 딥러닝은 이미지 분류 및 얼굴 인식 등을 포함한 여러 작업에서 놀라운 성과를 거두었다. ANN 딥러닝 모델을 사용하여 일련의 계층으로 구성되어

있기 때문에, 이 모델을 심층 신경망(Deep Neural Network, DNN)이라고 한다.

에지에서 딥러닝 모델을 훈련하는 방법은 데이터 센터에서의 분산 DNN 훈련방법을 모방한다. 데이터 센터에서 모델의 학습은 여러 작업자를 이용하여 실행한다. 데이터의 병렬화 또는 모델의 병렬화를 이용하여 각 작업자에서 모델을 학습한다. 데이터 병렬화는 데이터를 분할하여 각 작업자로 분배하는 것이고 모델 병렬화는 모델의 일부를 각 작업자로 분배하는 것이다. 데이터 병렬화 [18]는 분산 DNN학습으로 실제 시스템에서 널리 사용된다. 데이터 병렬에서 각 작업자는 데이터 세트의 로컬 파티션의 기울기 (gradients)를 계산하고, 계산된 기울기 중앙 서버에 의해 수집되며, 이를 업데이트하여 다시 로컬 작업자로 돌려보낸다.

에지 네트워크에서 딥러닝 모델을 학습하는 방법으로 연합학습이 있다 [19]. 연합학습은 프라이버시 문제를 최적화하는 데 전념한다. 연합학습은 여러 클라이언트가 생성한 데이터를 기반으로 DNN 모델을 학습할 때 프라이버시를 보존하기 위한 새로운 접근법이다. 연합학습은 훈련을 위해 원시 데이터를 중앙 집중식 데이터 센터에 집계하는 대신 클라이언트 (사물인터넷 디바이스, 모바일 장치)에 배포된 원시 데이터를 그대로 두고 로컬로 계산된 업데이트를 집계하여 서버에서 공유 모델을 학습한다. [20]은 연합학습을 위한 반복 모델 평균화의 FedAvg 방법을 제안한다. 여기서 반복 모델 평균은 클라이언트가 로컬에서 SGD (Stochastic Gradient Descent) 방법으로 모델을 업데이트한 이후, 중앙 서버가 가중치를 사용하여 결과 모델을 평균화하는 것을 의미한다. SGD는 딥러닝에서 간단하지만 널리 사용되는 최적화 방법으로 전체 데이터 세트의 매우 작은 하위 집합(미니배치)에 대한 경사도를 업데이트하는 방법이다.

연합학습에서 로컬 모델을 중앙서버에서 집계하여 업데이트한다. 중앙서버에서 모델을 업데이트하는 방식은 훈련된 모델의 가중치 값을 평균하는 방법이다. 그러므로 모든 데이터의 지식을 학습할 수 없다. 또한 중앙 집중식의 처리 방식은 단일 장애 지점의 문제가 있다. 기존 스웜학습 [21]은 전문 중앙서버가 모델 매개 변수를 수집하여 병합하는 반면 탈중앙화된 하드웨어 인프라를 이용하고, 표준화된 AI 엔진 기반으로 분산 기계학습을 실행하고, 구성원 간의 리더를 동적으로 선택하여 모델 매개 변수를 병합한다. 하지만 이 방법 또한 선택한 리더가 모델 매개 변수 즉 가중치 값을 평균과 같은 방식으로 업데이트하여 글로벌 모델을 생성하기 때문에 모든

데이터의 패턴을 인식할 수 없다. 또한 연합학습이든 스웜 학습이든 학습에 참여하는 에지 노드의 수가 많으면 중앙서버에서 모든 모델을 수집하여 글로벌 모델을 학습하게 되어 병목현상의 문제가 발생할 수 있다.

본 논문에서 분산 디지털 트윈 메커니즘을 구축하고 에지 컴퓨팅 환경에 개선된 스웜학습 전략을 제시하고 PMV 예측을 통해 성능의 우수성을 확인하고 PSO (Partial Swarm Optimization) 기반 PMV (Predicted mean Vote) 최적화 메커니즘을 제안한다. 기존 스웜학습의 문제를 해결하기 위하여 새로운 글로벌 모델 업데이트 방식을 구현한다. 본고에서 제안한 스웜학습에서의 글로벌 모델 업데이트는 에지 컴퓨팅 네트워크에 있는 모든 노드들의 데이터를 이용하여 하나의 모델을 훈련하는 방식이다. 개선한 스웜학습 전략은 탈중앙화의 방법으로 모델을 업데이트하여 기존 연합/스웜 학습 전략보다 컴퓨팅 자원에 대한 요구가 낮다. 그리고 에지 컴퓨팅 네트워크에서 데이터를 전달하는 대신 모델을 전달하여 데이터의 안전성을 보장한다. 에지 지능을 구현하기 위하여 에지 컴퓨팅환경에서 PSO 기반 PMV 최적화 서비스를 제공하여 스마트 홈 사용자의 열 쾌적성을 유지한다. 그리고 에지 컴퓨팅을 분산 디지털 트윈 메커니즘을 통하여 관리하도록 그래픽 유저 인터페이스 기반의 웹 앱을 제공하여 서비스의 관리를 쉽게 한다.

먼저 에지 컴퓨팅을 구성하는 에지 게이트웨이 및 사물인터넷 디바이스를 등록하여 가상의 디지털 객체를 생성하여 분산 디지털 트윈 에지 컴퓨팅 환경을 구현한다. 생성한 가상의 디지털 객체를 통하여 웹 페이지에서 에지 컴퓨팅 네트워크에서 실행하고 있는 에지 게이트웨이 및 사물인터넷 디바이스의 실행상태를 실시간으로 모니터링할 수 있다. 그리고 데이터베이스를 통하여 실행과정에서 수집한 데이터를 저장하고 시각화하여 분산 에지 컴퓨팅 가상공간에서 현실세계의 물리적 사물의 상태를 확인할 수 있다.

다음으로 기존 연합/스웜 학습 전략을 분석하고 개선된 스웜학습 전략을 제안하고 라즈베리파이 기반 분산 에지 지능 환경을 구축한다. 개선된 스웜학습 전략은 에지 컴퓨팅 네트워크 환경에서 수집한 데이터의 안전성을 보장하고 제한된 자원을 이용하여 모델을 훈련한다. 데이터의 안전성을 보장하기 위하여 모델을 에지 게이트웨이 기기들 사이에서 전달한다. 각각의 에지 게이트웨이는 로컬 데이터를 이용하여 전달 된 모델을 훈련하여 모델의 성능을 향상한다. 개선한 스웜학습 전략의 성능을

분석하기 위하여 개선한 스웜학습 전략과 기존 연합/스웜 학습 전략을 통하여 훈련한 모델의 성능 비교를 한다. 모델을 훈련하는 반복 회수를 각각 1, 10, 100으로 설정하여 모델을 훈련한다. 기존 연합/스웜 학습 전략을 통하여 훈련한 모델의 성능은 1, 10회에서 최고 성능이 0.5 좌우이고 100회에서 최고 성능이 0.3이다. 하지만 본고에서 개선한 스웜학습 전략으로 훈련한 모델은 항상 최고의 성능이 0.2 좌우의 성능을 보여주고 있다.

그리고 스마트 홈의 PMV 환경을 최적의 상태로 유지하기 위하여 PSO 알고리즘을 적용한다. 저희가 제안한 스웜학습 전략을 이용하여 훈련한 PMV 예측 모델을 이용하여 스마트 홈 환경의 열 쾌적 상태를 확인한다. PSO 알고리즘은 PMV 예측 모델을 이용하여 스마트 홈의 최적 상태 파마미터 즉 온도와 습도를 찾는다. PSO 알고리즘을 적용하기 위하여 스마트 홈의 환경데이터를 생성하는 에뮬레이터를 구현한다. 에뮬레이터에 PSO 알고리즘을 적용하여 스마트 홈의 PMV 환경을 최적의 상태로 유지하는 실험을 진행한다. 스마트 홈에 PMV 최적화 기법을 적용하였을 때 스마트 홈 에뮬레이터의 열 쾌적 레벨은 평균 -0.7를 유지하고 있고 적용하지 않았을 때는 평균 -1.2의 결과값을 보여주고 있다. 그리고 히터 전력 결과는 최적화 기법을 적용하였을 때 36,09 최적화 기법을 사용하지 않았을 때 35.93로 최적화 기법을 적용하였을 때 아주 미세한 차이로 에너지를 많이 사용하고 있다.

서론에 이어 2장 관련연구에서는 에지 컴퓨팅, EI, 디지털 트윈, 기계학습 그리고 스마트 홈 관련 최적화 알고리즘에 관하여 설명한다. 3장에서는 분산 디지털 트윈 에지 컴퓨팅 환경에 대하여 설명하고, 디지털 트윈 에지 컴퓨팅 환경에서 사물인터넷으로 데이터를 수집하는 실험을 진행한다. 4장에서는 기존 연합/스웜 및 개선된 스웜학습 기반 PMV 예측 메커니즘의 모델 구조, 모델 훈련 방식 및 성능 평가를 한다. 5장에서는 스마트 홈의 열 쾌적성을 최적의 상태로 유지하는 PSO 알고리즘 기반 최적화 메커니즘에 관하여 설명하고, 에지 게이트웨이 스마트 홈 에뮬레이터를 구현하여 본고에서 제안한 PSO 알고리즘의 성능을 평가한다. 6장에서는 결론을 맺는다.

II. 관련연구

1. 에지 컴퓨팅

사물인터넷은 초를 단위로 짧은 시간에 엄청난 양의 데이터를 생성하는 무수한 서로 연결된 장치를 통해 전 세계의 많은 측면에서 혁명을 일으켰다. 사물인터넷이 제공하는 작동 및 감지 서비스의 실행 방식은 원활한 인터넷 연결의 보장으로 더욱 발전한다. 사물인터넷을 통하여 데이터에 대한 액세스와 정보의 가용성이 향상되어 새로운 가능성과 기회가 창출되고 있다 [22]. 사물인터넷의 미래 비전은 임베디드 시스템을 가진 모든 일반적인 사물들의 연결성으로 볼 수 있다. 현재 90억 개 이상의 장치가 인터넷에 연결되어 있으며 그 수는 나날이 증가하고 있다 [23]. 향후 몇 년 안에 그 숫자는 500억까지 증가할 것으로 예상된다.

사물인터넷은 기존 인터넷 인프라를 확장하거나 디바이스의 전용 인터넷 인프라를 처음부터 다시 구축하는 두 가지 방식으로 개발할 수 있다. 두 접근방식 모두 나름의 이점과 약점을 가지고 있다 [24]. 사물인터넷은 또한 스마트 제조시스템 등 기계나 기기가 IP주소를 부여받고 기기간 (Machine to Machine, M2M) 통신 기능과 글로벌 네트워크를 통하여 연결되는 산업용 사물인터넷이 있는가 하면, M2M 통신 외에도 와이파이 (WiFi), 지그비 (ZigBee), 블루투스 (Bluetooth) 등의 통신기술을 활용한 근거리 통신망을 기반으로 기기 간 통신이 이뤄지는 소비자 사물인터넷도 있다.

사물인터넷의 엄청난 성장으로 인하여 헬스케어 시장 점유율 30.03%, 스마트 제조 시스템 시장 점유율 40.2%, 소매 시장 점유율은 8.3%, 보안 시장 점유율은 7.7%를 기록했다 [25]. 사물인터넷의 사용사례로는 스마트 홈, 스마트 공장, 스마트 빌딩, 스마트 그리드 등을 포함한다. 주어진 시나리오에서 로컬 수준의 장치 간 통신은 게이트웨이에 의해 관리되고 인터넷을 통하여 글로벌 연결을 제공한다. 근거리 통신망은 게이트웨이 노드를 통해 상호 연결된다. 노드에는 게이트웨이 도메인에서만 사용할 수 있는 주소가 부여됨으로 인하여 주소의 낭비를 방지한다.

지배적인 클라우드 인프라에서 클라우드의 한계와 결함을 극복하기 위해 포그/에지 (Fog/Edge) 컴퓨팅이 등장했다. Fog와 Edge 컴퓨팅은 유사한 개념을 공유하고 서로 교환적으로 사용되기 때문에 연구자들에 의해 널리 적용되고 있다. Fog에 대한 개념은 사물인터넷 기기에 가까운 곳으로의 클라우드 확장에 기초한다. 기기의 엄청난 증가로 인해 현재 센서는 매일 전 세계 데이터의 40%를 생성하고 있다. 클라우드 아키텍처에는 시간이 중요한 애플리케이션의 경우 지연 시간 및 대역폭 문제가 있다 [30]. 앞서 언급한 문제를 개선하기 위해 포그/에지 컴퓨팅은 사물인터넷 기기와 클라우드 사이에서 중간 계층 역할을 한다. 핵심 아이디어는 자원을 네트워크 에지 쪽으로 기기/사용자에게 더 가까이 가져오는 것이다. 그림 2.1은 클라우드 데이터 센터 포그 노드 및 에지 장치의 아키텍처를 보여준다.

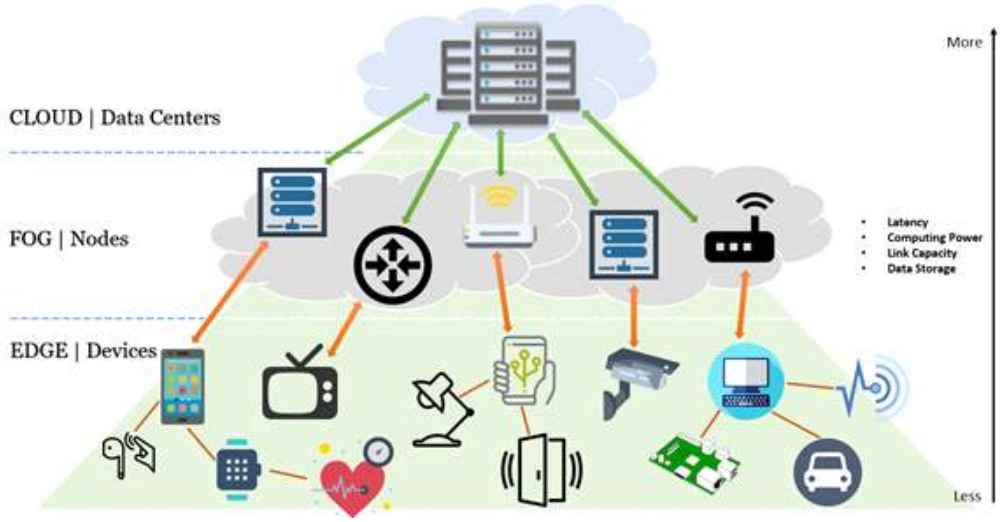


그림 2.1 사물인터넷을 구성하는 에지, 포그, 클라우드 계층적 구조

에지 컴퓨팅은 대기 시간을 줄이면서 사용자와 가까운 원격 지리적 위치에서 많은 서비스를 제공한다. 에지 플랫폼은 에지 노드에서 분산화를 통해 응답 시간을 개선하여 기존 시스템을 개선한다. 머신 러닝과 빅데이터 분석의 통합은 에지/클라우드 패러다임이 스마트 공간에서 향상된 효율성과 실시간 제어를 달성하는 데 도움이 될 수 있다. 유사한 패러다임으로 포그 컴퓨팅 [31] 및 모바일 에지 컴퓨팅 [32] 등이 있다. 그림 2.2는 에지 컴퓨팅 프레임워크의 구조에 대한 개요이다.

에지 컴퓨팅 패러다임은 클라우드 컴퓨팅의 문제인 지연 시간과 대역폭 문제를

해결하는 솔루션으로 제안되고 있다. 에지 컴퓨팅 패러다임 [33]-[35]은 데이터를 네트워크의 에지 근처에서 분석할 수 있다. EdgeX Foundry[36]-[39]는 클라우드 컴퓨팅 기능 변화에 대응하기 위한 솔루션 중 하나이다. EdgeX Foundry 프레임워크는 리눅스 foundation dell에 의해 소개되었고, 에지 컴퓨팅 패러다임을 실현하기 위해 마이크로 서비스 아키텍처를 채택한다. EdgeX Foundry에서 모든 마이크로 서비스는 일반적으로 경량 가상화 컨테이너로 구현되고 서로 분리되어서 EdgeX Foundry 프레임워크의 유지 보수성과 확장성을 보장한다. EdgeX Foundry 프로젝트[26-30]는 앞서 논의한 기존 사물인터넷 플랫폼과 달리 어떠한 운영 체제나 프로그래밍 언어에도 의존하지 않는다. EdgeX는 네트워크 에지에서 컴퓨팅 및 처리 기능을 제공하여 클라우드로 전송하는 데이터를 미리 처리하여 대역폭 및 대기 시간 문제를 해결한다. 에지 컴퓨팅은 사물인터넷의 모든 잠재력을 가능하게 하는 기술로 업계와 학계의 주목을 받고 있다 [39].

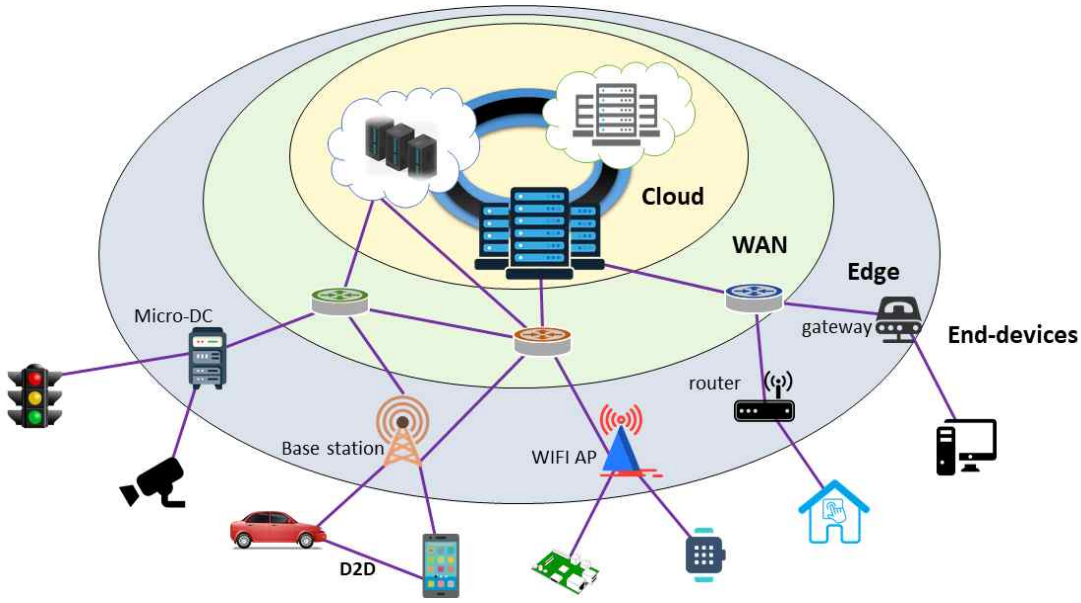


그림 2.2 에지 컴퓨팅 네트워크를 구성하는 사물인터넷 기기 및 구조

에지 컴퓨팅의 개발을 효율적으로 달성하기 위해, 몇 가지 이니셔티브(initiatives)가 설립되고 있다. 풍부한 리소스를 가진 컴퓨터 또는 클러스터를 구성하는 컴퓨터로 구성된 Cloudlet [40]은 가까운 모바일 장치에서 사용할 수 있도록 짧은 대기 시간으로 리소스 집약적인 애플리케이션을 지원하기 위해 네트워크 에지에 위

치한 소규모 버전의 클라우드 데이터센터이다. 오픈 에지 컴퓨팅(Open Edge Computing, OEC) 이니셔티브는 인텔 (Intel), 화웨이 (Huawei), Vodafone이 Carnegie Mellon University [73]와 손잡고 Cloudlet 생태계 개발을 위해 시작되고 있다. 모바일 에지 컴퓨팅(Mobile Edge Computing, MEC) [41]은 유럽 통신 표준 연구소(European Telecommunications Standards Institute, ETSI) 산업 규격 그룹(Industry Specification Group, ISG)에 의해 제안되었으며, 모바일 네트워크 에지에서 실행되어 모바일 가입자들에게 짧은 지연 시간과 매우 효율적인 네트워크 운영 및 서비스 제공으로 향상된 경험을 제공한다.

에지 컴퓨팅 패러다임을 실현하기 위하여 여러 오픈소스 플랫폼이 구현되고 있다. 분석기능을 클라우드에서 에지 장치로 이동하기 위해 Microsoft Azure는 Azure IoT Edge를 클라우드 서비스 프로바이더로 (Cloud Service Provider) 제공한다 [42]. 에지 장치는 컴퓨팅 리소스를 지원하는 라우터, 게이트웨이 또는 기타 노드가 될 수 있다. 애플리케이션을 클라우드에서 실행하는 사용자는 Azure IoT Edge를 통해 애플리케이션을 에지 디바이스로 이동하여 대기 시간을 줄일 수 있다. 에지 장치 응용 프로그램의 개발 복잡성을 단순화할 수 있고 Azure functions, Azure ML, Azure stream analytics를 활용해 머신러닝, 이미지 인식, AI 관련 서비스 등 에지 노드에 복잡한 작업을 설치할 수 있다. Apache Edgent [43]는 게이트웨이와 라우터 같은 작은 노드에서 실행할 수 있는 프로그래밍 모델이자 경량 런타임이다. 데이터 분석 구현을 가속화하기 위해 에지 네트워크에서 데이터 분석에 전념한다. CORD [44]은 AT&T에 의해 시작되어 네트워크 운영자들을 위해 제시된 ONF 프로젝트이다. 네트워크 장비 공급업체는 네트워크 인프라에 폐쇄적인 독점 통합 시스템을 제공한다. 네트워크 장비 공급업체에 의존하기 때문에 네트워크 기능을 네트워크 운영자와 함께 관리하기 어려워 컴퓨팅 및 네트워킹 리소스가 낭비된다. CORD는 소프트웨어 정의 네트워크 (Software Defined Networks, SDN), 네트워크 기능 가상화 (Network Function Virtualization, NFV) 및 클라우드 기술을 사용하여 에지 네트워크의 인프라를 재구성하여 데이터센터를 구축하려고 한다. 컴퓨팅, 스토리지, 네트워크 등 재구성된 데이터센터의 자원을 나눠 에지 네트워크에서 클라우드를 제공한다.

에지 컴퓨팅은 스토리지를 제공하고 클라우드의 데이터센터가 아닌 네트워크 에지에서 컴퓨팅 작업을 수행하기 때문에 보안 및 개인정보 보호 문제가 발생할 수

있다. [45] 저자들은 이러한 문제점을 해결하기 위해 여러 사물인터넷 네트워크에 대한 인증 서비스를 제공할 수 있는 보안 관리자를 자원이 제한된 디바이스로 구축하여 사물인터넷 네트워크에 대한 보안 인증 메커니즘을 제공하여 사물인터넷 네트워크에서 보안 데이터베이스 관리 비용을 절감하는 방법을 제안했다.

연구 [46]는 다중 마이크로컨트롤러를 사용하여 사물인터넷 기기에 업그레이드된 컴퓨팅 성능을 제공할 수 있는 에지 게이트웨이를 설계했다. 산업용 사물인터넷 시스템은 게이트웨이의 컴퓨팅 능력이 강화됨에 따라 클라우드와의 정보 전송 지연 시간이 단축되고 사물인터넷 기기와의 통신 반응 시간이 단축된다. 에지 게이트웨이는 전력 사용량을 더욱 단축하는 분산 컴퓨팅을 수행하기 위해 저비용 마이크로컨트롤러를 선택한다. Azure IoT Edge [47] CORD [48] 등 에지 컴퓨팅을 실현하는 프레임워크에 비해 EdgeX는 마이크로서비스 구조로 에지 컴퓨팅을 구현한다. EdgeX 에지 컴퓨팅 플랫폼은 에지 컴퓨팅 패러다임을 구현하는 벤더 (Vendor) 중립 오픈소스이다. EdgeX는 벤더 중립 플랫폼이기 때문에 라우터, 게이트웨이, 기타 노드를 포함한 컴퓨팅 자원을 제공하는 모든 장치에 설치할 수 있다. EdgeX는 여러 계층으로 기능을 나눠 구현한다. 특히 연결 계층에는 다양한 프로토콜로 구현된 마이크로 서비스가 있으며 프레임워크를 제공함으로써 사용자가 원하는 연결 프로토콜을 이용한 연결 서비스를 쉽게 구현할 수 있다. 개발 언어에서 Java뿐만 아니라 C와 Go lang 언어도 서비스를 구현하는 데 사용될 수 있다 [49]. 조사 플랫폼 비교 시 기기뿐만 아니라 개발 언어에 의존하지 않고 IoT 기기의 이기종 기기 요구 사항을 충족할 수 있는 개발 플랫폼으로 EdgeX가 선정됐다. 사물인터넷 디바이스의 제한된 자원과 다양성, 분산 컴퓨팅 등의 문제의 해결책으로 EdegX 플랫폼을 선택할 수 있다.

마이크로 서비스 기반 웹 애플리케이션 개발은 개발 및 배치에서 유연성, 경량성, 느슨한 결합의 특성을 통해 유지보수 및 기능 확장의 이점을 제공한다 [50]-[52]. 사물인터넷 네트워크에서 제한된 자원을 이용하는 애플리케이션을 개발하기 위해, 일련의 소규모 서비스는 독립형 프로세스에서 실행되는 마이크로 서비스를 통해 제공될 수 있다 [53]. EdgeX Foundry는 표준 사물인터넷 에지 컴퓨팅 프레임워크를 Raspberry Pi와 같은 자원이 제한된 장치를 이용하여 네트워크 에지에서 마이크로 서비스에 기반한 서비스를 구축할 것을 제안한다 [54]. EdgeX 프레임워크는 이기종 장치 프로토콜에 대한 연결을 지원하며 장치, 데이터 및 에지 컴퓨팅 환경을 위한

다양한 관리 기능을 제공한다. EdgeX를 사용하여 Edge Gateway에서 필요에 따라 여러 마이크로 서비스 서버 모듈을 추가하거나 삭제하여 기능을 확장 및 축소할 수 있다. 네트워크 에지에서 관리 기능을 제공하기 위해 애플리케이션, 네트워크, 시스템의 구성, 모니터링, 인터페이스와 같은 여러 기능이 필요하다 [55].

2. 디지털 트윈

디지털 트윈 기술은 물리적 세계를 데이터 변환하여 가상공간에 표현한다 [56]. 디지털 트윈 기술은 새로운 시스템에 대한 포괄적인 통찰력을 제공하여 설계 및 조사 프로세스를 변환한다 [57]. 디지털 트윈은 물리적 시스템을 디지털 공간으로 디지털 형태로 매핑하는 것이다. 실시간으로 물리적 시스템의 동작을 인식, 모니터링, 동기화, 시뮬레이션, 계산 및 테스트할 수 있는 통합 시스템이다. 디지털 트윈의 핵심 역할은 물리적 실체에 대해 매우 정확한 수치 표현을 제공하는 것이다 [64]. 디지털 트윈은 실시간 작업 상태를 모니터링하고 지능적인 의사 결정을 수행할 수 있도록 지원한다 [65]. 에지 컴퓨팅은 사물인터넷 디바이스의 물리적 인터페이스와 통신 프로토콜의 이질성을 수용하여 물리적 자원과 가상 모델의 실시간 상호작용을 촉진한다. 또한 에지 컴퓨팅의 빠른 속도와 짧은 지연 시간의 실시간 데이터 분석은 최적의 제어를 가능하게 한다.

논문 [66]는 에지, 포그 및 클라우드 컴퓨팅을 기반으로 하는 스마트 공장을 위한 참조 아키텍처를 제안한다. 스마트 공장 시스템은 여러 계층을 포함하고 있다. 최하위 계층은 데이터의 원천이며 에지 컴퓨팅을 제공하는 스마트 디바이스이다. 중간 계층은 포그 컴퓨팅이 실행하는 데이터를 전송하는 네트워크이다. 최상위 계층은 데이터를 저장, 분석하는 클라우드이다. 사이버와 물리 환경의 통합을 위한 효과적인 방법으로서 디지털 트윈은 제조 기업에 스마트 생산 및 정밀 관리를 수행할 수 있는 기능을 제공한다.

유지보수가 생산 라이프사이클 비용의 60~70%를 차지한다 [68]. 최적의 유지보수 정책은 운영 비용을 절감하고 장비 다운타임을 최소화하기 위해 매우 중요하다. 예

방적 유지보수는 기계 고장이 발생하기 전에 해결함으로써 생산 라이프사이클 비용을 최소화하기 위한 새로운 방법이다 [69]. 자동화 시스템의 이상 감지는 센서를 사용하여 장비를 지속적으로 모니터링하는 것과 탐지 모델을 이용하여 이상을 감지하는 두 단계를 포함한다. 이상 감지의 주요 과제 중 하나는 대량의 데이터 처리이다. 논문 [67]는 산업 시스템의 실시간 상태 모니터링 및 이상 예측을 가능하게 하는 새로운 디지털 트윈 기반 이상 감지 프레임워크를 제안한다. EI를 적용한 프레임워크는 동적 산업 에지/클라우드 네트워크에서 디지털 트윈 기술을 구현하여 고성능 이상 감지를 보장하는 실현 가능한 접근 방식을 제공한다. 에지 기반 디지털 트윈은 에지 장치에서 컴퓨팅 및 스토리지 기능을 제공하여 효율적인 데이터 처리를 가능하게 한다. 디지털 트윈 [70] 기술은 시스템을 모델링, 모니터링 및 분석을 위한 기술이다. 디지털 트윈은 물리 객체의 디지털 표현으로, 적절한 동기화 속도로 물리 객체와 디지털 표현 간의 융합을 가능하게 한다. 디지털 트윈은 물리 객체와 디지털 모델의 데이터를 저장한다. 저장된 데이터를 이용하여 모델을 훈련하고 실시간 데이터를 통하여 시스템의 이상을 감지할 수 있다. 에지 컴퓨팅을 적용하면 클라우드 기반 디지털 트윈에 비해 데이터 전송 효율을 높일 수 있고 실시간 성능이 필요한 애플리케이션을 만족할 수 있다.

논문 [71]는 사람과 디지털 트윈 기술을 통합하여 사람을 자동으로 모니터링하고 지원할 수 있는 생태계를 제시한다. 논문 [72]는 퍼지 규칙을 이용하여 환자의 현재 상태를 확인하는 홈 기반 모니터링 CPS (Cyber Physical System)을 제안한다. 환자의 데이터는 환자의 모바일 기기에 일시적으로 저장되고, 홈 기반의 웹서비스에 지속적으로 저장된다. 심근경색(Myocardial Infarction, MI)은 심장 근육으로 가는 산소가 혈액 흐름의 중단으로 공급이 잘 안 되어 심장이 손상을 입는 IHD (Ischemic Heart Disease)의 일종이다. 전문가에 의한 지속적인 모니터링은 이러한 시나리오를 막기 위한 좋은 전략이지만, 위험에 처한 모든 사람들을 감시하는 것은 불가능하다. IHD 또는 뇌졸중 등 응급 상황에 대처하기 위해서는 시간이 매우 중요하다. 논문 [73]은 에지에서 실행되도록 설계된 IHD 감지를 위한 Cardio Twin 아키텍처를 제시한다. 사람 몸에 장착하는 센서, 의료 기록, 소셜 네트워크 및 외부 센서로부터 데이터를 수집하고, 스마트폰을 이용하여 수집된 데이터를 처리하고 IHD를 예측한다.

논문 [58]은 디지털 트윈의 응용 중 하나로 교통 상태를 확인하는 차량 지원 주

행이다. 그림 2.3은 에지에서의 서비스 비용 최소화를 위한 작업 오프로드 전략의 최적화를 위한 디지털 트윈 및 다중 에이전트 학습 프레임워크에 기반한 양방향 협력 메커니즘의 작업 개요를 보여준다.

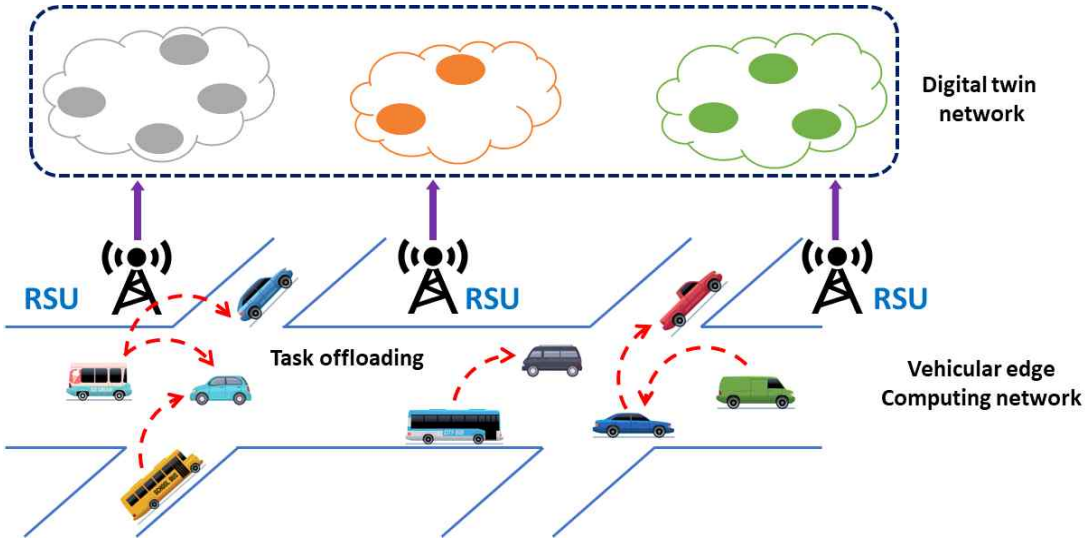


그림 2.3 차량들로 구성하는 에지 컴퓨팅 네트워크를 위한 디지털 트윈의 응용

3. 기계학습

기계학습은 수학, 통계, 인공지능 및 컴퓨터를 병합하여 입력 데이터에서 지식을 추출하여 자동으로 의사결정을 내리는 기술이다. 일반적인 학습방법은 지도 학습, 비지도 학습 그리고 강화 학습으로 분류한다 [84]. 지도학습은 입력 데이터와 예측할 데이터의 암묵적 관계를 도출하는 기법이다. 훈련데이터는 입력 데이터로 특징 값과 출력 데이터로 예측 값을 포함한다. 입력 값과 출력 값 사이의 기본 관계를 일반화하는 함수 모델을 생성한다 [85]. 비지도 학습은 레이블링 되지 않은 데이터 세트에서 숨겨진 구조를 탐지하도록 계획되고 있다. 학습의 일반적인 방법은 확률론적 데이터 모델을 통한 훈련을 포함한다. 결과적으로, 미리 지정된 종속 속성 없이 사례를 그룹화하는 기술이다. 이 기술은 일반적으로 순수 비정형 노이즈를 제거함으로써 데이터의 정형화된 패턴을 학습하는 것을 포함한다 [86]. 강화 학습은 일련의 행동에 대한 보상과 처벌을 수반하는 통제 이론적인 시행착오 학습 전략이다. 이 기술에서

에이전트는 환경과 상호 작용하면서 학습한다. 그리고 환경에서의 피드백을 근거로 결정을 한다. 이 기법은 일련의 행동에 대한 보상과 처벌이 있는 제어 이론적인 시행착오 학습 전략이다 [87].

기존 스마트 홈 시스템의 지능 수준은 세 개의 계층으로 나눌 수 있다 [88]. 첫째, 스마트 홈에서의 원격 운영은 자동화나 지능형 제어와는 거리가 먼 낮은 수준의 지능이다. 사용자는 모바일 애플리케이션을 통해 제어 메시지를 전송하여 구동체를 제어한다. 이 방법은 일반적인 원격 작동 방법이며 사용자가 수동으로 제어하지 않으면 수행할 수 없다. 둘째, 중간 수준의 지능형 스마트 홈은 지능형 환경 인식 장치에 의존하여 환경 정보를 수집한 후 미리 정의된 규칙 집합에 따라 적절한 대응을 한다. 이 지능형 컨트롤러는 환경 내의 조건이 변경되었을 때 자가 적응 제어를 실현할 수 있다. 셋째, 스마트 홈의 높은 지능은 개인화된 서비스와 친절한 사용자 경험을 제공한다. 이 솔루션은 환경 변화에 적응할 뿐만 아니라 사용자 행동도 스스로 적응한다. 따라서 학습 능력을 갖추고 사용자 행동을 예측할 수 있어야 한다 [89].

빅데이터와 기계학습 기술의 발전으로 스마트 홈 기기 관련 작동데이터를 이용하여 사용자의 행동 습관을 예측하는 모델을 훈련한다. 논문 [90]은 인공 신경망을 이용하여 비지도 학습방법으로 사용자의 행동을 예측하는 알고리즘을 제안한다. 스마트 홈의 가스 누출 및 화재는 인명 및 재산 손실을 초래하는 심각한 문제이다. 논문 [91]은 기계학습 전략을 이용하여 수집된 데이터에서 비정상적인 공기 상태 변화를 감지하여 위험 발생을 조기에 예측한다. 에너지 절도는 스마트 그리드 커뮤니티에서 심각한 문제이다. 에너지 절도 방법은 스마트 가전제품 해킹과 다른 가정의 전기 공급 장치에 대한 직접적인 연결이 포함된다 [92]. 관련된 다른 방법으로는 스마트 미터의 소프트웨어, 메커니즘 변경 및 클라우드에 저장된 데이터 조작이 있다. 따라서 공격자는 변조 및 해킹을 통해 다른 가구를 조작하여 커뮤니티 내 모든 고객의 총 전기요금을 그대로 유지함으로써 자신의 전기사용량을 줄일 수 있다 [93]. 논문 [94]은 기계 학습 및 통계 모델을 기반으로 스마트 홈의 에너지 절도를 방지하는 에너지 감지 시스템을 개발한다. 의사결정 모듈에는 3가지 단계가 있는데, 첫 번째 단계는 멀티모델 예측 시스템을 사용하는 예측 모델이다. 이 시스템은 전력 소비를 예측하기 위해 다양한 기계 학습 모델을 단일 예측 시스템에 통합한다. 2단계는

비정상 현상을 필터링하기 위해 단순이동평균을 사용하는 1차 의사결정 모델이다. 세 번째 단계는 에너지 절도에 대한 결정의 최종 단계인 2차 의사결정 모델이다.

4. 에지 지능 (Edge Intelligence, EI)

인공지능 기반 애플리케이션을 에지에서 실행해야 하는 요건을 만족하기 위하여 EI는 최근 엄청난 관심을 받고 있다 [62]. 아마존, 마이크로소프트, 구글과 같은 회사들은 이미 지능을 에지에 이동하는 서비스 플랫폼을 출시했다. 에지에서의 분산 DNN 훈련 아키텍처는 중앙 집중형, 분산, 하이브리드의 세 가지 모드로 나눌 수 있다. 그림 2.4는 (a), (b) 및 (c)에 각각 설명된 세 가지 아키텍처를 보여준다. 클라우드는 중앙 데이터 센터를 가리키는 반면, 엔드 디바이스는 데이터 소스이기도 한 휴대 전화, 자동차 및 감시 카메라로 표현된다. 에지 서버의 경우 기지국을 가리킨다.

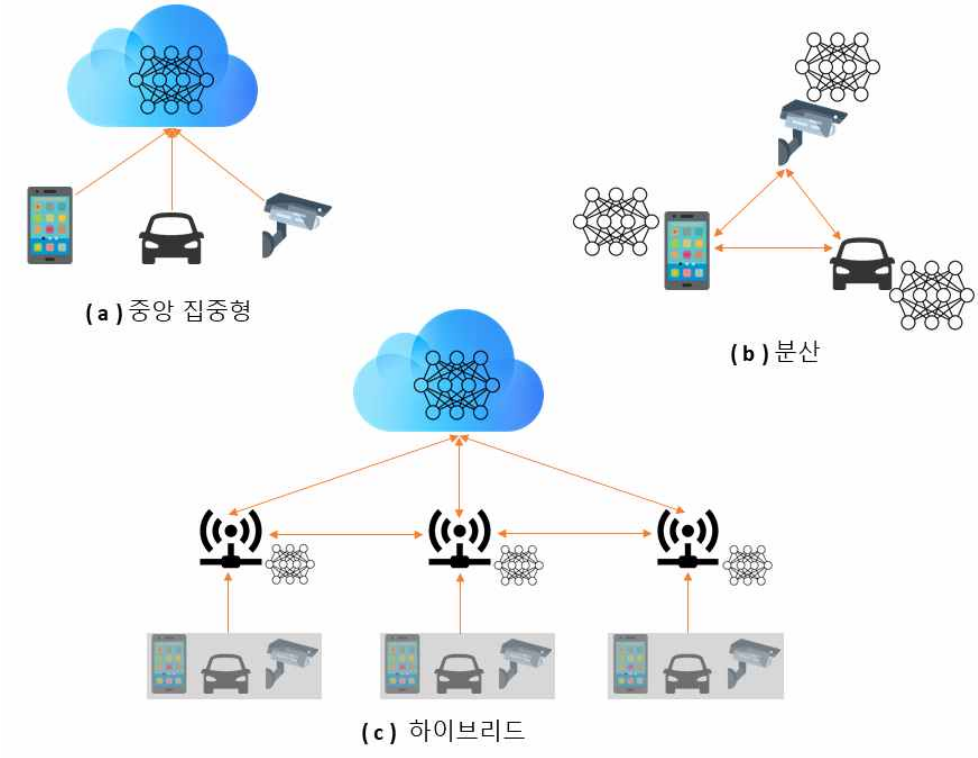


그림 2.4 에지 컴퓨팅 네트워크에서 분산 DNN 훈련 아키텍처

(a)는 DNN 모델이 클라우드 데이터 센터에서 교육되는 중앙 집중식 DNN 훈련을 설명한다. 훈련을 위한 데이터는 휴대폰, 자동차, 감시 카메라와 같은 분산된 최종 장치에서 생성되고 수집된다. 데이터가 도착하면 클라우드 데이터센터에서 해당 데이터를 사용하여 DNN 학습을 한다.

(b)에 표시된 것처럼 분산 모드에서 각 컴퓨팅 노드는 로컬 데이터로 자체 DNN 모델을 로컬에서 학습한다. 개인정보는 로컬 노드에 보존한다. 로컬에서 학습한 모델을 공유하여 글로벌 DNN 모델을 얻기 위해 네트워크의 노드가 서로 통신하여 로컬 모델 업데이트를 교환한다. 이 모드에서는 클라우드 데이터센터의 개입 없이 글로벌 DNN 모델을 학습할 수 있다.

(c) 하이브리드 모드는 중앙 집중식 모드와 분산 모드를 결합한다. 에지 서버는 아키텍처의 허브로서 DNN 모델을 서로 분산 업데이트하거나 클라우드 데이터센터와 중앙 집중화된 훈련을 통해 모델을 학습할 수 있다.

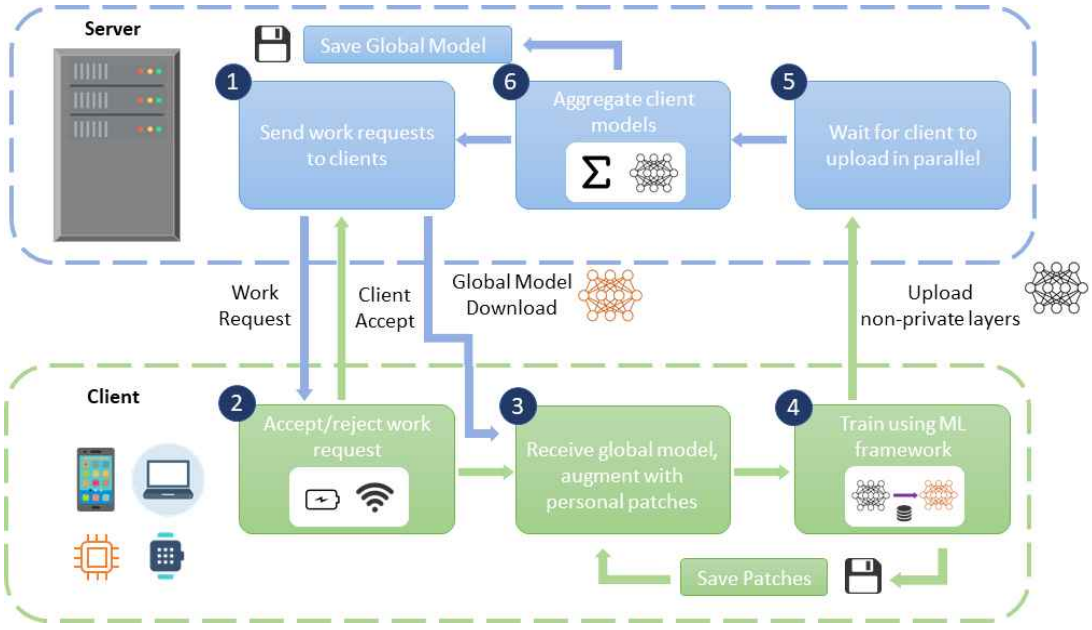


그림 2.5 다중 작업 연합학습 전략

그림 2.5는 개별 모델의 정확도를 향상시키기 위해 비연합 배치 정규화 계층을 사용하여 작동하는 다중 작업 연합 학습 알고리즘을 제시한다. 클라이언트와 서버로 구성되고 있다. 클라이언트는 에지 네트워크 단말에 위치한 핸드폰, 노트북 등 모바일 기기를 예로 들 수 있다. 서버에서 클라이언트들에게 작업 요청을 보내고 클라이

언트는 작업 요청을 거절하거나 수락한다. 작업을 수락하면 서버로부터 글로벌 모델을 다운로드 하고 로컬 데이터와 기계학습 프레임워크를 이용하여 모델을 훈련한다. 훈련된 모델은 서버로 전달한다. 서버는 모든 클라이언트가 전달한 모든 모델을 수집하고 글로벌 모델을 업데이트 하고 저장한다.

에지에서 에너지 소비를 예측하기 위해 [59]에서 연합 학습 방식이 사용된다. 제안된 연구는 LSTM(Long Short-Tterm Memory unit) 알고리즘을 사용했다. 이 모델은 구체적인 사용자의 에너지 소비데이터를 기반으로 로컬에서 훈련함으로써 사용자의 개인정보를 보호한다. 아래 그림 2.6은 제안된 시스템의 아키텍처 개요를 보여준다. 모델을 훈련할 때 원시데이터를 분산된 클라이언트들이 유지함으로써 프라이버시를 보장할 수 있다 [60].

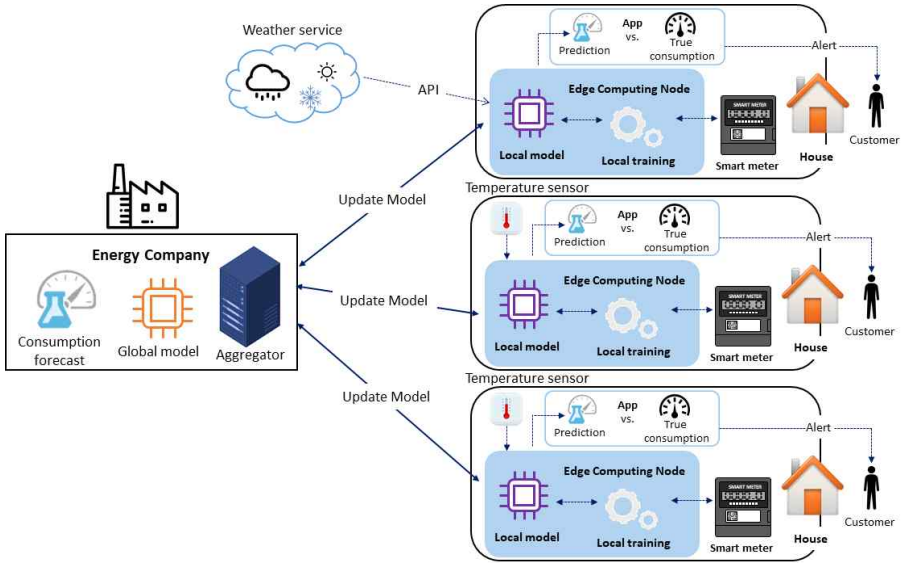


그림 2.6 LSTM 기반의 단기 에너지 소비 예측을 위한 연합학습 전략

연합학습에서 로컬 모델을 서버에서 집계하여 업데이트한다. 그러나 서버에서 모델을 업데이트하는 방식이 훈련된 모델의 가중치 값을 평균하는 방법이다. 그러므로 모든 데이터의 지식을 학습할 수 없다. 또한 중앙 집중식의 처리 방식은 단일 장애 지점의 문제가 있다.

스웜학습은 단지 모델을 에지 네트워크의 노드들을 이용하여 학습한다. 데이터의

개인정보를 보호하고, 기밀성을 유지한다. 분산 스웜학습 모델은 대규모 데이터 세트의 관리, 저장 및 분석에 유용하다. [61]는 질병을 분류하기 위해 분산 스웜학습 접근법을 사용했다. 그림 2.7은 에지 장치에서의 학습 전략을 표현한다. 제안된 스웜학습 모델의 실험 결과는 혁신적인 정밀 의학을 기반으로 학습한 모델의 효과를 입증한다. 글로벌적인 협업, 확실한 데이터 보안으로 지식 공유와 같은 스웜학습의 고유한 특성은 스웜학습을 스마트 공간에 적용하기에 최적의 프레임워크로 만든다.

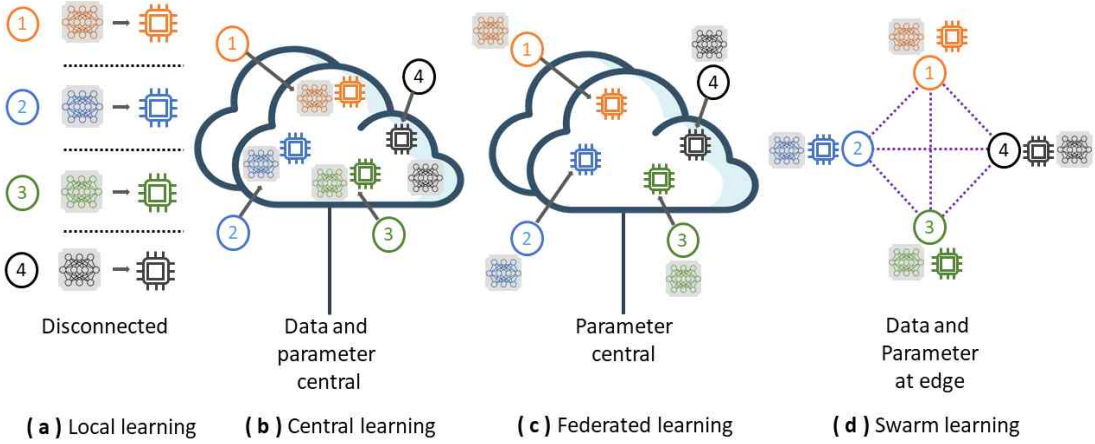


그림 2.7 분산 에지 지능 환경에서 예측 학습 모델링 전략

5. 스마트 홈을 위한 최적화 알고리즘

최적화는 문제를 해결할 수 있는 모든 가능한 해결책으로부터 정해진 목적에 맞는 최적의 결과를 찾는 방법이다 [74]. 고차원 문제를 해결하는 데 있어 휴리스틱(heuristic) 혹은 메타 휴리스틱 (metaheuristic) 알고리즘을 적용한다 [75].

스마트 홈 가전제품의 전력 소비를 최소화하기 위하여 메타 휴리스틱 알고리즘을 이용하여 스마트 홈 에너지 사용을 스케줄링한다. 전력 스케줄링 문제(Power Scheduling Problem, PSP)는 전기 요금 체계에 따라 스마트 가전제품의 실행을 적절하게 스케줄링하는 문제이다. 스마트 가전제품의 스케줄링은 실행시간은 한 기간에서 다른 기간으로 시간 운영을 옮겨서 한다. 스케줄링 프로세스의 목표는 전기 요금을 줄이고 사용자 편의 수준을 향상시키는 것이다 [76]. 논문 [77]은 가전제품을 효

율적으로 제어할 수 있도록 홈 에너지 관리 시스템을 제안한다. 시스템의 목적은 전력 수요가 최대치에 이르는 시간대에 전기 요금을 줄이는 것이다. 가전제품에 대한 적절한 스케줄링을 위해 PSO 및 GA 메타 휴리스틱 최적화 알고리즘을 적용한다. 논문 [78]는 BPSO 최적화 알고리즘을 사용하여 스마트 홈에서 작업의 균형을 맞춘다. 전력은 전통 에너지와 재생 에너지를 고려한다. 최적화 문제의 목적은 전기 요금을 최소화하고 재생 에너지의 전력 소비를 극대화하는 것이다. 논문 [79]는 미리 정의된 시간 범위 동안 전력 수요의 균형을 맞추는 두 가지 동적 전기 요금 체계 조합을 제안한다. 전기 요금과 사용자의 불편을 최소화하는 다목적 접근법을 사용했다. 90일 동안 하루를 120 등분하여 가전제품을 6번 실행하는 PSP문제를 해결하기 위하여 유전자 알고리즘 (Genetic Algorithm, GA)을 사용한다. 논문 [8]에서 PSP를 해결하기 위하여 BPSO (Binary Particle Swarm Optimization)와 GA 알고리즘을 적용한다. 동적 가격 체계를 적용하여 10개 가전제품의 하루 실행 스케줄링을 한 결과 BPSO 알고리즘이 더 좋은 성능을 보여주는 것을 알 수 있었다. 전력 스케줄링 문제 해결의 목표 중 하나로 거주자의 편의 수준을 고려하여, 가전제품 실행의 지연 시간을 사용자 편의 수준을 평가하는 데 사용한다 [81].

PSO 알고리즘은 새와 물고기의 먹이를 찾는 행동에서 파생되고 있다. PSO는 입자라고 불리는 개인 간의 정보 교환에 의존한다. PSO에서, 각 입자는 이전에 최고의 성능과 이웃 또는 전체 군집의 최고의 이전 성능의 위치로 궤적을 확률적으로 조정한다 [82]. 에지 컴퓨팅은 사물인터넷의 에지에서 사물인터넷 디바이스에 컴퓨팅, 스토리지 자원을 지원한다. 하지만 컴퓨팅 자원이 클라우드나 전문 컴퓨터에 비해 제한되어 있는 것은 사실이다. PSO 알고리즘은 간단하고 구현하기 쉬운 최적화 알고리즘이다 [83]. 구현하기 쉽고 간단하여 다른 방법과 빠르게 통합이 되고, 계산 비용이 낮으므로 에지 컴퓨팅 환경에서 응용프로그램으로 실행하여 서비스를 제공하기에 적합하다. 위의 논문들은 스케줄을 통하여 가전제품을 제어하기 때문에 능동적으로 스마트 홈 환경의 변화에 반응할 수 없고, 거주자의 편의 수준을 고려한다고 하였지만 열 쾌적성에 대해 고려하지 않고 있다. 본고에서는 PSO 알고리즘 적용을 통해 거주자의 열 쾌적성을 고려하여 스마트 홈 환경을 최적 온도와 습도의 상태로 유지하는 방법을 제안한다.

III. 분산 디지털 트윈 에지 컴퓨팅 환경 구축

1. 분산 디지털 트윈을 위한 에지 컴퓨팅 가상화

사물인터넷이 지속적으로 증가하고 있다. 지역적으로 분산되어 있는 사물인터넷 장비들을 이용하기 위하여 에지 컴퓨팅 기술을 사용하고 있다. 이러한 환경에서 분산된 에지 컴퓨팅 환경을 효율적이고 용이하게 사용하고 상태를 실시간으로 모니터링 하기 위하여 디지털 에지 컴퓨팅 환경을 구성한다.

분산 디지털 트윈 에지 컴퓨팅 환경은 물리적인 분산 에지 컴퓨팅 환경의 데이터 기반 표현을 인터넷을 통하여 사이버 세계로 매핑하여 사용자로 하여금 전반적인 시스템의 통찰력을 가질 수 있도록 한다. 개개의 기기와 연결하여 관리하는 방법은 사물인터넷의 폭발적인 발전으로 인하여 시간이나 효율 방면에서 많이 부족하다. 하나의 통합된 환경에서 시스템 전체의 상태를 확인하고 관리할 수 있는 패러다임에 대한 요구를 충족하기 위하여 디지털 트윈 기술을 분산 에지 컴퓨팅 환경에 도입한다. 분산 디지털 트윈 에지 컴퓨팅은 그림 3.1과 같이 물리적인 에지 컴퓨팅 네트워크 환경을 가상화하여 사이버 환경에서 가상의 에지 컴퓨팅 네트워크 환경을 구성한다.

물리적인 에지 컴퓨팅 네트워크 환경은 여러 에지 컴퓨팅 서비스를 실행하는 에지 게이트웨이 (Edge Gateway)들과 게이트웨이에 연결된 여러 사물인터넷 디바이스로 (IoT Device) 구성한다. 에지 게이트웨이 그리고 사물인터넷 디바이스는 각각 라즈베리 파이 4와 라즈베리 파이 3에 우분투 운영체제를 설치하여 구축한다. 모든 기기는 와이파이를 통하여 라우터와 연결하였고 고유의 네트워크 주소를 가진다.

가상 에지 컴퓨팅 네트워크 환경은 물리적 엔티티를 데이터로 가상화하여 표현한다. 에지 게이트웨이는 가상의 에지 게이트웨이, 사물인터넷 디바이스는 가상의 사물인터넷 디바이스와 1대1로 대칭한다. 물리적인 기기와 가상의 기기는 실시간으로 정보를 교환하여 상태변화를 웹 페이지를 통하여 반영한다.

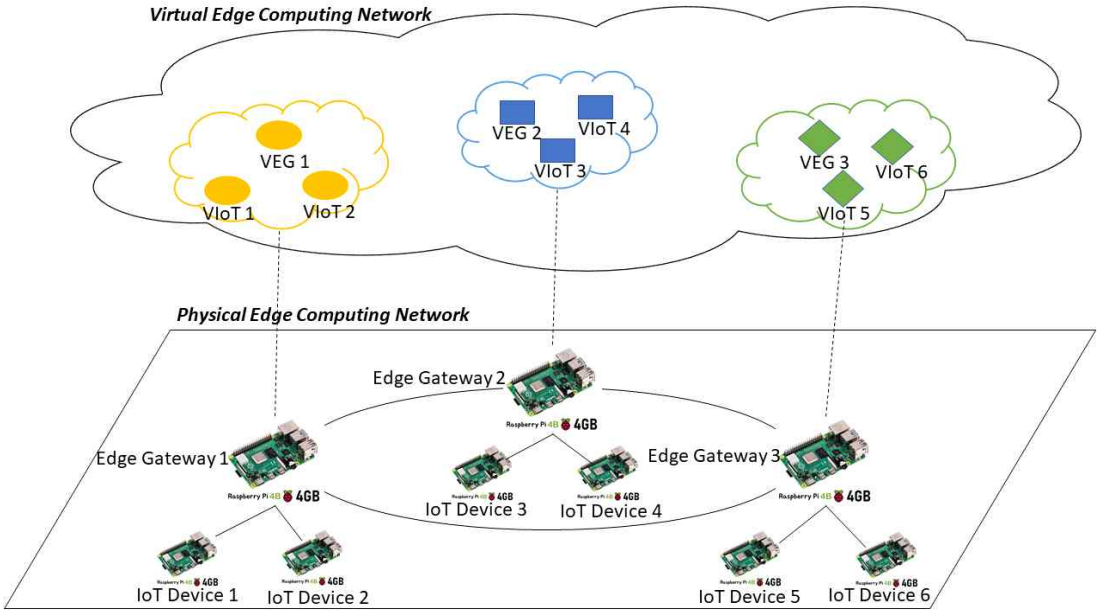


그림 3.1 사물인터넷 기반 분산 디지털 트윈 에지 컴퓨팅의 개념적 구조

물리적인 에지 컴퓨팅 네트워크 환경을 가상화하기 위한 서비스의 기능 구조는 그림 3.2와 같이 설계한다. 크게 3개의 부분으로 구성한다.

사물인터넷 인프라는 (IoT Infrastructure) 사물인터넷 디바이스를 사용하는 스마트 홈 (Smart Home), 스마트 공장 (Smart Factory), 스마트 빌딩 (Smart Building), 스마트 그리드 (Smart Grid) 등을 예로 들 수 있다.

에지 컴퓨팅 네트워크 (Edge Computing Network)는 다양한 에지 게이트웨이로 구성되고 있다. 에지 게이트웨이는 도커 (Docker) 기반의 에지 컴퓨팅 서비스들과 응용서비스들을 포함하고 있다. 도커 기술은 도커 엔진 (Docker Engine)에 의하여 이미지 기반의 서비스들의 볼륨, 네트워크 자원을 관리한다.

에지 서버의 에지 컴퓨팅 슈퍼바이저 (Edge Computing Supervisor, ECS)는 물리적인 에지 컴퓨팅 네트워크 자원을 가상화하는 디지털 트윈 오브젝트 생성자 (Digital Twin Object Generator), 상태를 모니터링하는 모니터링 기능, 에지 게이트웨이의 관리를 지원하는 에지 게이트웨이 관리 지원자 (Edge Gateway Management Supporter, EGMS), 작업 생성과 실행을 할 수 있도록 하는 작업 실행자 (Task Operator), 스웜 학습 (Swarm Learning)기능 그리고 데이터베이스 기능을 제공한다.

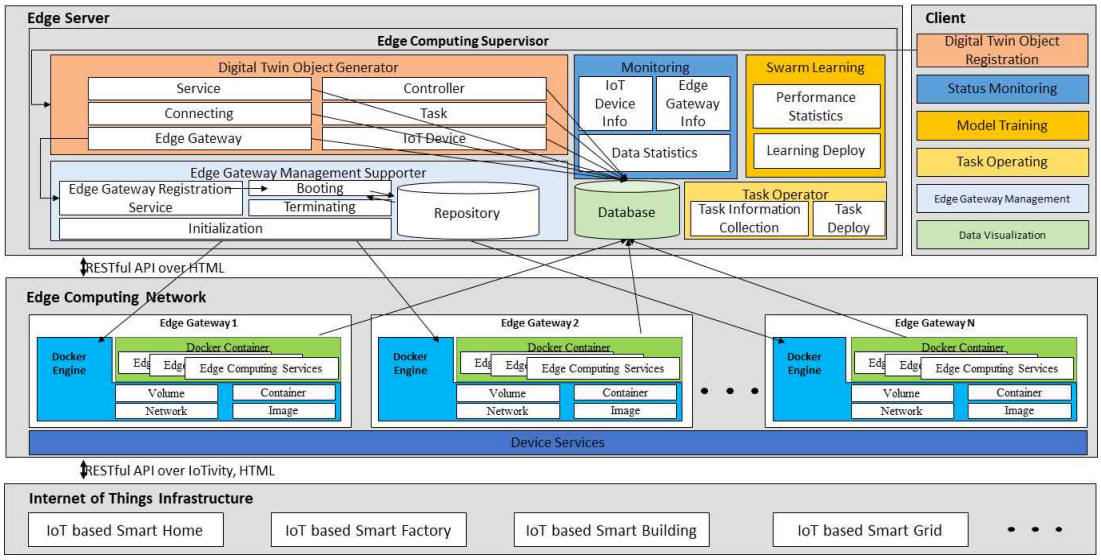


그림 3.2 분산 디지털 트윈 기반 에지 컴퓨팅 기능 구조

그림 3.3은 에지 게이트웨이 등록에 대한 기능 다이어그램이다. 사용자는 GUI (Graphc User Interface)를 통하여 에지 게이트웨이를 등록할 수 있다. 클라이언트, 에지 컴퓨팅 슈퍼바이저 그리고 에지 게이트웨이 등 컴포넌트를 포함하고 있다. 사용자는 클라이언트의 웹 브라우저를 이용하여 에지 컴퓨팅 슈퍼바이저에서 제공하는 웹 페이지를 방문한다. 에지 컴퓨팅 슈퍼바이저는 물리적인 에지 게이트웨이의 정보를 입력할 수 있는 웹 페이지를 제공한다. 사용자가 정보를 입력하고 등록 요청을 보내면 에지 컴퓨팅 슈퍼바이저의 디지털 트윈 오브젝트 생성자 블록은 사용자로부터 전달된 게이트웨이 등록 요청을 처리한다. 정보를 데이터베이스에 저장하기 전 먼저 등록하려는 이제 게이트웨이의 실행상태를 확인한다. 디지털 트윈 오브젝트 생성자는 검증 (validation) 기능을 통하여 에지 게이트웨이와 EGMS의 상태를 확인한다. 에지 게이트웨이는 에지 컴퓨팅 서비스들을 실행하는 플랫폼이다. 등록하려는 에지 게이트웨이에서 에지 컴퓥이 서비스가 실행되고 있지 않는 상태를 전제로 하고 있다. 그러므로 정상적으로 실행되고 있으면 EGMS를 통하여 등록하려는 게이트웨이에서 에지 컴퓨팅 서비스들을 실행하도록 한다. EGMS는 실행 (booting)기능을 이용하여 에지 게이트웨이 서비스들을 실행시킨다. 서비스들이 정상적으로 실행되면 마지막으로 사용자로부터 입력된 데이터를 데이터베이스에 저장한다. 반대로 에지 게이트웨이가 실행되고 있지 않으면 사용자에게 등록할 수 없는 알람을 전달하고

등록 과정을 마친다.

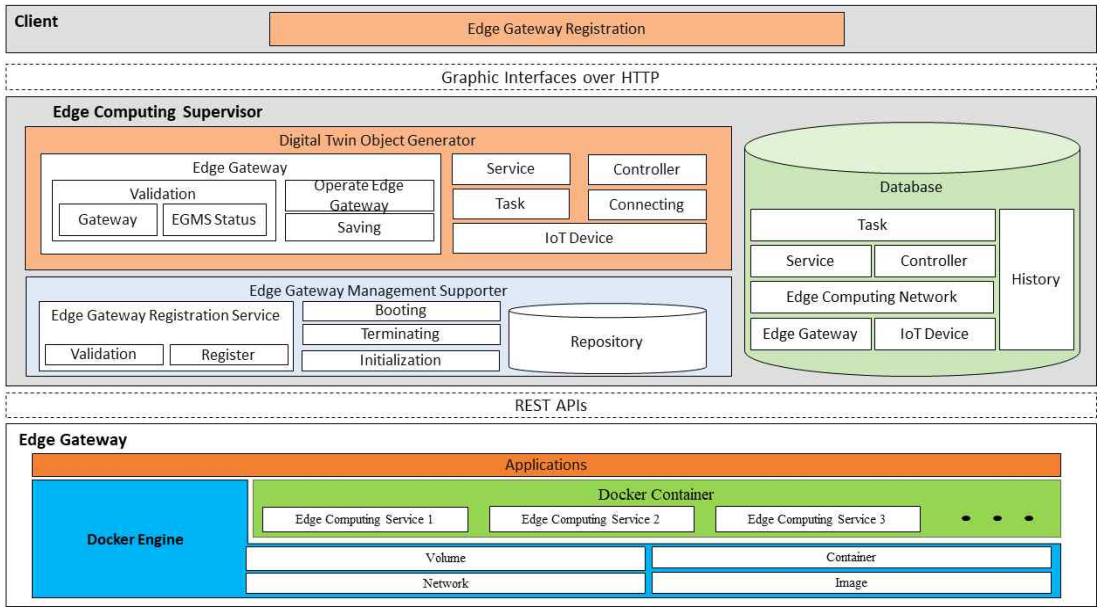


그림 3.3 분산 디지털 트윈 기반 에지 게이트웨이 등록 기능 구조

EGMS는 운영에 앞서 초기화 단계를 실행한다. EGMS는 에지 게이트웨이에서 실행하는 에지 컴퓨팅 서비스에 대한 제어 기능을 제공한다. 에지 컴퓨팅 서비스들의 정보는 파일로 EGMS에 전달한다. 그러므로 EGMS는 초기화 단계에서 에지 컴퓨팅 서비스를 제어하는 준비를 와성한다. EGMS 초기화 단계는 크게 에지 게이트웨이 정보 초기화와 에지 게이트웨이 실행 초기화 단계로 되어 있다.

그림 3.4는 에지 게이트웨이 정보 초기화 시퀀스 다이어그램이다. 에지 게이트웨이 정보 초기화는 먼저 데이터베이스로부터 등록된 에지 게이트웨이 정보를 질의한다. 등록된 정보가 없으면 초기화 단계를 마치고 실행단계로 진입한다. 등록된 게이트웨이들의 정보를 데이터베이스로부터 반환받으면 각각의 게이트웨이의 IP 주소정보에 근거하여 물리 에지 게이트웨이 디바이스로 구체적인 정보를 요청한다. 에지 게이트웨이가 실행하지 않거나 에러가 있으면 현재 루프를 중단하고 다음 게이트웨이에 대한 루프를 실행한다. 정보가 정상적으로 반환이 되면 저장소에 지정한 형식으로 정보를 저장한다.

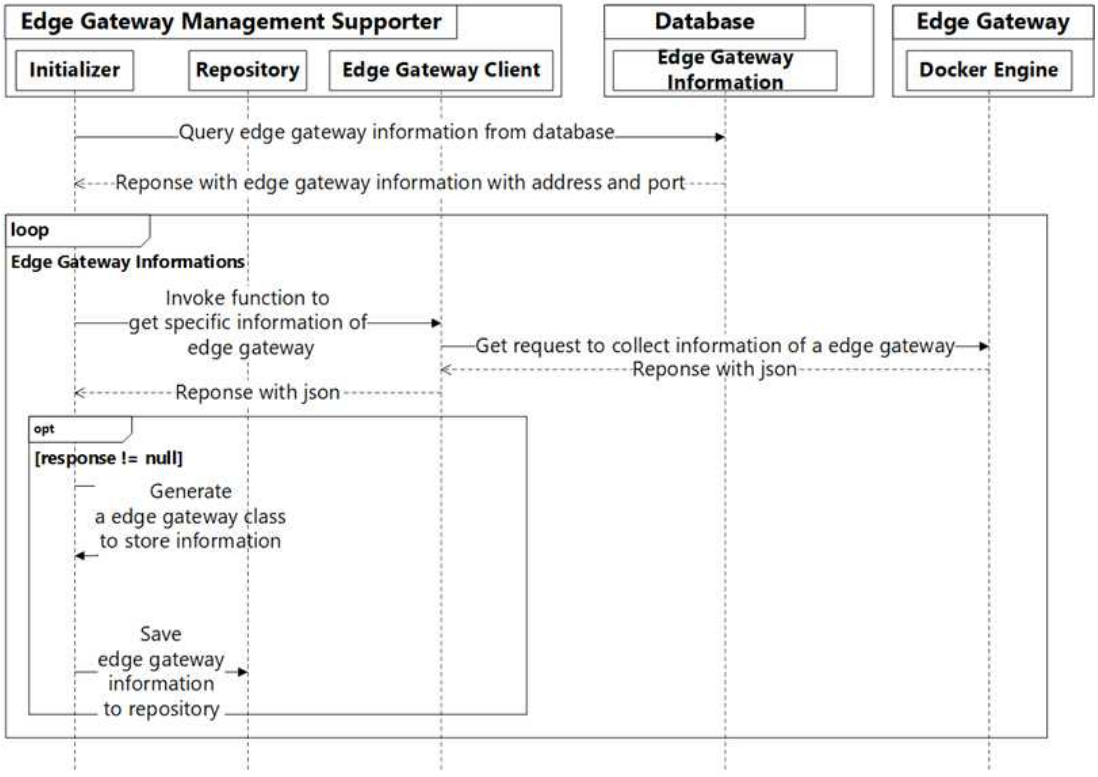


그림 3.4 에지 서버 (Edge Gateway Management Supporter)를 통한 에지 게이트웨이 초기화 시퀀스 다이어그램

에지 게이트웨이의 코어 서비스들은 도커 기반의 컨테이너화 (Containerization)된 마이크로 서비스들이다. 서비스들은 도커가 제공하는 네트워크, 볼륨, 그리고 다른 서비스에 대한 의존성을 가지고 있다. 서비스들의 실행을 간편화하기 위하여 설정 파일로 제공되는 의존성을 처리하여 실행환경을 조성해주는 과정이 에지 게이트웨이 실행 초기화이다. 에지 게이트웨이 실행 초기화 단계는 네트워크, 볼륨 그리고 다른 서비스에 대한 설정 파일을 파싱 (Parsing)하는 것에서부터 시작된다. 설정 파일은 네트워크, 볼륨 그리고 서비스들로 구분하여 설정 정보들이 구체적으로 작성되고 있다. 파싱된 정보는 저장소에서 요구하는 포맷으로 변환하고 저장된다. 파싱된 정보를 에지 게이트웨이에 설정하기 위하여 저장소로부터 에지 게이트웨이 정보를 호출한다. 에지 게이트웨이들에 각각 설정을 처리하는 과정이 루프로 진행된다. 에지 게이트웨이 클라이언트 (Edge Gateway Client)를 통하여 에지 게이트웨이의 도커 엔진과 연결을 한다. 그리고 네트워크, 볼륨, 서비스 컨테이너 (Container)를 생성하

는 명령어를 전달하여 에지 게이트웨이 실행 초기화 단계를 완성한다. 초기화 단계는 EGMS가 이상 현상으로 중단이 되었거나 시스템이 다운되어 다시 시작하였을 때 시스템의 정상적인 운영을 보장하기 위하여 진행한다.

그림 3.5는 에지 게이트웨이 네트워크 설정 초기화 시퀀스 다이어그램이다. EGMS는 네트워크 설정 정보를 파싱하여 저장한다. 그리고 저장소로부터 에지 컴퓨팅 네트워크에 있는 게이트웨이 정보를 조회한다. 조회된 게이트웨이 정보와 에지 게이트웨이 클라이언트를 통하여 에지 게이트웨이에서 네트워크를 구축한다. 구축하기에 앞서 먼저 구축하려는 네트워크가 존재하는지를 조회한다. 해당 네트워크가 존재하지 않으면 이를 생성하고, 있는 경우 실행을 종료한다.

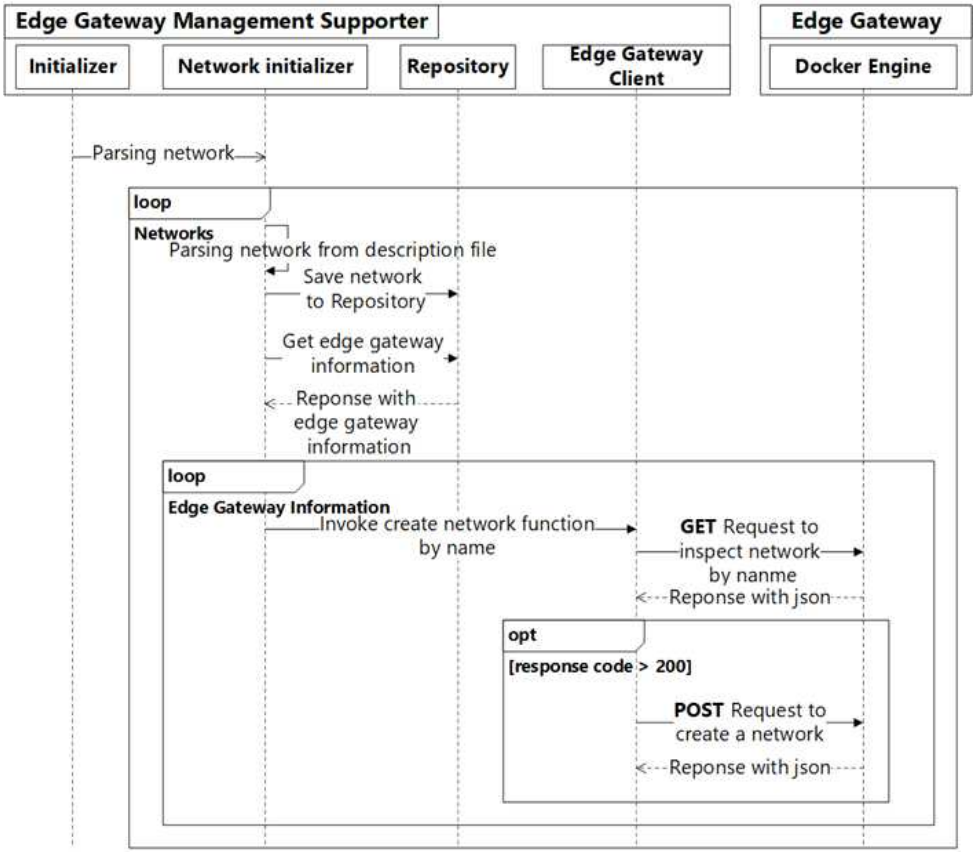


그림 3.5 에지 서버 (Edge Gateway Management Supporter)를 통한 에지 게이트웨이 네트워크 초기화 시퀀스 다이어그램

그림 3.6은 에지 게이트웨이 볼륨 초기화 시퀀스 다이어그램이다. EGMS는 볼륨 관련 설정 정보를 파싱하여 저장한다. 그리고 저장소로부터 에지 컴퓨팅 네트워크에

있는 게이트웨이 정보를 조회한다. 조회된 게이트웨이 정보와 에지 게이트웨이 클라이언트를 통하여 에지 게이트웨이에서 볼륨을 생성한다. 볼륨이 이미 존재하는지를 확인하기 위하여 볼륨의 존재 여부를 조회한다. 존재하지 않으면 생성하고, 있으면 실행을 종료한다.

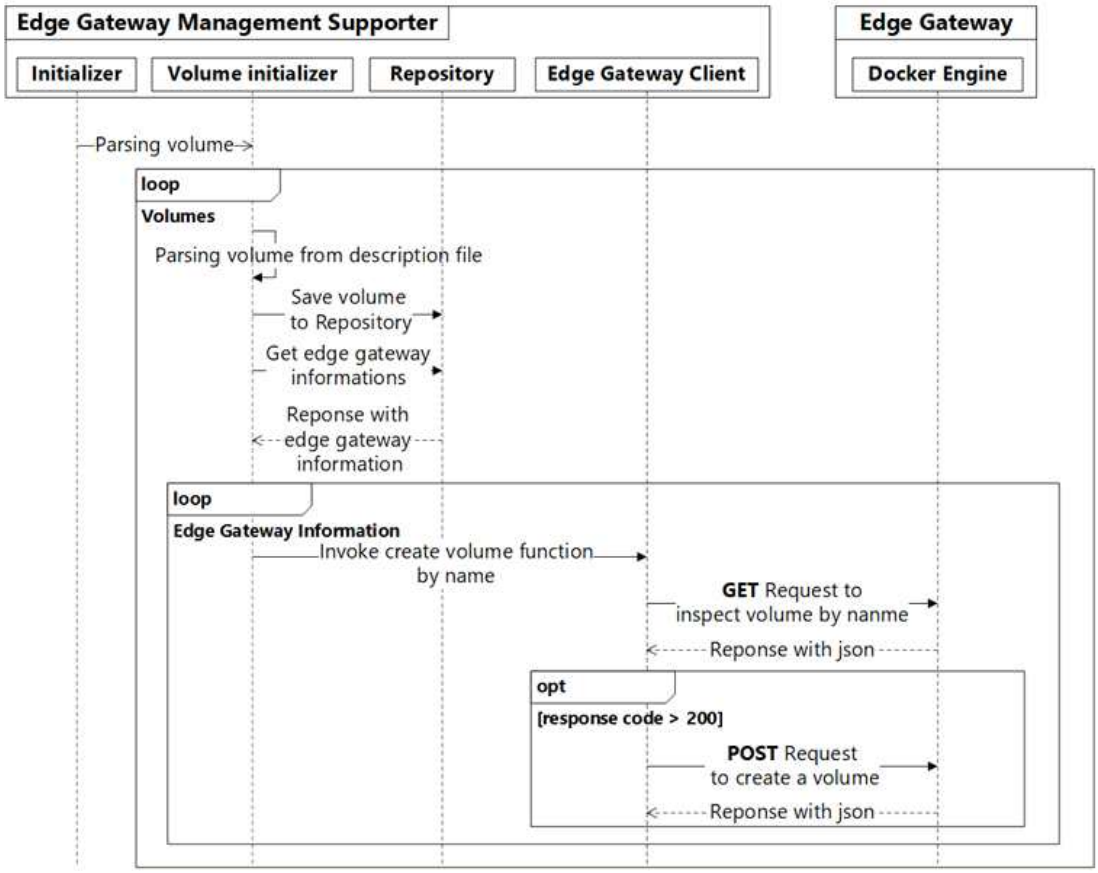


그림 3.6 에지 서버 (Edge Gateway Management Supporter)를 통한 에지 게이트웨이 볼륨 초기화 시퀀스 다이어그램

그림 3.7은 에지 게이트웨이 볼륨 설정 초기화 시퀀스 다이어그램이다. EGMS는 EdgeX 마이크로 서비스 관련 설정 정보를 파싱하여 저장한다. 그리고 저장소로부터 에지 컴퓨팅 네트워크에 있는 게이트웨이 정보를 조회한다. 조회된 게이트웨이 정보와 에지 게이트웨이 클라이언트를 통하여 에지 게이트웨이에서 서비스들의 컨테이너를 생성한다. 컨테이너를 생성하기에 앞서 먼저 서비스가 존재하는지를 조회한다. 존재하지 않으면 생성하고, 있으면 실행을 종료한 후 다음 서비스를 생성한다. 모든

서비스의 컨테이너가 생성되면 초기화 단계를 종료한다.

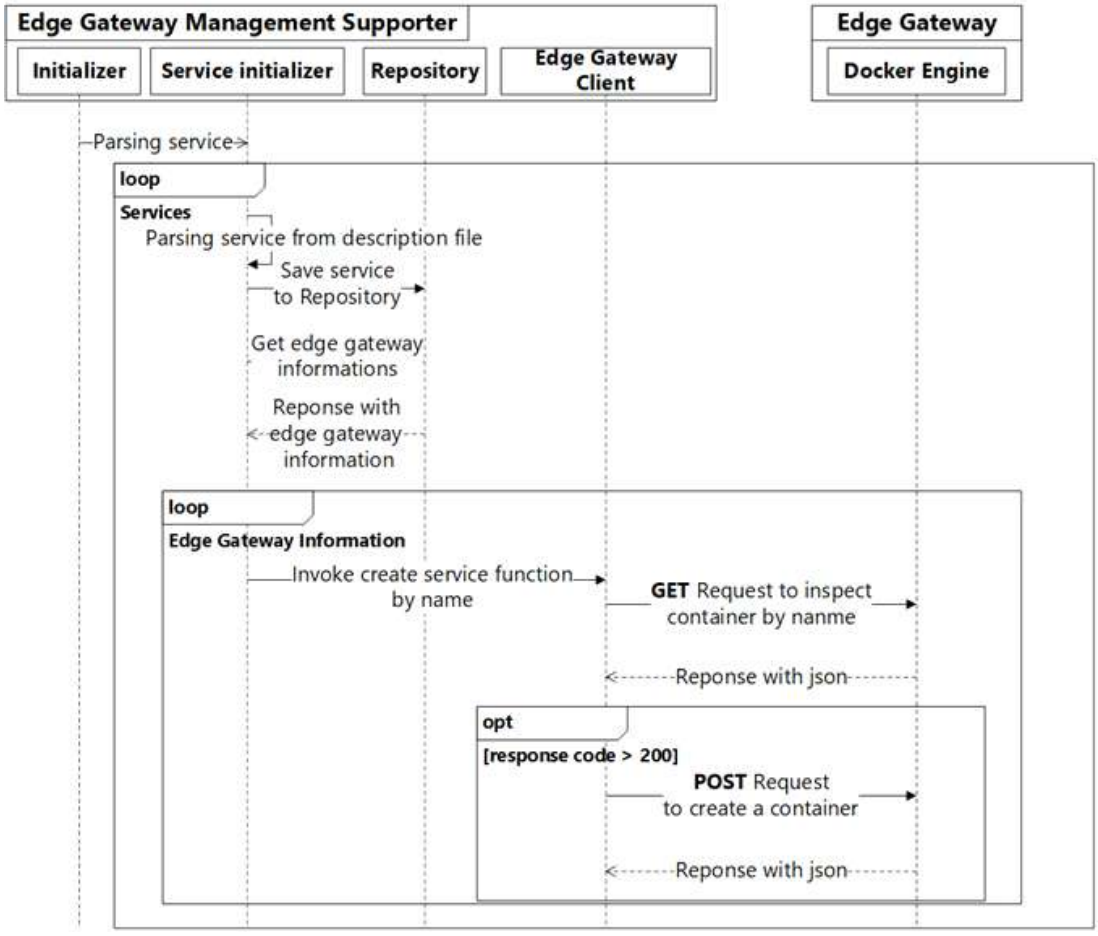


그림 3.7 에지 서버 (Edge Gateway Management Supporter)를 통한 에지 게이트웨이 서비스 초기화 시퀀스 다이어그램

에지 게이트웨이를 디지털 트윈 플랫폼 (Digital Twin Platform)에 등록하는 과정을 그림 3.8에서 시퀀스 다이어그램으로 설명한다. 사용자는 브라우저를 통하여 에지 게이트웨이 등록 요청을 디지털 트윈 오브젝트 생성자로 보낸다. 후자는 먼저 EGMS와 에지 게이트웨이가 정상적으로 실행하고 있는지를 검증 기능을 통하여 확인한다. 검증이 통과되면 EGMS를 통하여 에지 게이트웨이에서 에지 컴퓨팅 서비스들을 실행하도록 한다. EGMS는 사용자가 입력한 정보에 근거하여 에지 게이트웨이와 연결을 하여 상세 정보를 요청한다. 요청된 정보는 저장소에 지정한 형식으로 정보를 저장하고 저장소에서 초기화 단계에서 읽은 서비스 정보를 읽는다. 서비스들은

의존성이 없는 서비스들로부터 차례로 실행된다. 실행되기에 앞서 먼저 에지 게이트웨이의 도커 엔지에 서비스 상태를 이름을 통하여 조회한다. 서비스가 실행되어 있지 않았을 때 실행하도록 명령을 전달한다. 정상적으로 서비스가 에지 게이트웨이에서 실행된다면 디지털 트윈 오브젝트 생성자가 에지 게이트웨이 정보를 데이터베이스로 저장하고 결과를 사용자에게 전달한다.

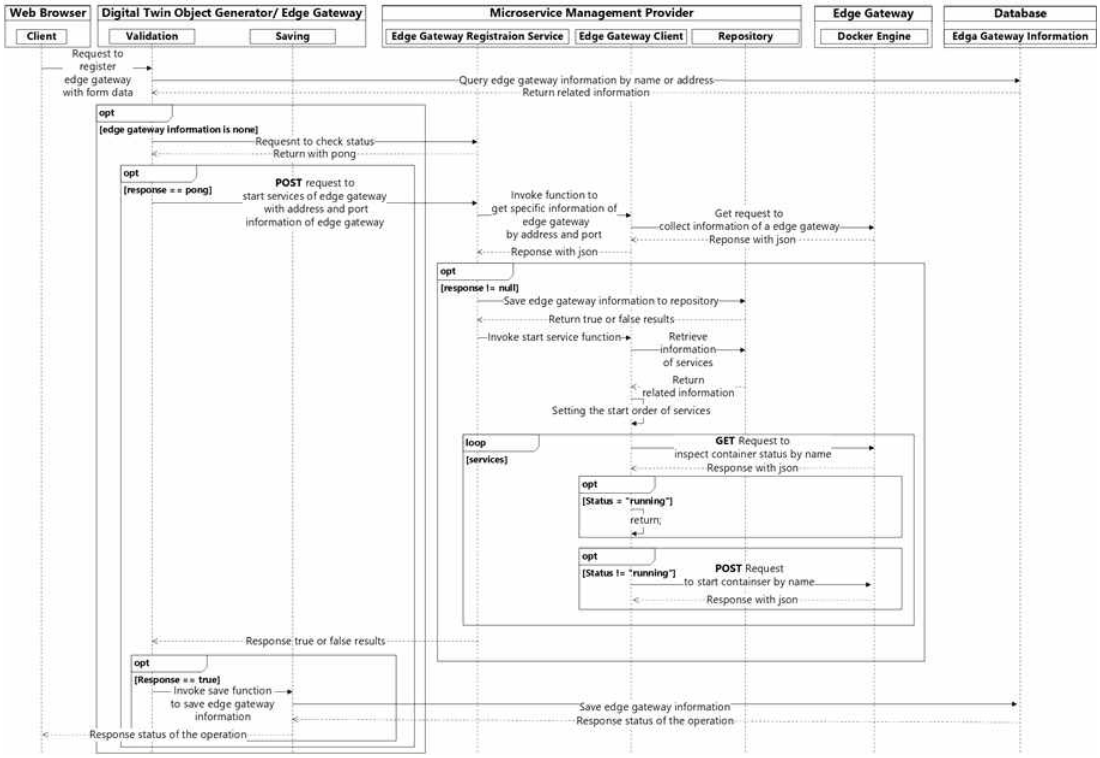


그림 3.8 에지 서버를 통한 에지 게이트웨이 등록 시퀀스 다이어그램

그림 3.9는 서비스, 컨트롤러 그리고 작업을 디지털 트윈 플랫폼에 등록하는 기능 구조이다. 디지털 트윈 플랫폼은 사물인터넷 디바이스와 에지 게이트웨이뿐만 아니라 수집된 데이터에 근거하여 예측 및 최적화 기능을 제공하는 서비스 (Service)와 서비스에서 출력된 결과를 응용하는 컨트롤러 (Controller) 그리고 이러한 기능을 실행 단위인 작업 (Task)으로 생성하여 에지 게이트웨이에 배포한다. 서비스와 컨트롤러는 이름과 주소 외에 필요로 하는 리소스와 센서 이름을 속성으로 전달하여 데이터베이스에 저장한다. 작업은 주어진 에지 게이트웨이와 사물인터넷 디바이스 그리고 실행하려는 서비스와 컨트롤러를 지정하여 생성한다. 서비스와 컨트롤러는 중복

등록을 방지하기 위하여 데이터베이스에 저장하기에 앞서 중복등록 확인절차를 실행하지만, 반대로 작업은 중복등록을 허용한다.

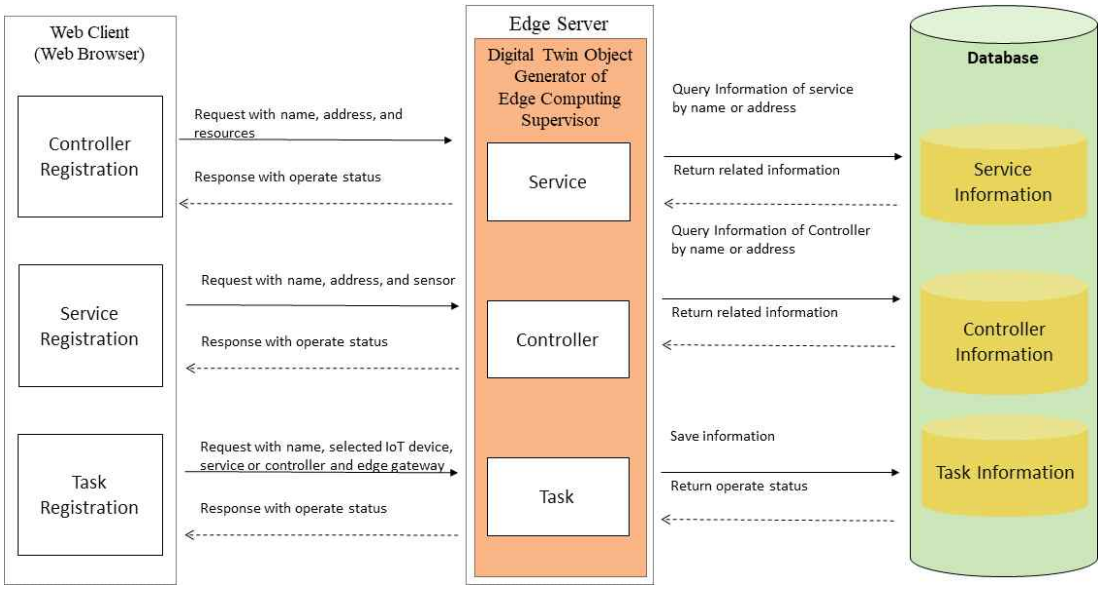


그림 3.9 에지 서버에 서비스, 컨트롤러 그리고 작업을 등록하는 기능 구조

그림 3.10은 사물인터넷 디바이스와 에지 게이트웨이의 연결을 하는 시퀀스 다이어그램이다. 사용자는 ECS가 제공하는 GUI를 통하여 드래그 앤드 드롭(Drag-and-drop)기능을 이용하여 사물인터넷 디바이스와 에지 게이트웨이의 객체를 클릭하면서 연결을 생성할 수 있다. 연결 요청이 ECS로 전달되면 디지털 트윈 오브젝트 생성자의 연결 (Connecting)기능은 중복을 확인하기 위하여 데이터베이스로 조회를 진행한다. 연결된 데이터가 없으면 전달된 식별자 (ID)를 기반으로 데이터베이스로부터 상세 정보를 조회한다. 조회된 정보를 디바이스 서비스로 전달한다. 디바이스 서비스는 전달된 정보를 에지 게이트웨이에 등록하도록 에지 게이트웨이에서 실행하고 있는 코어 메타데이터 (Core Metadata) 서비스로 요청을 보낸다. 정상적으로 등록이 완성되면 디지털 트윈 오브젝트 생성자의 연결기능에서 사용자가 요청한 연결 정보를 데이터베이스로 저장한다.

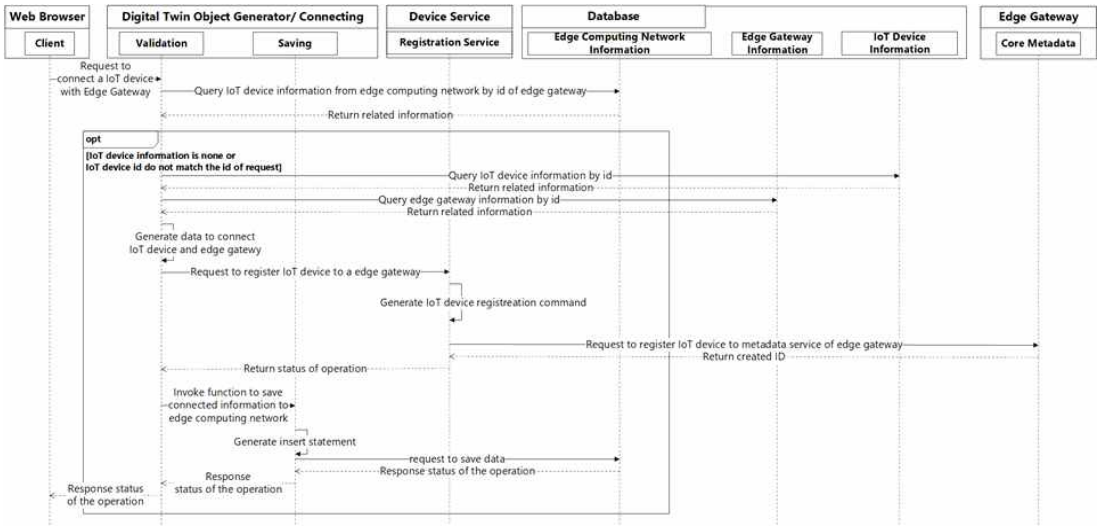


그림 3.10 사물인터넷 디바이스와 에지 게이트웨이의 연결을 하는 시퀀스 다이어그램

2. 분산 디지털 트윈 에지 컴퓨팅에서의 작업 관리 아키텍처

그림 3.11은 분산 디지털 트윈 에지 컴퓨팅 기반의 작업 관리 아키텍처이다. 작업 관리 기능은 가상환경에 등록한 에지 컴퓨팅 네트워크의 정보들에 근거하여 작업을 생성하여 물리적인 에지 네트워크에 배포하도록 지원한다. 작업 관리 컴포넌트는 작업 등록, 배포, 작업 정지 그리고 작업 결과 데이터 시각화의 기능을 제공한다. 에지 컴퓨팅 네트워크는 에지 컴퓨팅 서비스들이 실행하고 있는 에지 게이트웨이와 환경정보를 수집하는 사물인터넷 디바이스들로 구성되고 있다. 작업을 에지 게이트웨이에 배포하여 사물인터넷 디바이스의 근처에서 데이터를 수집하고 처리하여 빠른 처리속도를 제공한다. 배포한 작업은 지정한 에지 게이트웨이에서 실행하고 결과값을 데이터베이스로 전달한다. 작업의 실행상태를 실시간으로 확인할 수 있고 작업 데이터는 시각화되어 사용자에게 제공된다.

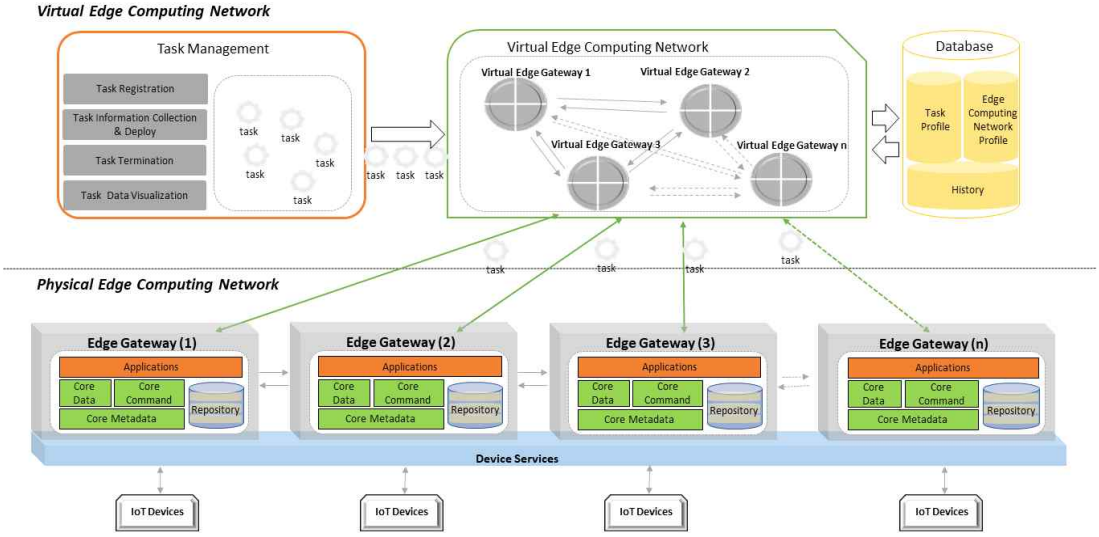


그림 3.11 물리적인 에지 컴퓨팅 환경에 작업을 배포하는 분산 디지털 트윈 에지 컴퓨팅 아키텍처

그림 3.12는 작업 관리 에이전트 (Task Management Agent, TMA)의 기능구조를 표현한다. 에지 게이트웨이에서 실행하는 응용프로그램 중의 하나이다.

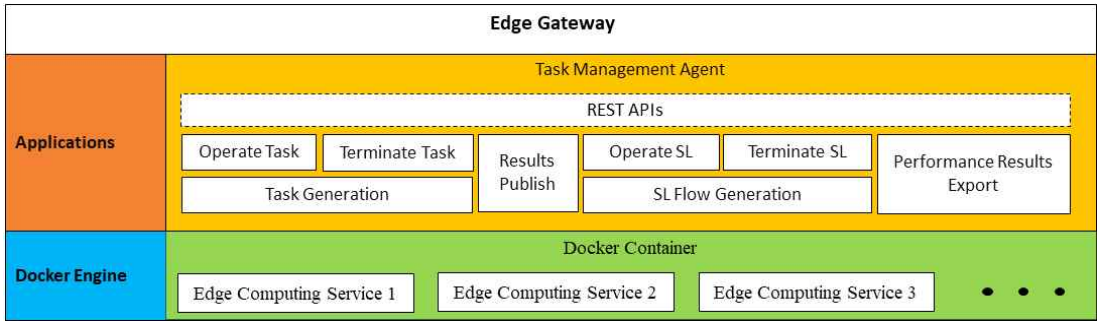


그림 3.12 에지 게이트웨이에 배포한 작업을 관리하는 에이전트 기능 구조

작업 관리로부터 생성한 작업을 에지 게이트웨이에 배포하고 실행하기 위하여 TMA가 필요하다. TMA는 작업 관리로부터 작업을 배포 받으면 작업을 실행할 수 있도록 작업 생성 (Task Generation)기능을 통하여 관련 명령어를 생성한다. 생성된 명령어는 작업 실행 (Operate Task)기능에 따라 스레드 형태로 반복적으로 실행된다. 실행과정에서 생성된 데이터는 결과 발표 (Results Publish)기능에 의하여 ECS의 데이터베이스로 저장한다. TMA에서 동시에 강제 군집 학습을 할 수 있도록 SL Flow

Generation, Operate SL, Terminate SL, 그리고 Performance Results Export 기능을 제공한다.

3. EdgeX 기반 분산 에지 컴퓨팅 환경 구축

그림 3.13은 EdgeX 기반 분산 에지 컴퓨팅 환경 기능 블록이다. 그림과 같이 에지 컴퓨팅 네트워크는 에지 게이트웨이들로 구성되고 있다. 에지 게이트웨이는 에지 컴퓨팅 서비스들을 호스팅하는 노드이다. 본 연구에서 에지 게이트웨이에서 실행하는 에지 컴퓨팅 서비스는 다양한 EdgeX서비스들이다. EdgeX 서비스들은 사물인터넷 디바이스와 연결기능을 제공하는 디바이스 서비스, 에지 컴퓨팅에서 관리하는 사물인터넷 디바이스의 정보에 관한 내용을 저장하는 코어 메타데이터 (Core Metadata), 사물인터넷 디바이스로부터 데이터를 저장하고 분배하는 코어 데이터 (Core Data), 그리고 사물인터넷 디바이스가 제공하는 서비스를 명령어 형식으로 변환하여 전달하는 코어 커맨드 (Core Command) 그 외에도 저장공간을 제공하는 저장소 (Repository)가 있다.

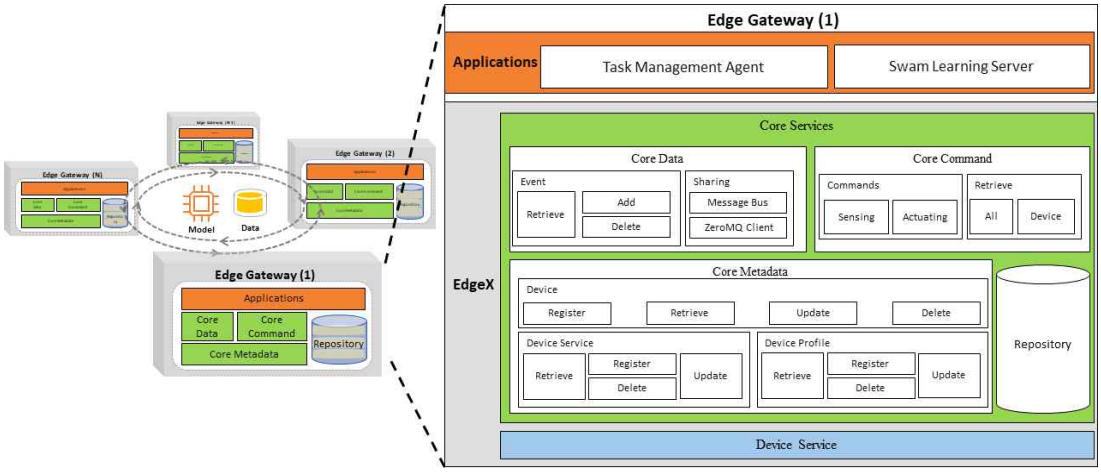


그림 3.13 EdgeX 기반 분산 에지 게이트웨이 기능 구조

코어 커맨드에서는 사물인터넷 디바이스, 사물인터넷 디바이스 프로파일, 디바이스 서비스 정보에 대한 조회, 등록, 삭제, 그리고 업데이트 기능을 각각 제공한다.

디바이스 프로파일에는 디바이스가 제공하는 서비스에 대한 정보를 포함하고 있다. 코어 데이터는 사물인터넷 디바이스로부터 수집한 데이터에 대한 조회, 추가, 삭제 기능과 응용프로그램으로 데이터를 배달하는 기능을 제공한다. 코어 커맨드는 사물인터넷 디바이스가 제공하는 서비스에 대한 조회, 명령 기능을 제공한다.

4. OCF IoTivity 기반 사물인터넷 환경 구축

그림 3.14는 OCF IoTivity 기반 분산 에지 컴퓨팅 환경 기능 블록이다. 사물인터넷 디바이스는 제조사에 따라 서로 다른 프로토콜을 사용한다. 이러한 다양성을 해결하기 위하여 표준화 단체인 OCF에서는 통일된 표준을 생성한다. 표준을 구현한 프레임워크는 IoTivity이다. IoTivity는 클라이언트와 서버의 형식으로 통신을 한다. EdgeX 에지 컴퓨팅 서비스는 디바이스 서비스를 통하여 사물인터넷 디바이스와의 연결을 생성한다. IoTivity통신을 실행하기 위하여 디바이스 서비스에서는 IoTivity Client를 구현하고 사물인터넷 디바이스에서는 IoTivity Server를 구현한다.

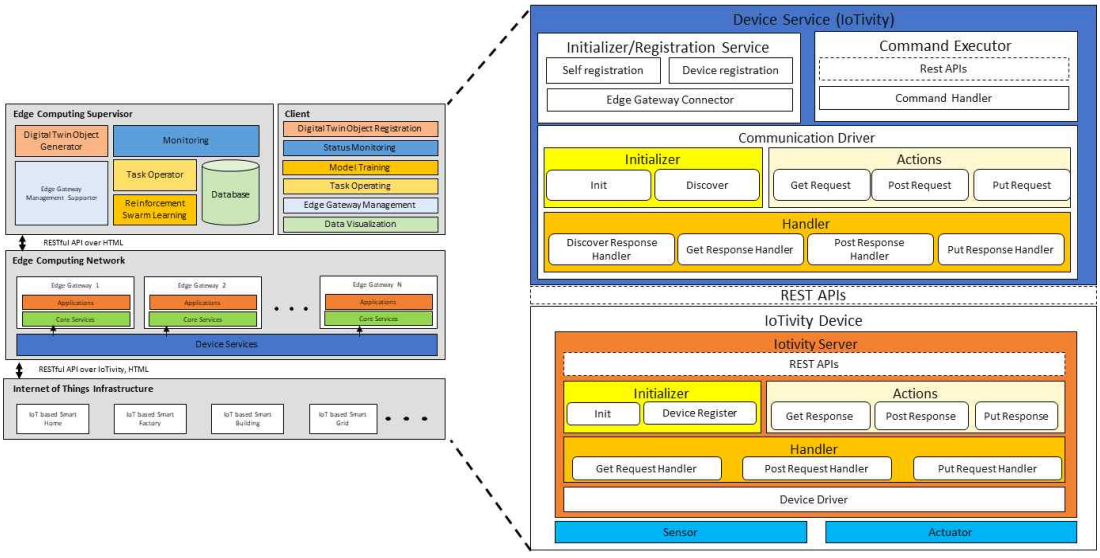


그림 3.14 OCF IoTivity 기반 사물인터넷 기능 구성

디바이스 서비스는 사물인터넷 디바이스를 등록하는 기능, 코어 커맨드로 부터 전달해온 명령어를 실행하는 커맨드 실행자 (Command Executor), 통신 드라이브

(Communication Driver)등의 기능으로 구성되고 있다. IoTivity 클라이언트를 구현한 통신 드라이브는 초기화 (Initializer), 활동들 (Actions), 핸들러 (handler)로 구성되고 있다. 초기화를 통하여 IoTivity 프레임워크를 활성화한다. 커맨드 실행자로부터 여러 활동 중의 하나를 실행하면 상응한 핸들러가 IoTivity 서버에서의 응답을 처리하게 된다.

IoTivity 사물인터넷 디바이스는 IoTivity에서 필요한 초기화 (Initializer), 활동들 (Actions), 핸들러 (handler)기능 외에 디바이스 드라이브 (Device Deriver), 그리고 센서와 엑츄에이터가 연결된다. 센서와 엑츄에이터는 고유한 드라이브를 통하여 작동한다. IoTivity 서버는 초기화 단계에서 IoTivity 프레임워크를 활성화하는 동시에 리소스를 핸드러와 연결을 한다. IoTivity 클라이언트로부터 요청이 전달되면 핸들러가 상응한 활동을 호출하여 응답을 한다.

5. EdgeX와 OCF IoTivity 기반 디지털 트윈 에지 컴퓨팅 환경 구축 및 결과

분산 에지 컴퓨팅 실험 환경은 그림 3.15와 같이 구축한다. 실험 환경은 에지 서버, 에지 컴퓨팅 네트워크 그리고 사물인터넷 디바이스들로 구성되고 있다.

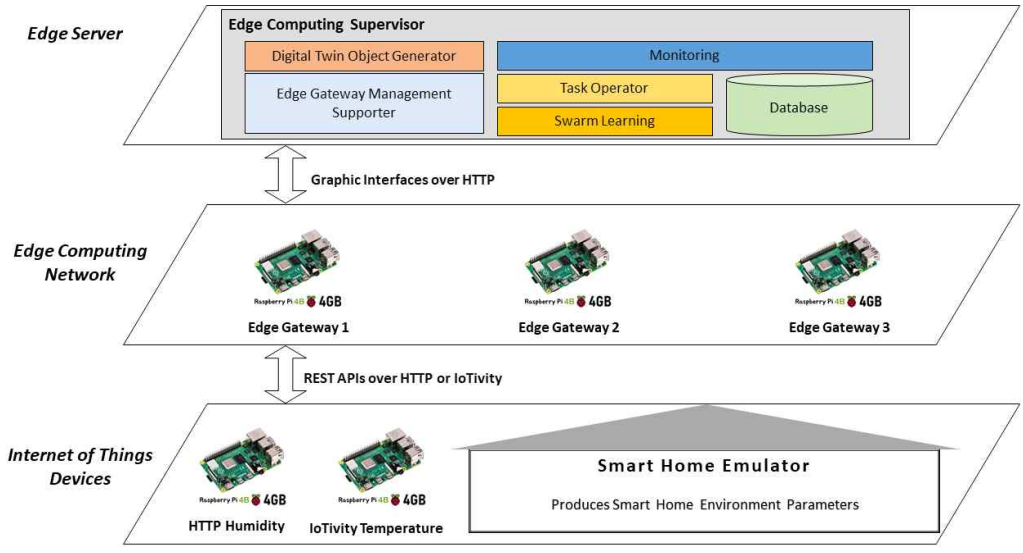


그림 3.15 디지털 트윈 기반 물리적인 분산 에지 컴퓨팅 실험 환경

에지 서버의 에지 컴퓨팅 슈퍼바이저를 통하여 사용자는 작업을 등록, 실행 및 결과를 확인 할 수 있다. 에지 컴퓨팅 네트워크는 3대의 라즈베리파이로 구성한다. EdgeX 마이크로서비스들이 각각의 디바이스에서 운영하여 사물인터넷 디바이스의 등록, 통신, 명령어 전달 기능을 제공한다. 그리고 사물인터넷 디바이스는 HTTP 프로토콜과 IoTivity 프레임워크를 구현한 디바이스와 스마트 홈 에뮬레이터를 포함하고 있다. HTTP와 IoTivity 디바이스를 통하여 온도와 습도 정보를 수집할 수 있고 스마트 홈 에뮬레이터는 실내 온도, 습도, 실외 온도, 습도, 히터 전력 그리고 컨트롤러를 이용하여 실내 환경 변화도 반영할 수 있다.

실험을 진행하기에 앞서 먼저 사물인터넷 디바이스, 에지 게이트웨이 그리고 에지 컴퓨팅 슈퍼바이저 개발환경에 관하여 알아본다. 에지 게이트웨이는 라즈베리 파이4 하드웨어에 우분투 20.04 버전을 설치한다. 개발언어는 python 3.8 버전을 사용한다. Python 언어는 우분투 운영체제에서 기본으로 제공하는 언어이고 간결하고 문법이 쉬운 고급 프로그래밍 언어이다. 본고에서 제안하는 스웜 학습 기능을 구현하기 위하여 기계학습 구현에 널리 사용되는 플랫폼인 TensorFlow 2.6버전을 설치한다. TensorFlow를 이용하여 지도학습을 통하여 모델을 생성하고 생성된 모델을 이용하여 예측기능도 구현한다. 생성된 기능을 REST APIs로 제공하기 위하여 웹 어플리케이션 개발에 사용되는 Flask 프레임워크도 설치한다. Flask를 이용하여 서버를 구축하고 외부에서 호출할 수 있도록 REST APIs도 구현한다. 라즈베리 파이의 구현을 원격에서 쉽게 실현하기 위하여 Visual Studio Code 에디터를 설치하여 원격접속을 통하여 프로그래밍, 실행, 디버깅을 한다. 표 3.1은 에지 게이트웨이 개발환경 상세정보이다.

표 3.1 에지 게이트웨이 개발환경

<table>
<tr><th colspan="3">Hardware</th><th colspan="2">Software</th></tr>
<tr><td rowspan="6">Raspberry Pi 4</td><td>OS</td><td>Ubuntu 20.04 64bit</td><td rowspan="2">Programming Language</td><td rowspan="2">Python 3.8</td></tr>
<tr><td>CPU</td><td>Quad Core 1.5GHz 64bit</td></tr>
<tr><td>Memory</td><td>4GB</td><td>Application</td><td>Visual Studio Code</td></tr>
<tr><td rowspan="3">MicroSD Card</td><td rowspan="3">16GB</td><td>Framework</td><td>Flask</td></tr>
<tr><td rowspan="2">Platform</td><td>TensorFlow 2.6</td></tr>
<tr><td>Docker & Docker Compose</td></tr>
</table>

사물인터넷 디바이스를 구현하기 위하여 라즈베리 파이3 하드웨어에 우분투 20.04 버전을 설치한다. 개발언어는 자바 1.8 버전을 사용한다. 자바는 이식성이 높은 언어로서 거의 모든 운영 체제에서 실행할 수 있다. 사물인터넷 통신 표준을 구현한 IoTivity 프레임워크를 기반으로 데이터를 제공할 수 있도록 서버를 구축한다. 서버는 자바언어를 기반으로 하는 Spring Boot 프레임워크를 이용하여 구축한다. 별도로 서버를 구현하지 않고 프레임워크만 이용하여 서버가 실행된다. 개발 툴은 자바 프로그래밍 언어에 널리 사용되는 Eclipse를 사용한다. 표 3.2에서 사물인터넷 디바이스 개발환경을 상세히 설명한다.

표 3.2 사물인터넷 디바이스 개발환경

<table>
<tr><th colspan="3">Hardware</th><th colspan="2">Software</th></tr>
<tr><td rowspan="5">Raspberry Pi 3</td><td>OS</td><td>Ubuntu 20.04 64bit</td><td rowspan="2">Programming Language</td><td rowspan="2">Java 1.8</td></tr>
<tr><td>CPU</td><td>Quad Core 1.2GHz 64bit</td></tr>
<tr><td>Memory</td><td>1GB</td><td>Application</td><td>Eclipse</td></tr>
<tr><td rowspan="2">MicroSD Card</td><td rowspan="2">16GB</td><td rowspan="2">Framework</td><td>Spring Boot</td></tr>
<tr><td>IoTivity</td></tr>
</table>

에지 컴퓨팅 슈퍼바이저는 디지털 트윈 에지 컴퓨팅 환경을 구축하는 서버이다. 데스크톱 컴퓨터에서 가상 머신을 실행하여 우분투 18.04 운영체제를 설치한다. 개발언어는 우분투 운영체제에서 기본으로 제공하는 python 3.8 버전을 사용한다. 프런트 엔드 (Front End) 개발언어로는 HTML5, CSS 3, 그리고 자바스크립트를 사용한다. 프런트 엔드 통하여 가상의 에지 게이트웨이와 사물인터넷 디바이스 객체를 생성하였고 사용자와의 상호 운용성을 제공한다. 프런트 엔드의 가시화의 효과를 향상하기 위하여 Bootstrap 프레임워크를 사용하여 색상과 위젯, 애니메이션의 효과를 향상한다. 서버를 구축하기 위하여 Flask 프레임워크를 설치한다. Flask는 Python 개발언어를 기반으로 하고 웹 애플리케이션을 개발하는 데 필요로 하는 라이브러리를 제공한다. 프로그램에서 사용되는 데이터를 저장하기 위하여 데이터베이스로 MariaDB를 구축한다. 표 3.3에서 에지 컴퓨팅 슈퍼바이저 개발환경을 설명한다.

표 3.3 에지 서버 (Edge Computing Supervisor) 개발환경

Hardware			Software	
Desktop (Virtual Machine)	OS	Ubuntu 18.04 64bit	Programming Language	Python 3.8
	CPU	Intel® core ™ i5-8500 3GHz		
	Memory	8GB	Application	Visual Studio Code
	Hard Disk	100GB	Framework	Flask
				Bootstrap
			Database	MariaDB

분산 디지털 트윈 에지 컴퓨팅 구현 구조는 그림 3.16과 같다. 에지 컴퓨팅은 3대의 라즈베리파이를 이용하여 구성한다. 각각의 라즈베리파이에서 에지 컴퓨팅 서비스들을 운영하고 있다. 에지 게이트웨이를 구분하기 위하여 각각의 에지 게이트웨이에 대하여 이름을 접두사 “EdgeGW-KR-JNU-” 와 번호 “001” 로 생성한다. 에지 게이트웨이는 001에서 003까지의 번호를 부여한다.

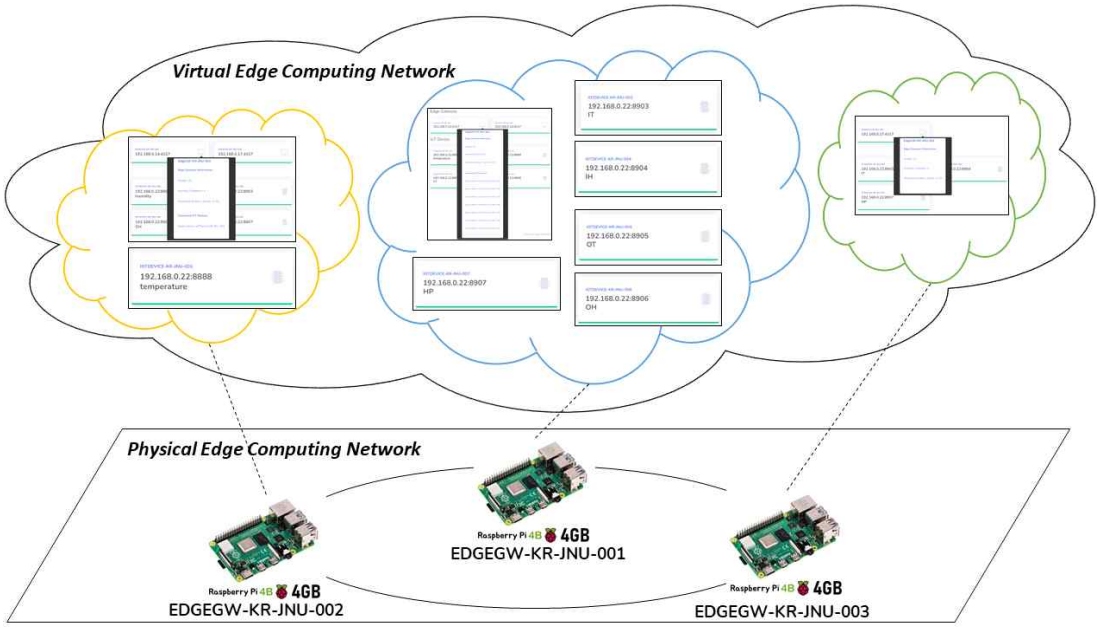


그림 3.16 라즈베리 파이를 이용한 분산 디지털 트윈 에지 컴퓨팅 구현 구조

에지 컴퓨팅 네트워크의 실행상태를 실시간으로 확인하고, 정보를 파악하기 위하여 물리적인 자원은 객체로 추상화하여 웹 패지에서 카드의 형태로 표현한다. 물리적인 객체를 표현하는 카드를 통하여 라즈베리파이에서 실행하고 있는 에지 컴퓨팅

서비스의 실행상태를 한눈에 확인할 수 있다. 에지 컴퓨팅을 구성하는 객체 즉 에지 게이트웨이 그리고 그와 연결된 사물인터넷 디바이스들의 정보를 쉽게 확인할 수 있도록 마우스가 가리키는 객체에 대하여 팝업 기능을 추가하여 정보를 나열하도록 한다.

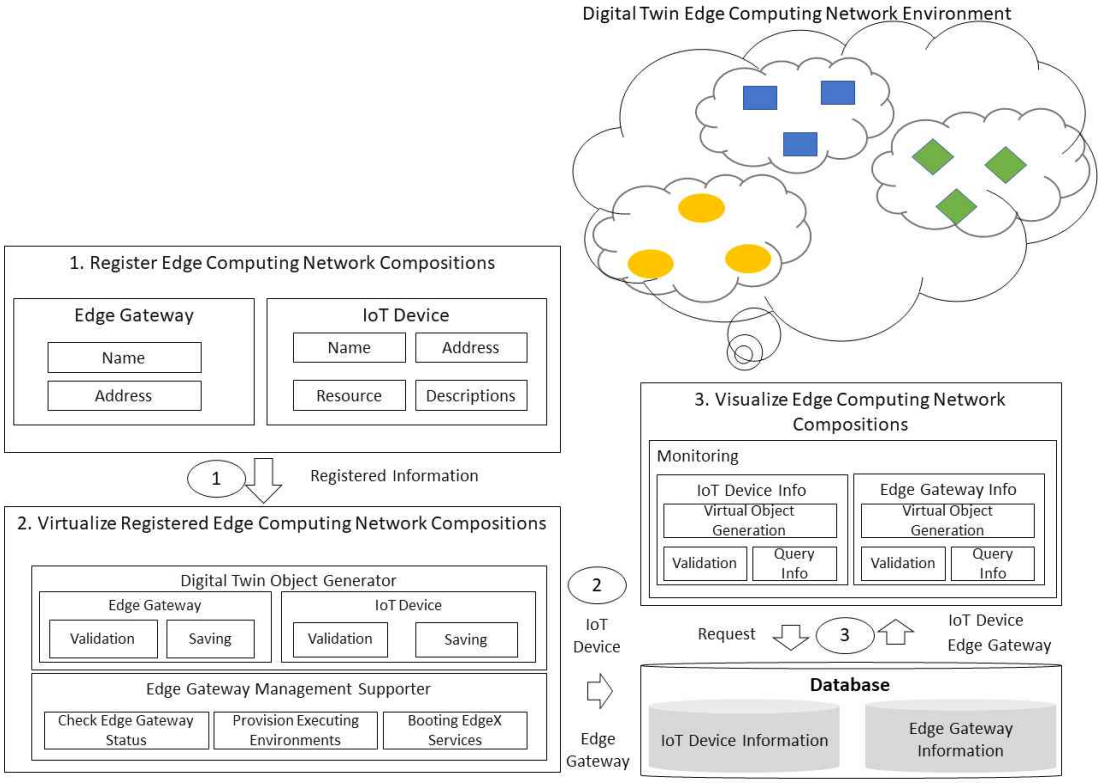


그림 3.17 분산 디지털 트윈 에지 컴퓨팅 환경 생성과정

물리적인 에지 컴퓨팅 환경을 디지털 트윈 에지 컴퓨팅 환경으로 변환하기 위하여 그림과 3.17과 같이 진행한다. 사용자는 에지 컴퓨팅 슈퍼바이저에서 제공하는 웹 페이지를 이용하여 에지 컴퓨팅을 구성하는 에지 게이트웨이와 사물인터넷 디바이스의 정보를 등록한다. 에지 게이트웨이는 이름과 네트워크 주소에 대한 정보를 요구하고 사물인터넷 디바이스는 이름, 네트워크 주소, 제공하는 자원과 간단한 설명 정보를 요구한다. 등록된 정보에 근거하여 에지 컴퓨팅 슈퍼바이저는 에지 게이트웨이와 사물인터넷 디바이스를 가상화한다. 먼저 네트워크 주소를 통하여 기기의 실행 상태를 파악한다. 에지 게이트웨이가 정상적으로 실행하고 있으면 에지 컴퓨팅 서비스들을 실행하고 에지 게이트웨이 관련 정보를 데이터베이스로 저장한다. 저장

된 정보는 물리적인 기기에 대한 가상화된 데이터 표현이다. 마지막으로 사용자가 웹 브라우저를 통하여 에지 컴퓨팅 슈퍼바이저에서 제공하는 웹 페이지를 접속하면 가상화된 에지 게이트웨이와 사물인터넷 디바이스의 객체를 확인할 수 있다. 에지 컴퓨팅 슈퍼바이저는 데이터베이스를 질의하여 저장한 정보를 수집하고 네트워크 주소를 이용하여 기기의 실행상태 및 상세 정보를 요청한다. 수집된 정보를 기반으로 웹 페이지에서 각각의 기기를 카드 형태로 추상화하여 표현한다.

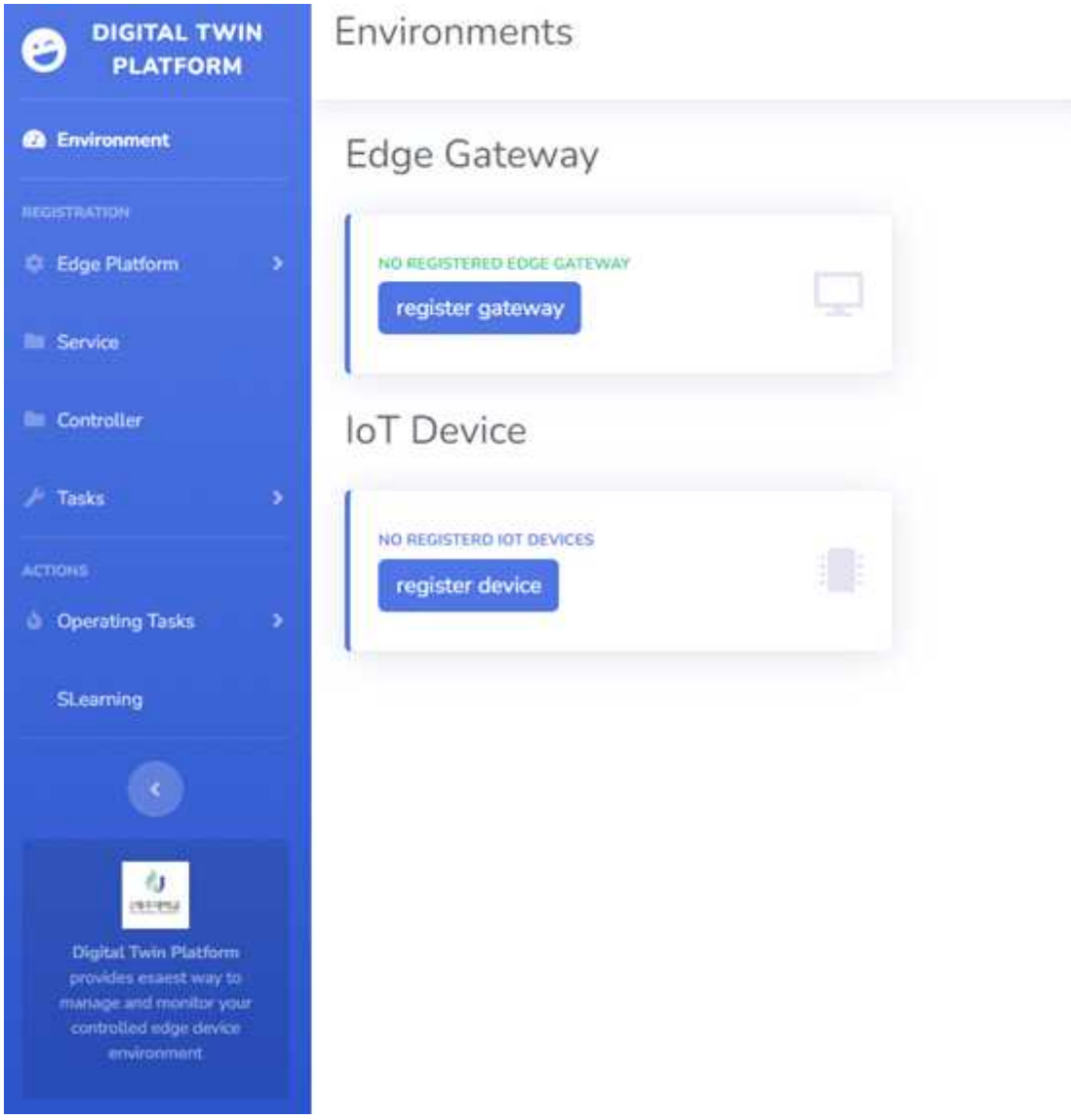


그림 3.18 에지 서버 (Edge Computing Supervisor)의 초기화 메인화면

그림 3.18는 EGMS에서 제공하는 GUI의 초기화 메인 화면이다. 외쪽의 사이드 바

에서는 여러 옵션들이 나열되어 있다. 크게 등록 (Registration), 활동 (Actions)의 두 개 부분으로 구성되고 있다. 등록에는 에지 플랫폼 (Edge Platform), 서비스 (Service), 컨트롤러 (Controller), 작업들 (Tasks) 링크들이 있는데, 에지 플랫폼은 사물인터넷 디바이스, 에지 게이트웨이, 그리고 연결을 생성하는 기능을 제공하는 링크이다. 서비스와 컨트롤러는 각각 서비스 기능 그리고 컨트롤러 기능을 생성하는 링크이다. 마지막 작업 단계는 위에서 등록한 내용을 이용하여 실행하려는 작업을 생성할 수 있도록 하는 링크이다. 활동에는 작업 실행 (Operating Tasks)과 SL를 실행할 수 있는 링크를 제공한다.

가운데 메인 화면에서는 에지 게이트웨이와 사물인터넷 상태와 간단한 정보를 확인할 수 있도록 각각 카드 형태로 표현되어 있다. 처음 시작한 상태로 지금은 아무런 정보가 없기 때문에 에지 게이트웨이 그리고 사물인터넷 디바이스에서 각각의 등록 화면으로 이동할 수 있는 링크를 버튼으로 표현하고 있다.

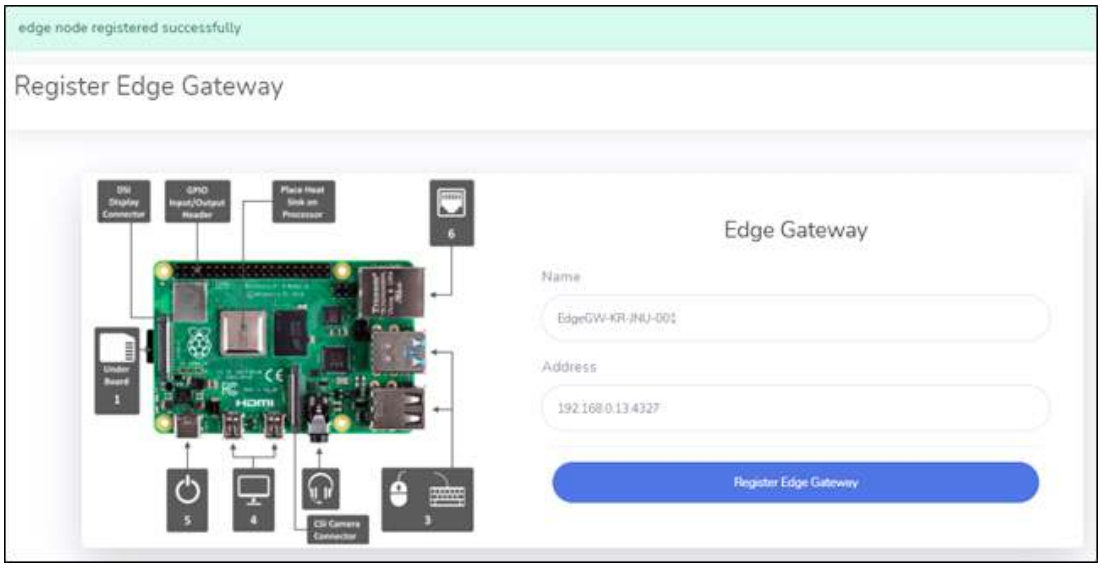


그림 3.19 첫 번째 에지 게이트웨이 등록화면

에지 게이트웨이 등록화면은 그림 3.19와 같다. 외쪽 화면에 저희가 사용하는 라즈베리 파이의 이미지를 출력한다. 라즈베리 파이를 이용하여 에지 게이트웨이를 구축한다는 것을 확인 할 수 있다. 사용자는 에지 게이트웨이의 이름과 주소 정보를 입력하고 등록 버튼을 클릭함으로써 등록 작업을 쉽게 완성할 수 있다. 이름은 에지 게이트웨이를 구분하는 식별자로 사용되기 때문에 서로 다른 이름을 사용하여 중복

을 피해야 한다. 운영체제에 설정한 호스트 이름과 같은 이름을 사용할 것을 추천한다. 주소는 인터넷에서 단말을 식별할 수 있는 IP (Internet of protocol)주소와 서비스가 사용하는 포트 번호가 포함되어야 한다. 에지 컴퓨팅 서비스가 도커 엔진에 의해 제어 되기 때문에 도커 엔진에서 허용하는 원격 접속 포트를 지정한다. 정상적으로 등록이 완성되면 성공적으로 등록되고 있다는 문구가 위쪽에 표현이 된다.

그림 3.20 두 번째 에지 게이트웨이 등록화면

그림 3.20은 에지 게이트웨이 “EdgeGW-KR-JNU-002”를 등록하는 화면이다. 에지 게이트웨이의 이름은 접두사 “EdgeGW-KR-JNU-”와 번호 “002”로 구성되고 있다. 에지 컴퓨팅 네트워크를 구성하는 에지 게이트웨이들을 구별하기 위하여 같은 접두사와 다른 번호를 접미사로 결합하여 에지 게이트웨이의 이름을 구성한다. 네트워크 주소는 라우터에서 부여한 IP주소로 설정한다. 에지 게이트웨이 EdgeGW-KR-JNU-002는 네트워크 주소가 192.168.0.14이고 포트는 4327이다. 포트번호는 라즈베리파이에서 실행하는 도커 엔진이 사용하는 포트번호를 가리킨다. 에지 컴퓨팅 서비스들은 쉽게 실행할 수 있도록 도커 기반의 컨테이너로 실행한다. 도커 엔진을 통하여 실행하고 있는 서비스의 상태, 이름 등의 정보를 확인할 수 있다. 그뿐만 아니라 서비스 컨테이너 생성, 실행, 중지 등 작업도 수행할 수 있다. 에지 컴퓨팅 슈퍼바이저는 등록된 에지 게이트웨이를 이용하여 분산 에지 컴퓨팅 환경에서 기계학습을 통하여 예측 모델을 훈련할 수 있도록 한다. 에지 게이트웨이들의 컴퓨팅 자원과 로컬 데이터를 이용하여 분산 환경에서 네트워크를 통하여 모델을 전달하여 예측 모델을 훈련하는 방법을 사용한다.

그림 3.21은 에지 게이트웨이 “EdgeGW-KR-JNU-003”을 등록하는 화면이다. 그림 3.16 분산 디지털 트윈 에지 컴퓨팅 구현 구조에서 설명한 것과 같이 본고에서는 3대의 라즈베리파이를 이용하여 에지 컴퓨팅 네트워크를 구성한다. 그중 3번째 에지 게이트웨이의 네트워크 주소는 192.168.0.17이다. 에지 게이트웨이가 같은 네트워크 도메인에서 실행하여 서로 통신을 할 수 있도록 설정한다. 에지 컴퓨팅 네트워크를 구성하는 모든 에지 게이트웨이들의 등록이 완성되고 있다. 에지 게이트웨이는 사물인터넷 디바이스로부터 데이터를 수집하고 명령어를 전달하는 역할을 한다. 각각의 에지 게이트웨이는 와이파이를 통하여 라우터와 연결되었고 같은 도메인에 설치하여 서로 통신을 할 수 있다. 그리고 도커 엔진의 포트 번호는 모두 4327로 설정한다.

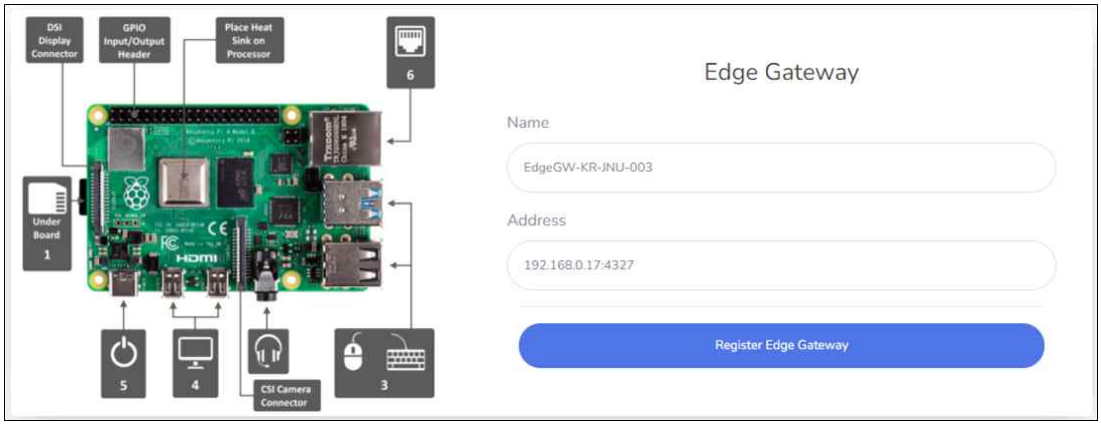


그림 3.21 세 번째 에지 게이트웨이 등록화면

사물인터넷 디바이스 등록화면은 그림 3.22와 같다. 사용자는 사물인터넷 디바이스 이름, 주소, 자원 그리고 설명 정보를 입력하고 등록 버튼을 클릭함으로써 등록 작업을 완성한다. 이름은 사물인터넷 디바이스를 구분하는 식별자로 사용되기 때문에 서로 다른 이름을 사용한다. 주소는 인터넷에서 단말을 식별할 수 있는 IP (Internet of protocol) 주소와 서비스가 사용하는 포트 번호가 포함되어야 한다. 자원은 디바이스에서 제공하는 정보에 대한 표현이다. 설명은 디바이스에 대한 이해를 돕는 데 있으므로 디바이스에 대한 간단한 설명을 작성하면 된다. 문제없이 등록이 완성되면 성공적으로 등록되고 있다는 문구가 위쪽에 표현이 된다.

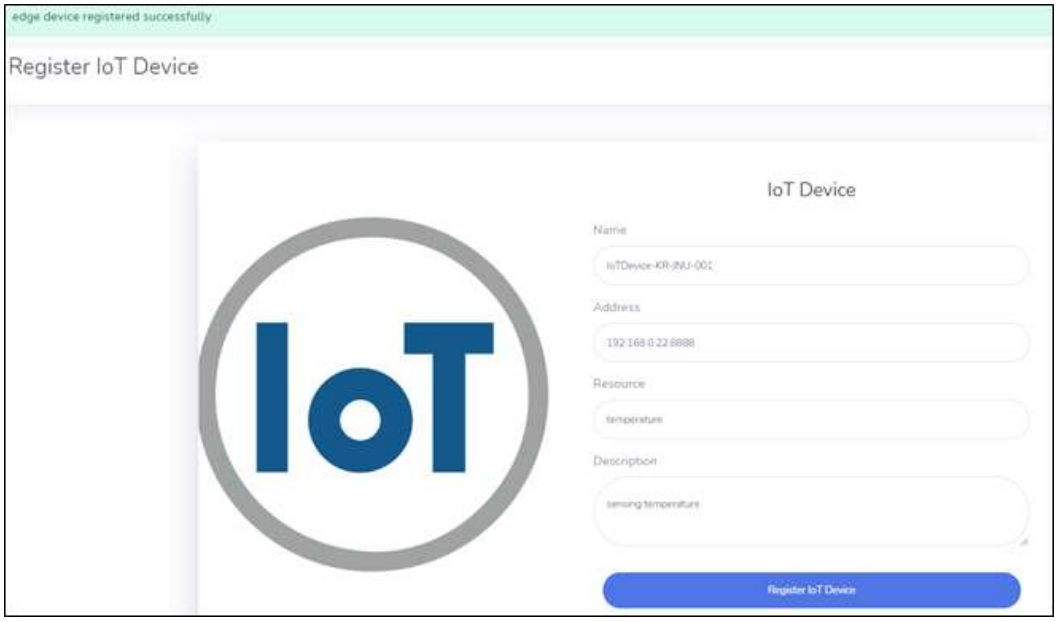


그림 3.22 OCF IoTivity 기반 사물인터넷 디바이스 등록화면

그림 3.23은 사물인터넷 디바이스 “IoTDevice-KR-JNU-002”를 등록하는 화면이다. 사물인터넷 디바이스의 이름은 접두사 “IoTDevice-KR-JNU-”와 번호 “002”로 구성되고 있다.

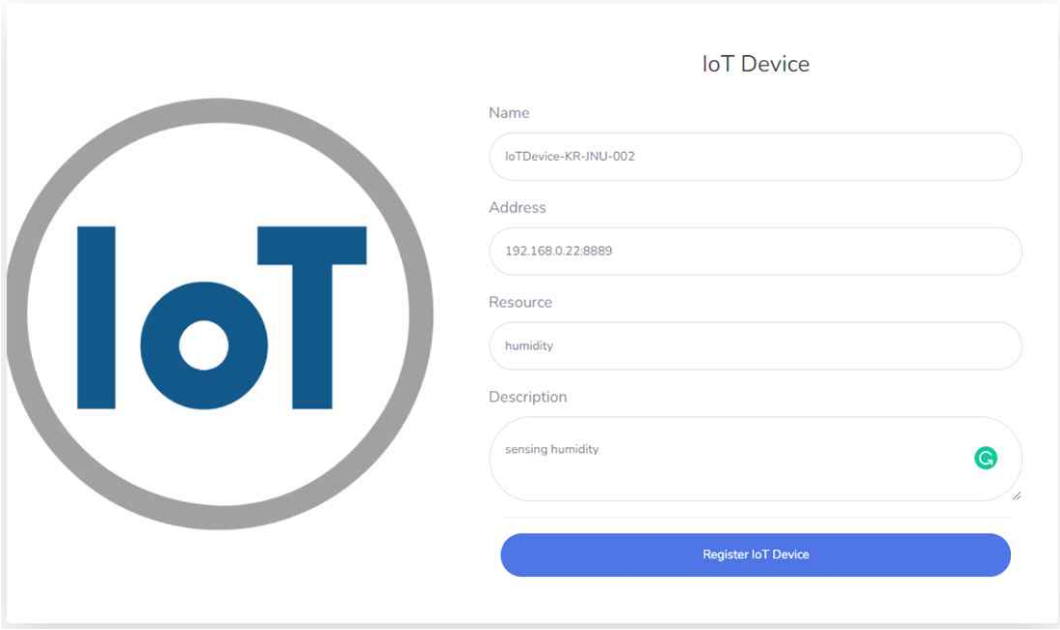


그림 3.23 HTTP 기반 사물인터넷 디바이스 등록화면

자원 정보에 입력한 것을 보면 알 수 있듯이 사물인터넷 디바이스 “IoTDevice-KR-JNU-002”는 습도를 제공하는 센서 장치이다. 네트워크 주소는 192.168.0.22이고 포트는 8889로 설정한다. 설명에 이 디바이스가 습도 정보를 제공한다고 명시하여 확인할 수 있도록 한다. 이 디바이스는 HTTP프로토콜을 통하여 습도 정보를 수집할 수 있도록 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 클라이언트는 이 디바이스가 제공하는 REST API를 HTTP프로토콜을 통하여 접근하여 습도 정보를 수집할 수 있다.

그림 3.24는 사물인터넷 디바이스 “IoTDevice-KR-JNU-003”을 등록하는 화면이다. 그림 3.15 분산 에지 컴퓨팅 실험 환경에서 스마트 홈 에뮬레이터에서 제공하는 실내 온도 관련 정보를 수집하는 사물인터넷을 등록하는 화면이다.

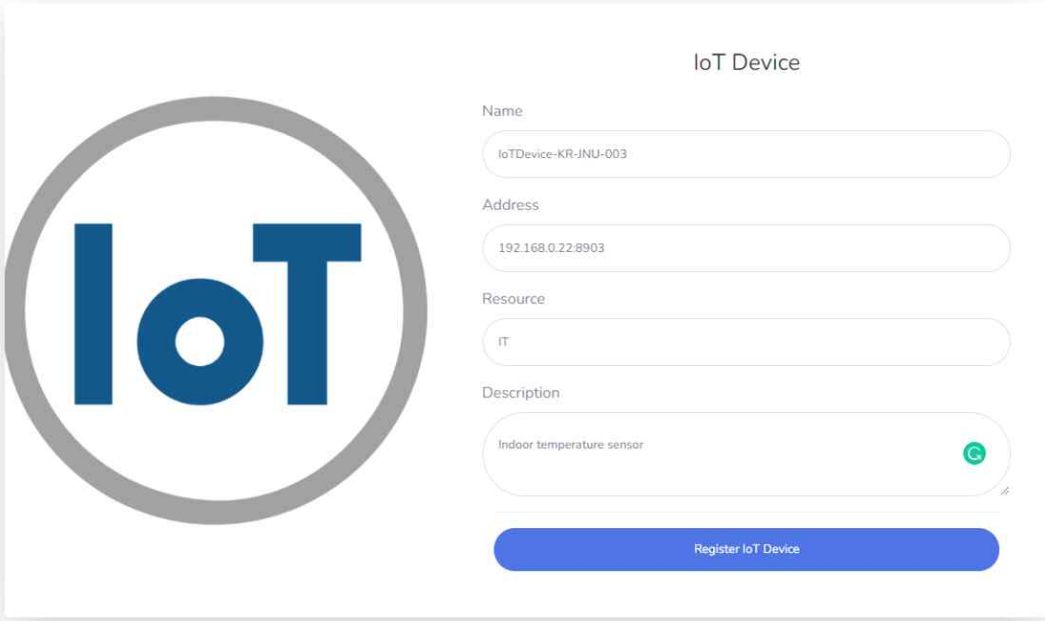


그림 3.24 스마트 홈 에뮬레이터 기반 실내 온도 사물인터넷 디바이스 등록화면

자원 정보를 IT (Indoor Temperature)로 설정한다. 네트워크 주소는 192.168.0.22이고 포트는 8903으로 설정한다. 설명에 이 디바이스가 실내 온도 정보를 제공한다고 명시하고 있는 것을 확인할 수 있다. 이 디바이스는 스마트 홈 에뮬레이터의 실내 온도를 수집할 수 있는 가상의 사물인터넷 디바이스이고 HTTP 프로토콜을 통하여 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 이 디바이스와 연결된 에

지 게이트웨이는 REST API를 호출하여 스마트 홈 에뮬레이터에서 제공하는 실내 온도 정보를 수집할 수 있다. 스마트 홈 에뮬레이터는 스마트 홈에 관한 상태 정보 데이터를 제공한다. 스마트 홈 에뮬레이터에서 작동하고 있는 컨트롤러를 이용하여 현재 상태 정보에 변화를 가져다 줄 수 있다. 에지 게이트웨이는 사물인터넷 디바이스를 통하여 스마트 홈 상태 정보를 수집하고 컨트롤러를 통하여 현재 스마트 홈 상태를 변환하여 실제 물리적인 스마트 홈을 제어하는 것 철럼 실험을 진행 할 수 있다. 스마트 홈 에뮬레이터는 프로그램으로 구현하여 데스크톱에서 실행한다.

그림 3.25는 사물인터넷 디바이스 “IoTDevice-KR-JNU-004”를 등록하는 화면이다. 스마트 홈 에뮬레이터 실내 습도 관련 정보를 수집하는 사물인터넷을 등록하는 화면이다.

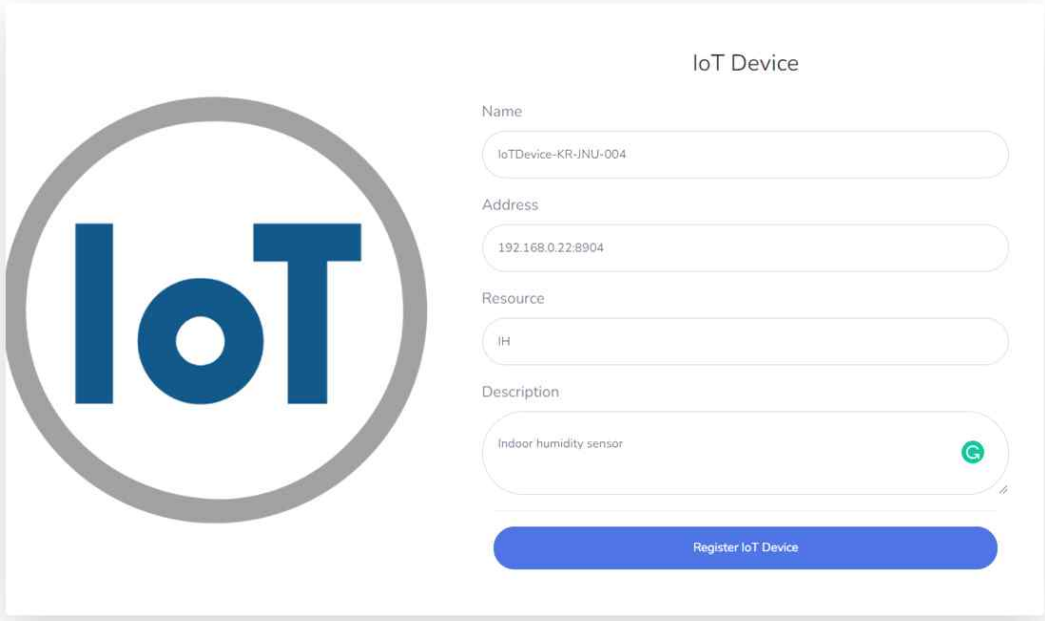


그림 3.25 스마트 홈 에뮬레이터 기반 실내 습도 사물인터넷 디바이스 등록화면

자원 정보를 IH (Indoor Humidity)로 설정한다. 네트워크 주소는 192.168.0.22이고 포트는 8904로 설정한다. 설명에 이 디바이스가 실내 습도 정보를 제공한다고 명시하여 확인할 수 있도록 한다. 이 디바이스는 스마트 홈 에뮬레이터의 실내 습도를 수집할 수 있는 가상의 사물인터넷 디바이스이고 HTTP 프로토콜을 통하여 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 이 디바이스와 연결된 에지 게이트

웨이는 REST API를 호출하여 스마트 홈 에뮬레이터에서 제공하는 실내 습도 정보를 수집할 수 있다. 에지 컴퓨팅 슈퍼바이저를 통하여 에지 게이트웨이에 사물인터넷 디바이스로부터 데이터를 수집하는 작업을 배포하면 지정된 에지 게이트웨이는 작업을 실행한다. 작업은 지정한 에지 게이트웨이를 통하여 사물인터넷 디바이스와 통신을 한다. 작업에 포함된 명령어를 사물인터넷 디바이스로 전달하여 데이터를 수집한다. 사물인터넷 디바이스는 데이터를 사물인터넷 게이트웨이로 전달하고 에지 게이트웨이는 전달된 데이터를 실행된 작업을 통하여 데이터베이스로 저장한다. 저장된 데이터는 실행한 작업의 결과 값으로 가시화하여 사용자에서 작업의 실행결과를 보여준다.

그림 3.26은 스마트 홈 에뮬레이터 실외 온도 관련 데이터를 수집하는 사물인터넷 디바이스를 에지 게이트웨이 슈퍼바이저에 등록하는 화면이다.

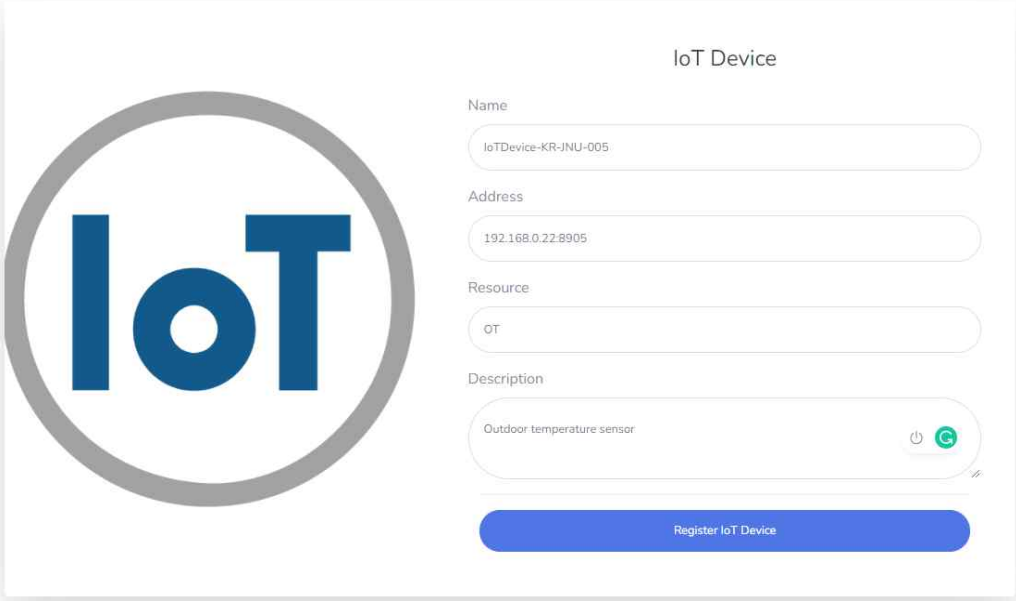


그림 3.26 스마트 홈 에뮬레이터 기반 실외 온도 사물인터넷 디바이스 등록화면

등록하려는 사물인터넷 디바이스의 이름은 “IoTDevice-KR-JNU-005” 로 설정하였고 자원 정보를 OT (Outdoor Temperature)로 설정한다. 네트워크 주소는 192.168.0.22이고 포트는 8905로 설정한다. 설명에 실외 온도 데이터를 제공하는 센서 장치라고 명시한다. 이 디바이스는 스마트 홈 에뮬레이터의 실외 온도를 수집할

수 있는 가상의 사물인터넷 디바이스이고 HTTP프로토콜을 통하여 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 이 디바이스와 연결된 에지 게이트웨이는 REST API를 호출하여 스마트 홈 에뮬레이터에서 제공하는 실외 온도 데이터를 수집할 수 있다.

그림 3.27은 스마트 홈 에뮬레이터 실외 습도 관련 정보를 수집하는 사물인터넷을 등록하는 화면이다.

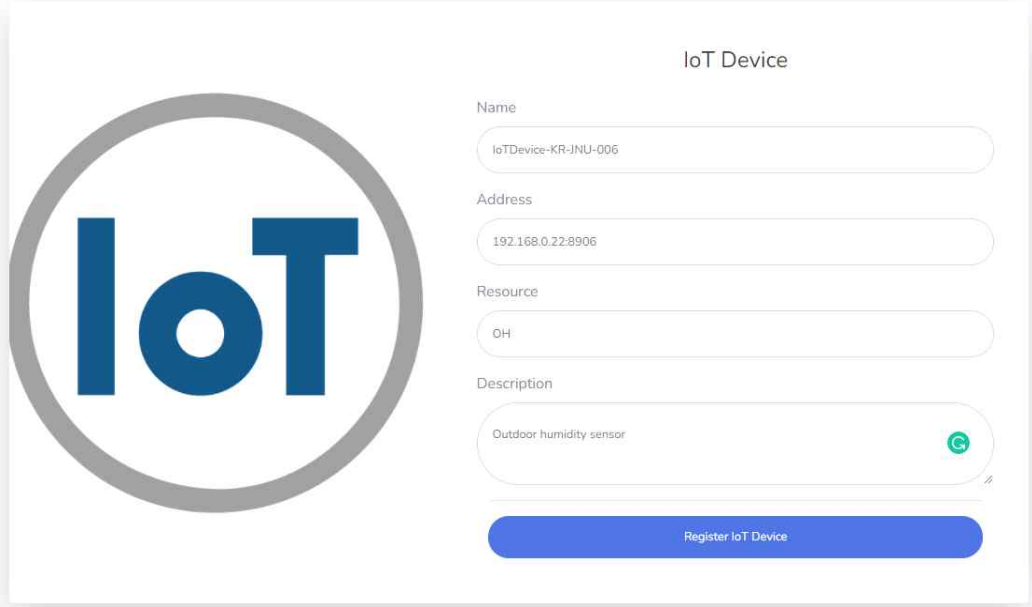


그림 3.27 스마트 홈 에뮬레이터 기반 실외 습도 사물인터넷 디바이스 등록화면

사물인터넷 디바이스 이름은 “IoTDevice-KR-JNU-006”로 설정하였고 자원 정보를 OH (Outdoor Humidity)로 설정한다. 네트워크 주소는 192.168.0.22이고 포트는 8906으로 설정한다. 설명에 실외 습도 데이터를 제공하는 센서 장치라고 명시한다. 이 디바이스는 스마트 홈 에뮬레이터의 실외 습도를 수집할 수 있는 가상의 사물인터넷 디바이스이고 HTTP 프로토콜을 통하여 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 이 디바이스와 연결된 에지 게이트웨이는 REST API를 호출하여 스마트 홈 에뮬레이터에서 제공하는 실외 습도 데이터를 수집할 수 있다.

그림 3.28은 스마트 홈 에뮬레이터 히터 전력 관련 정보를 수집하는 사물인터넷을 등록하는 화면이다. 사물인터넷 디바이스 이름은 “IoTDevice-KR-JNU-007”로 설

정하였고 자원 정보를 HP로 설정한다. 네트워크 주소는 192.168.0.22이고 포트는 8907으로 설정한다. 설명에 히터 전력 데이터를 수집하는 센서 장치라고 명시한다. 이 디바이스는 스마트 홈 에뮬레이터의 히터 전력 데이터를 수집할 수 있는 가상의 사물인터넷 디바이스이고 HTTP 프로토콜을 통하여 REST APIs를 제공한다. 같은 네트워크 도메인에 있는 이 디바이스와 연결된 에지 게이트웨이는 REST API를 호출하여 스마트 홈 에뮬레이터에서 제공하는 히터 전력 데이터를 수집할 수 있다. 스마트 홈 에뮬레이터는 실내 온도, 습도, 실외 온도, 습도 그리고 히터 전력 데이터를 가상의 사물인터넷 디바이스들을 통하여 제공한다. 각각의 사물인터넷 디바이스는 고유의 네트워크 주소를 가지고 있으며 HTTP를 통하여 REST API를 제공하여 인터넷을 이용하여 접근할 수 있도록 한다. 모든 사물인터넷 디바이스들은 에지 컴퓨팅 슈퍼바이저를 통하여 데이터베이스에 저장이 되고 있다.

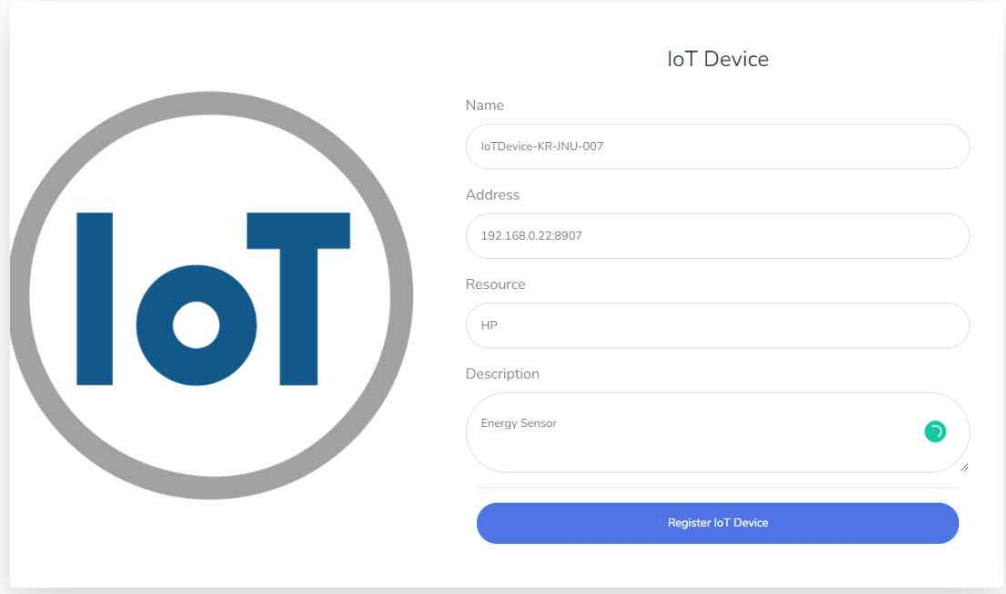


그림 3.28 스마트 홈 에뮬레이터 기반 히터 전력 사물인터넷 디바이스 등록화면

그림 3.29은 사물인터넷 디바이스와 에지 게이트웨이 연결을 생성하는 결과화면이다. 에지 컴퓨팅 네트워크에서 실행하는 모든 사물인터넷 디바이스들과 통신을 하기 위하여 에지 게이트웨이와 연결을 생성해야한다. 본고에서는 에지 컴퓨팅 슈퍼바이저에서 GUI를 통하여 드래그 앤드 드롭(Drag-and-drop)의 형식으로 쉽게 사물인터

넷 디바이스와 에지 게이트웨이의 연결을 할 수 있는 기능을 제공한다.

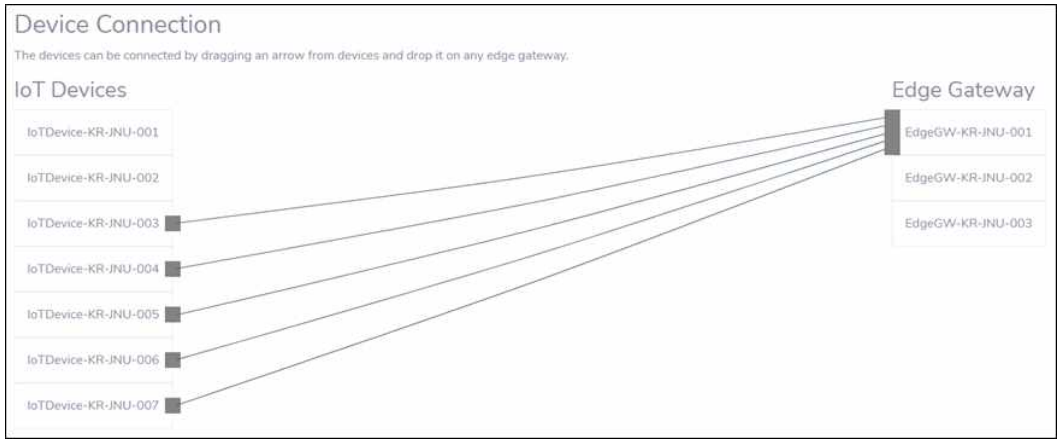


그림 3.29 에지 서버에서 사물인터넷 디바이스와 에지 게이트웨이 연결화면

그림과 같이 왼쪽의 사물인터넷 디바이스로 표현된 객체를 선택하여 마우스로 드래그상태에서 연결하려는 에지 게이트웨이에 드롭하면 연결이 완성된다. 연결을 형성하는 일련의 복잡한 실행과정은 사용자에게 투명하게 처리되고 단지 드래그와 드롭의 작업으로 쉽게 연결 과정을 완성할 수 있다. 연결된 정보는 메인 화면에서 마우스를 사물인터넷 혹은 에지 게이트웨이를 표현하는 카드로 이동하여 확인할 수 있다. 혹은 확인하려는 사물인터넷 디바이스 혹은 에지 게이트웨이 상세페이지로 이동하여서도 확인할 수 있다. 연결을 완성한 후 작업을 생성하여 에지 게이트웨이로 배포하면 작업이 에지 게이트웨이를 통하여 연결된 사물인터넷 디바이스로 데이터를 수집하는 명령을 전달하여 데이터를 수집할 수 있다.

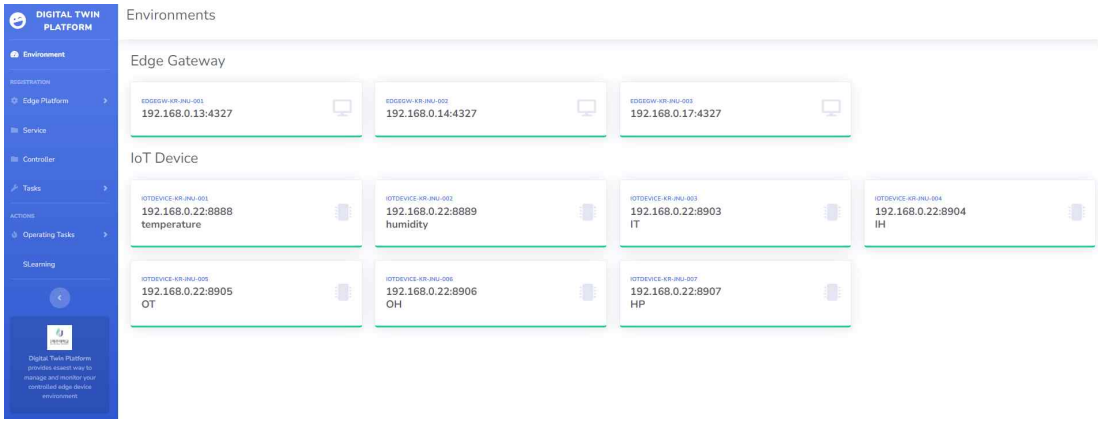


그림 3.30 에지 서버의 분산 디지털 트윈 에지 지능 메인화면

그림 3.30은 디지털 트윈 에지 컴퓨팅 메인 화면이고, 등록된 에지 게이트웨이와 사물인터넷 디바이스의 정보를 확인할 수 있다. 메인 화면에서 에지 컴퓨팅 네트워크에서의 물리 객체를 가상화하여 웹 페이지의 카드 형태로 표현한다. 물리적인 객체의 정보에 근거하여 웹 페이지 상의 카드의 형태로 변환하여 사이버 세계로 미러링한 것이다. 물리적 객체와 카드는 1대1로 대칭하여 표현되고 있다. 메인 화면에서 물리 객체의 실행상태를 실시간으로 확인할 수 있다. 카드 변두리의 색상으로 물리 객체의 실행상태를 표현하고 있다. 초록색은 객체가 정상적으로 작동하고 있음을 나타내고 반대로 파란색으로 표현된 것은 실행되고 있지 않음을 표현하고 있다. 카드에 나타내고 있는 텍스트를 통하여 객체의 이름과 주소를 확인할 수 있으며, 텍스트 옆의 작은 아이콘으로 사물인터넷 디바이스와 게이트웨이를 구분할 수 있도록 에지 게이트웨이는 컴퓨터 모양의 아이콘으로 표현하고, 사물인터넷 디바이스는 프로세서와 비슷한 아이콘으로 표현한다. 그리고 사물인터넷 디바이스에서는 제공하는 리소스의 정보까지도 텍스트로 표현한다. 그림에서는 3대의 에지 게이트웨이와 7대의 사물인터넷 디바이스 등록이 되었고 정상적으로 실행하고 있다는 것을 확인할 수 있다.

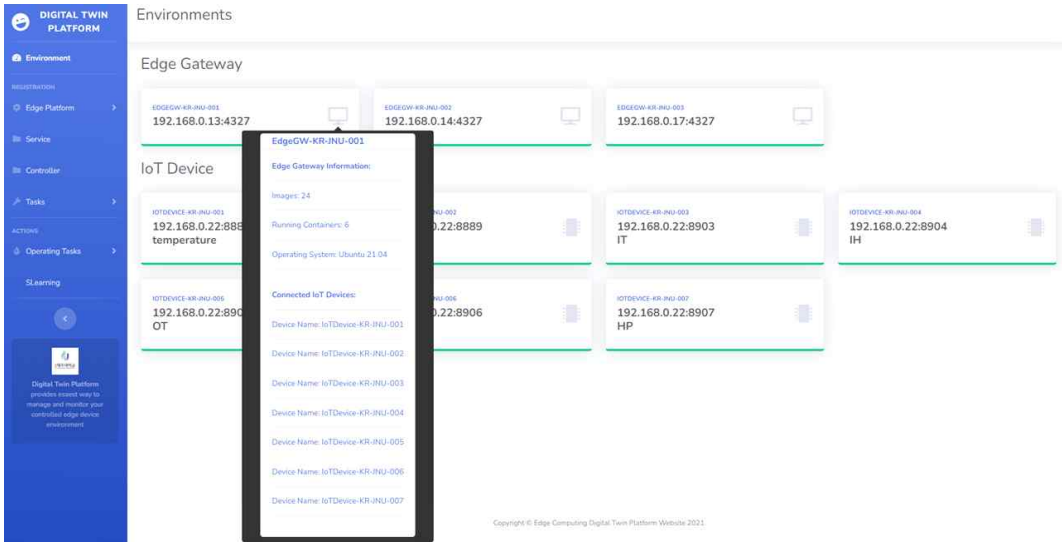


그림 3.31 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-001) 가시화 화면

마우스를 미러링된 객체 카드의 아이콘에 놓으면 해당 객체의 기본정보를 팝 오

버 (pop over) 기능을 통하여 나타낸다. 그림 3.31와 같이 에지 게이트웨이에 놓였을 때 해당 에지 게이트웨이에서 실행하고 있는 운영 체제와 도커 컨테이너 개수를 확인할 수 있을 뿐만 아니라 연결된 사물인터넷 디바이스 정보도 리스트로 표현한다. 리스트에서 확인할 수 있듯이 에지 게이트웨이 EdgeGW-KR-JNU-001은 사물인터넷 디바이스 IoTDevice-KR-JNU-001에서 IoTDevice-KR-JNU-007까지 등록된 모든 사물인터넷 디바이스와 연결되어 있다.

즉 에지 게이트웨이 EdgeGW-KR-JNU-001을 통하여 사용자는 온도, 습도 혹은 스마트 홈 에뮬레이터에서 제공하는 모든 데이터를 수집할 수 있다. 팝 오버 기능을 제공하기 위하여 마우스가 객체에 놓일 때마다 에지 컴퓨팅 슈퍼바이저는 해당 객체의 주소를 통하여 운영 체제, 컨테이너 개수 등의 정보를 요청하고 미리 저장한 객체 관련 정보도 데이터베이스를 통하여 조회한다. 사용자의 편의를 고려하여 마우스의 이동만으로도 객체에 관한 정보를 팝 오버 기능으로 표현하도록 구현한다.

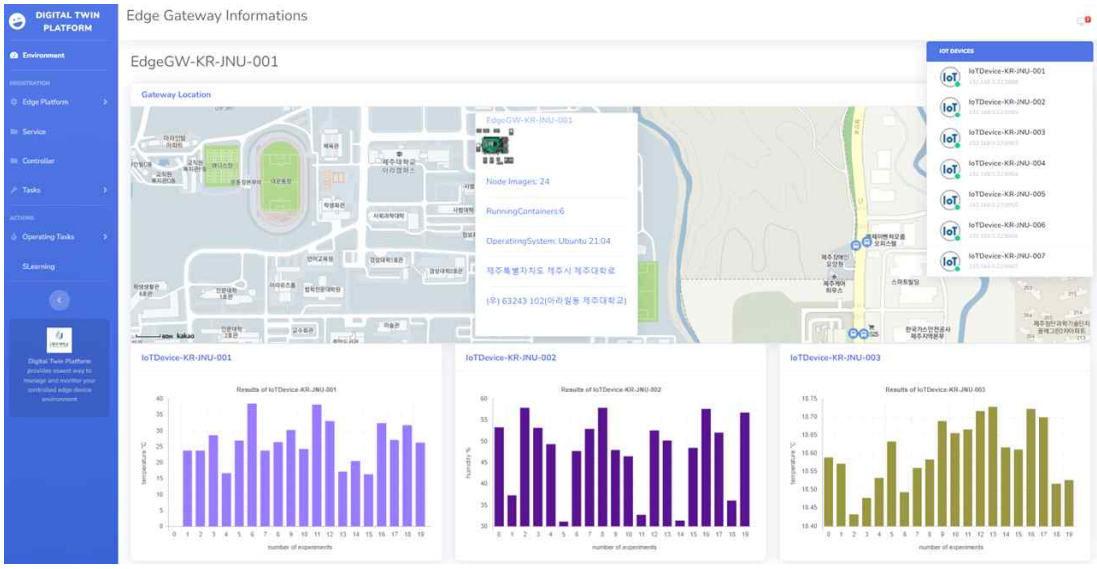


그림 3.32 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-001) 상세 정보화면

미러링 된 객체 카드의 이름을 마우스로 클릭하였을 때 상세 정보를 표현하는 화면으로 변환한다. 그림 3.32은 에지 게이트웨이 EdgeGW-KR-JNU-001의 상세 정보 페이지를 표현한다. 먼저 눈에 크게 보이는 것은 에지 게이트웨이 위치 정보를 이용하여 지도에 그려진 상세 위치 정보이다. 지도에 마크로 에지 게이트웨이가 위치한

장소를 표현할 뿐만 아니라 기본정보도 함께 보여준다. 다음으로 오른쪽 위쪽에 위치한 작은 위젯인데 이는 수치로 에지 게이트웨이와 연결한 사물인터넷 디바이스의 개수를 표현하고 있다. 마우스로 클릭하였을 때 사물인터넷 디바이스 IoTDevice-KR-JNU-001에서 IoTDevice-KR-JNU-007의 이름들이 나열되고 있다. 각각의 사물인터넷 디바이스는 상세 정보 페이지로 이동할 수 있도록 링크로 되어있다. 이어서 도표로 가시화한 데이터 정보를 확인할 수 있다. 그것은 연결된 사물인터넷 디바이스로부터 수집한 데이터에 관한 정보이다. 한 페이지의 화면에 표시할 수 있는 정보가 제한되어 3개의 도표만 표현되고 있다.

그림 3.33은 에지 게이트웨이 EdgeGW-KR-JNU-002에 마우스를 놓았을 때 표현된 팝 오버 정보를 보여주고 있다. 그림에서 알 수 있듯이 에지 게이트웨이 EdgeGW-KR-JNU-002는 사물인터넷 디바이스 IoTDevice-KR-JNU-001와 연결되어 있다. 즉 에지 게이트웨이 EdgeGW-KR-JNU-002를 통하여 사용자는 사물인터넷 디바이스 IoTDevice-KR-JNU-001에서 제공하는 온도 데이터를 수집할 수 있다. 에지 게이트웨이 EdgeGW-KR-JNU-002는 우분투 21.04 버전의 운영 체제를 설치하였고 24개의 도커 이미지가 다운로드 되어있다. 그리고 현재 실행되고 있는 컨테이너는 6개이다. 에지 게이트웨이 EdgeGW-KR-JNU-002를 대표하는 카드의 색상이 초록색이므로 정상적으로 실행이 잘되고 있다는 것을 알 수 있다.

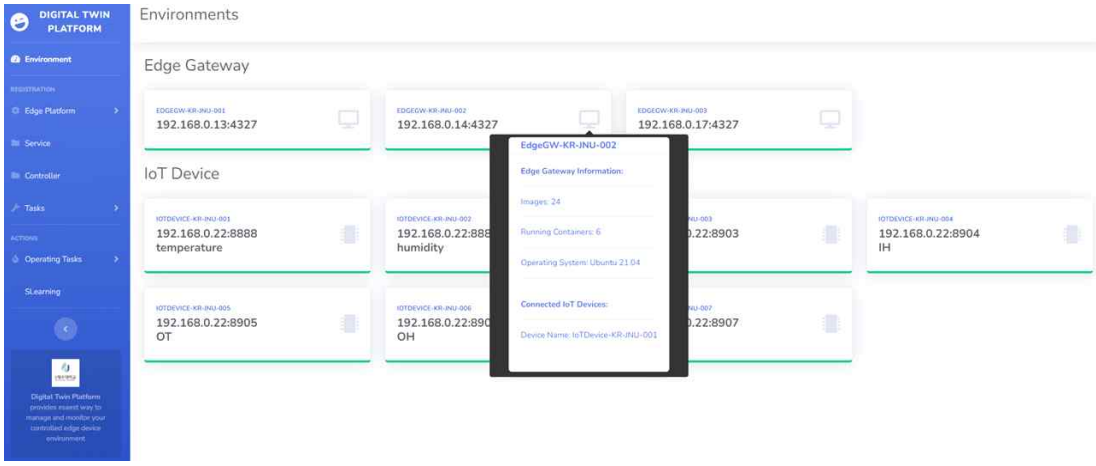


그림 3.33 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-002) 가시화 화면

그림 3.34는 에지 게이트웨이 EdgeGW-KR-JNU-002의 상세 정보 페이지를 보여주고 있다. 에지 게이트웨이 위치 정보를 이용하여 지도에 그려진 상세 위치 정보에서 확인할 수 있듯이 에지 게이트웨이 EdgeGW-KR-JNU-002는 제주대학교에서 실행하고 있다. 오른쪽 위쪽의 위젯을 통하여 연결된 사물인터넷 디바이스의 개수가 하나 인 것을 확인할 수 있고, 사물인터넷 디바이스 IoTDevice-KR-JNU-001와 연결되어 있다는 것도 확인할 수 있다. 아래 도표에서 사용자가 에지 게이트웨이 EdgeGW-KR-JNU-002가 사물인터넷 디바이스 IoTDevice-KR-JNU-001로부터 데이터를 수집했던 회수와 구체적인 데이터 값을 확인할 수 있다. 도표에서 확인할 수 있듯이 19번 데이터를 수집하였고 데이터는 15에서 40 미만의 사이의 값들이다.

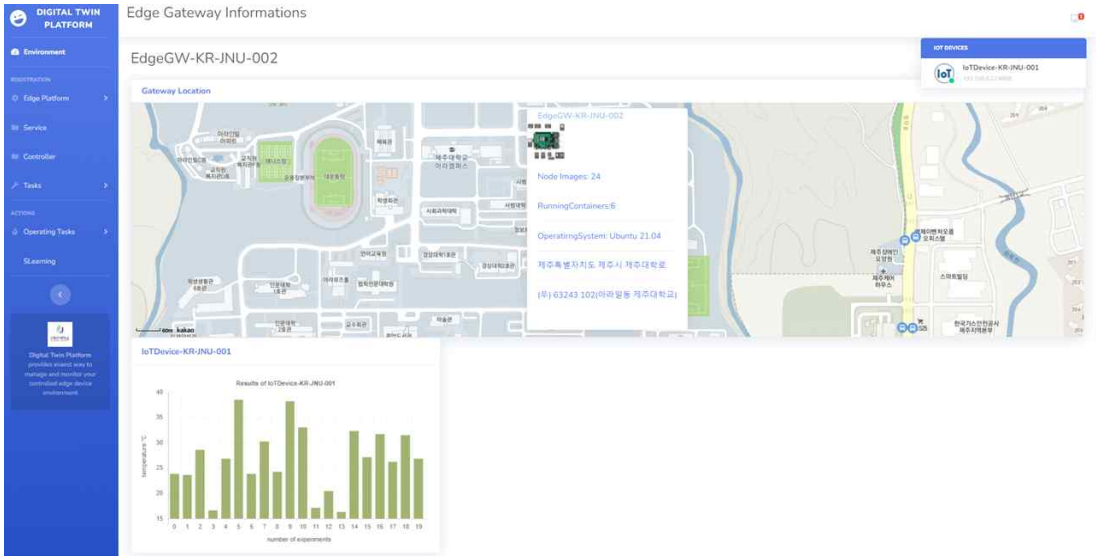


그림 3.34 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-002) 상세 정보화면

그림 3.35은 에지 게이트웨이 EdgeGW-KR-JNU-003에 마우스를 놓았을 때 표현된 팝 오버 정보를 보여주고 있다. 그림과 같이 연결된 사물인터넷 디바이스가 없다. 에지 게이트웨이 EdgeGW-KR-JNU-003는 우분투 21.04 버전의 운영 체제를 설치하였고 12개의 도커 이미지가 다운로드 되어있다. 그리고 현재 실행되고 있는 컨테이너는 6개이다. 에지 게이트웨이 EdgeGW-KR-JNU-003을 대표하는 카드의 색상이 초록색을 나타내고 있어서 정상적으로 실행되고 있는 상태인 것을 알 수 있다.

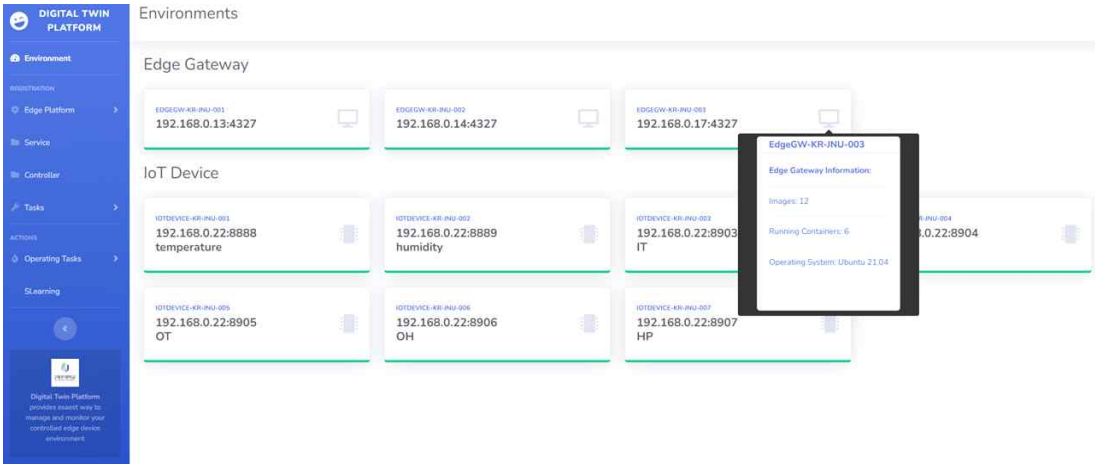


그림 3.35 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-003) 가시화 화면

그림 3.36은 에지 게이트웨이 EdgeGW-KR-JNU-003의 상세 정보 페이지를 표현한다. 연결된 사물인터넷이 없기 때문에 에지 게이트웨이에 대한 상세 정보 및 지도에서의 위치 정보만 확인할 수 있다. 에지 게이트웨이 EdgeGW-KR-JNU-003도 제주대학교에서 실행하고 있다. 에지 게이트웨이 상세 정보 페이지는 연결된 사물인터넷 디바이스와 에지 게이트웨이의 정보에 근거하여 능동적으로 화면 구성을 한다. 에지 게이트웨이 EdgeGW-KR-JNU-003은 연결된 사물인터넷 디바이스가 없기 때문에 오른쪽 위쪽의 위젯이 나타나지 않을 뿐만 아니라 도표 정보도 나타나지 않는다.

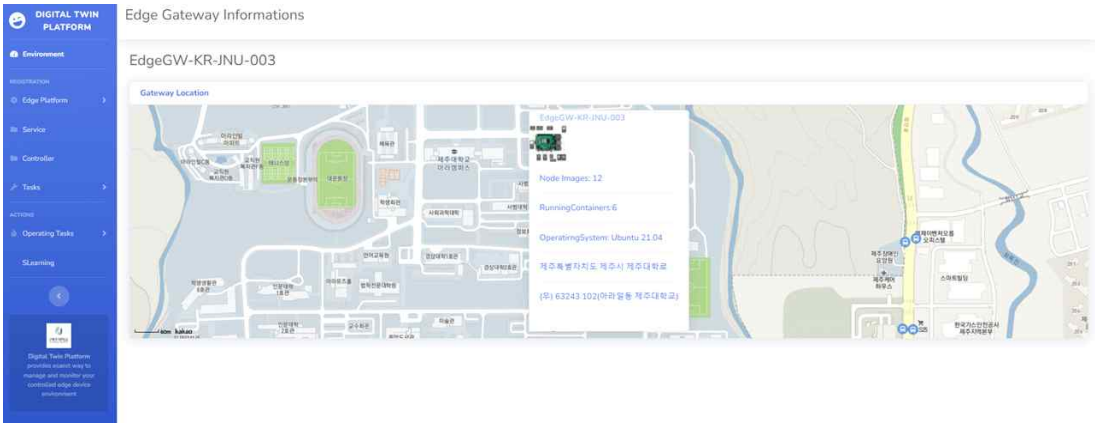


그림 3.36 에지 서버에서 가상 에지 게이트웨이 (EdgeGW-KR-JNU-003) 상세 정보화면

그림 3.37은 사물인터넷 디바이스 IoTDevice-KR-JNU-001에 마우스를 놓았을 때 표현된 팝 오버 정보를 보여주고 있다. 사물인터넷의 이름과 자원 정보를 확인할 수 있고, 연결된 에지 컴퓨팅 게이트웨이 정보를 확인할 수 있다. 사물인터넷 디바이스 IoTDevice-KR-JNU-001은 온도 데이터를 제공하고 있다는 것을 알 수 있다. 연결된 에지 게이트웨이는 EdgeGW-KR-JNU-001와 EdgeGW-KR-JNU-002이다. 즉 에지 게이트웨이 EdgeGW-KR-JNU-001와 EdgeGW-KR-JNU-002를 통하여 사물인터넷 디바이스 IoTDevice-KR-JNU-001에 접근할 수 있다. 사물인터넷 디바이스 IoTDevice-KR-JNU-001를 대표하는 카드의 색상이 초록색을 나타내고 있다. 디바이스가 정상적으로 실행되고 있는 상태인 것을 알 수 있다.

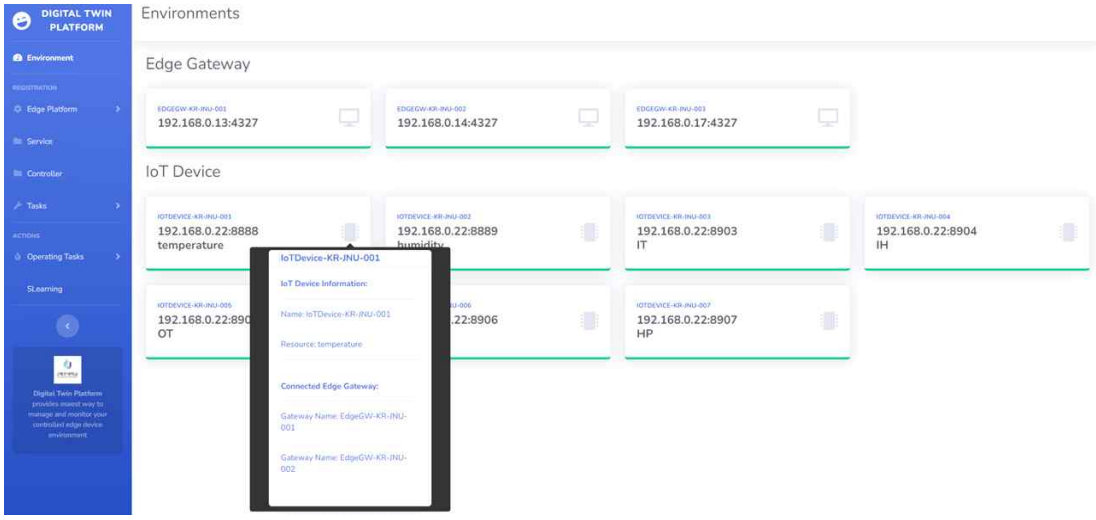


그림 3.37 에지 서버에서 가상 사물인터넷 디바이스 (IoTDevice-KR-JNU-001) 가시화 화면

그림 3.38은 사물인터넷 디바이스 IoTDevice-KR-JNU-001의 상세 정보 페이지를 보여주고 있다. 디지털 트윈 에지 컴퓨팅 메인 화면에서 사물인터넷 디바이스 IoTDevice-KR-JNU-001의 이름을 클릭하였을 때 나타나는 상세 정보 페이지이다. 오른쪽 위쪽의 위젯에서 연결된 에지 게이트웨이의 링크가 나열되어 있다. 나열된 링크를 통하여 연결된 에지 게이트웨이의 이름을 알 수 있다. 그리고 각각의 에지 게이트웨이에서 수집한 데이터의 정보를 도표로 화면 아래쪽에 위치한다. 첫 번째 도표를 보았을 때 에지 게이트웨이 EdgeGW-KR-JNU-001에서 데이터를 수집하였고 온도 데이터를 수집하였으며, 데이터는 0에서 40사이의 값이며 19번의 데이터 수집 결

과를 나타내고 있는 것을 알 수 있다.

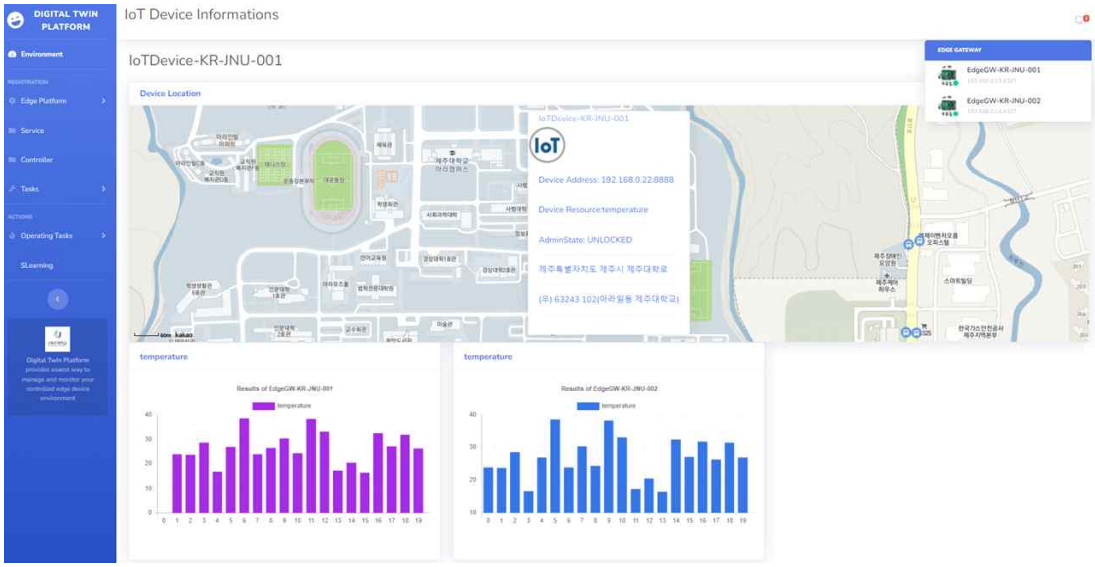


그림 3.38 에지 서버에서 가상 사물인터넷 디바이스 (IoTDevice-KR-JNU-001) 상세 정보화면

그림 3.39는 사물인터넷 디바이스 IoTDevice-KR-JNU-002에 마우스를 놓았을 때 표현된 팝 오버 정보를 보여주고 있다.

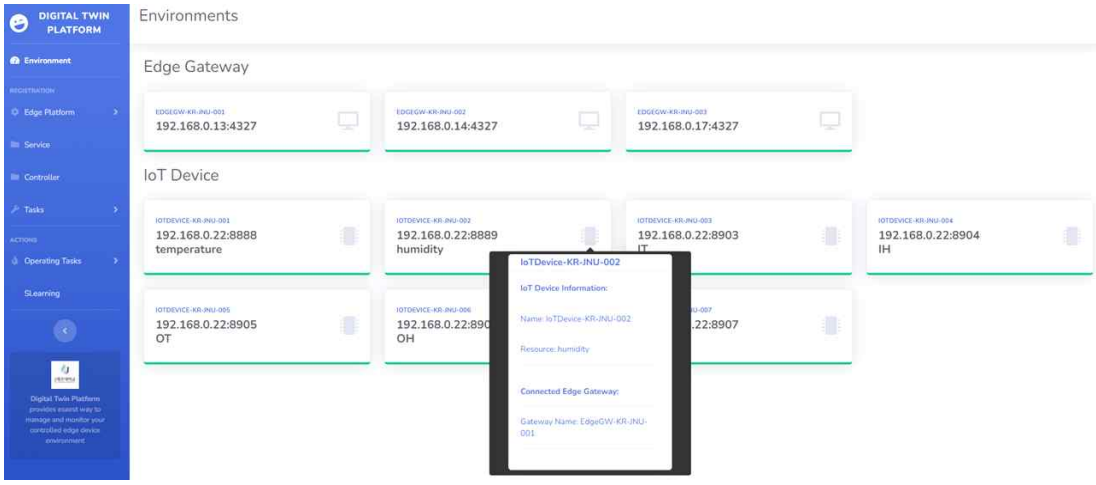


그림 3.39 에지 서버에서 가상 사물인터넷 디바이스 (IoTDevice-KR-JNU-002) 가시화 화면

사물인터넷 디바이스 IoTDevice-KR-JNU-002는 습도 데이터를 제공하고 있다는

것을 알 수 있다. 연결된 에지 게이트웨이는 EdgeGW-KR-JNU-001이다. 즉 에지 게이트웨이 EdgeGW-KR-JNU-001을 통하여 사물인터넷 디바이스 IoTDevice-KR-JNU-002에 접근할 수 있다. 사물인터넷 디바이스 IoTDevice-KR-JNU-002를 대표하는 카드의 색상이 초록색을 나타내고 있으므로 디바이스가 정상적으로 실행되고 있다.

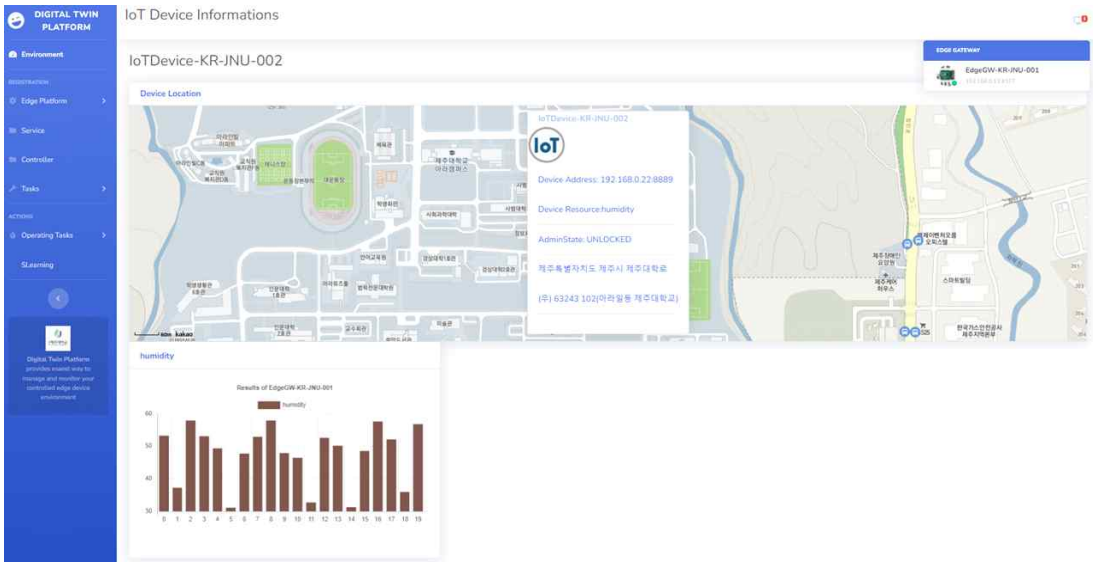


그림 3.40 에지 서버에서 가상 사물인터넷 디바이스 (IoTDevice-KR-JNU-002) 상세 정보화면

그림 3.40은 사물인터넷 디바이스 IoTDevice-KR-JNU-002의 상세 정보 페이지를 보여주고 있다. 디지털 트윈 에지 컴퓨팅 메인 화면에서 사물인터넷 디바이스 IoTDevice-KR-JNU-002의 이름을 클릭하였을 때 나타나는 상세 정보 페이지이다. 오른쪽 위쪽의 위젯에서 연결된 에지 게이트웨이 EdgeGW-KR-JNU-001으로 페이지를 전환할 수 있는 링크를 확인할 수 있다. 그리고 지도 아래에는 에지 게이트웨이에서 수집한 데이터의 정보를 도표로 표현한다. 도표를 보았을 때 에지 게이트웨이 EdgeGW-KR-JNU-001에서 데이터를 수집하였고 습도 데이터를 수집하였으며 데이터는 30에서 60사이의 값이며, 19번의 데이터 수집 결과를 나타내고 있는 것을 알 수 있다.

분산 디지털 트윈 에지 컴퓨팅환경에서 서비스와 작업을 생성하는 과정은 그림 3.41과 같다. 사용자는 ECS의 디지털 트윈 오브젝트 생성자를 통하여 작업과 서비스

를 데이터베이스로 저장한다. 작업은 이름과 실행할 에지 게이트웨이 그리고 사물인터넷 디바이스 정보를 입력하여 생성한다. 서비스는 이름과 실행하고 있는 네트워크 주소 그리고 필요로 하는 사물인터넷 자원을 지정하여 생성한다. 생성한 작업은 ECS를 통하여 에지 게이트웨이로 배포하여 실행한다. 작업이 실행하면서 수집된 데이터는 다시 데이터베이스로 저장된다.

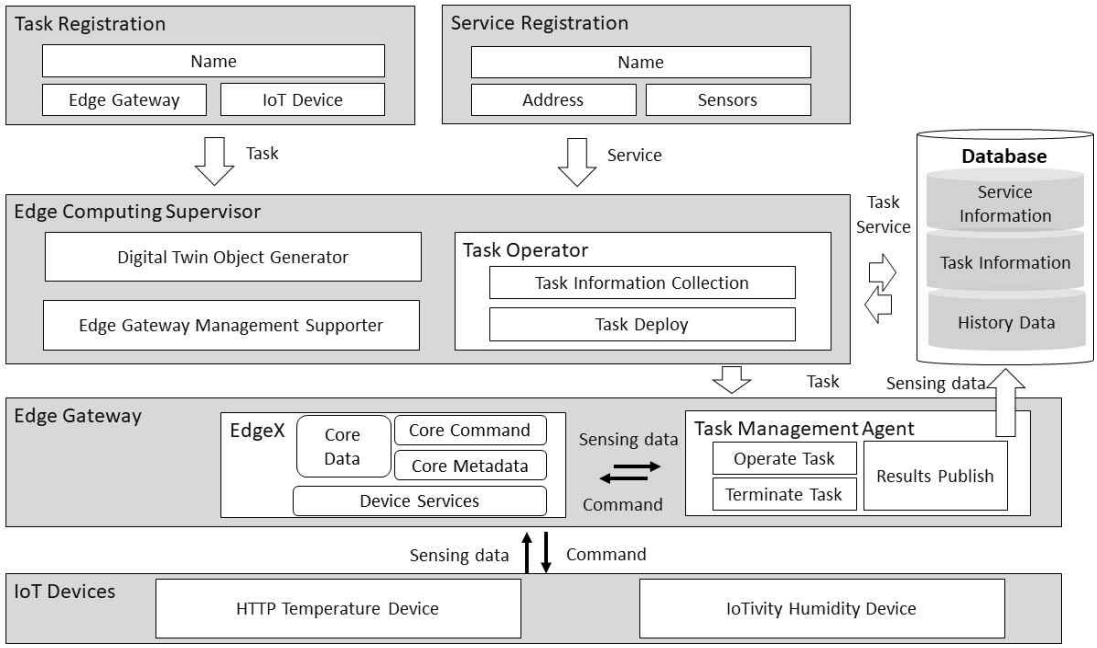


그림 3.41 에지 서버 (Edge Computing Supervisor) 서비스 및 작업 생성 과정

서비스 등록화면은 그림 3.42와 같다. 서비스는 사물인터넷 디바이스로부터 수집된 데이터를 이용하여 최적화 기법을 제공하는 객체를 가리킨다. 서비스 관련 이름, 주소, 그리고 센서 (Sensors)들은 입력 자원 정보를 가리킨다. 등록 버튼을 클릭함으로써 등록 작업을 완성한다. 이름은 서비스를 구분하는 식별자로 사용되기 때문에 서로 다른 이름을 사용한다. 주소는 또한 인터넷에서 단말을 식별할 수 있는 IP (Internet of protocol)주소와 서비스가 사용하는 포트번호가 포함되어야 한다. 센서들은 서비스가 필요로 하는 데이터들이다. 마찬가지로 문제없이 등록이 완성되면 성공적으로 등록되고 있다는 문구가 위쪽에 표현이 된다.

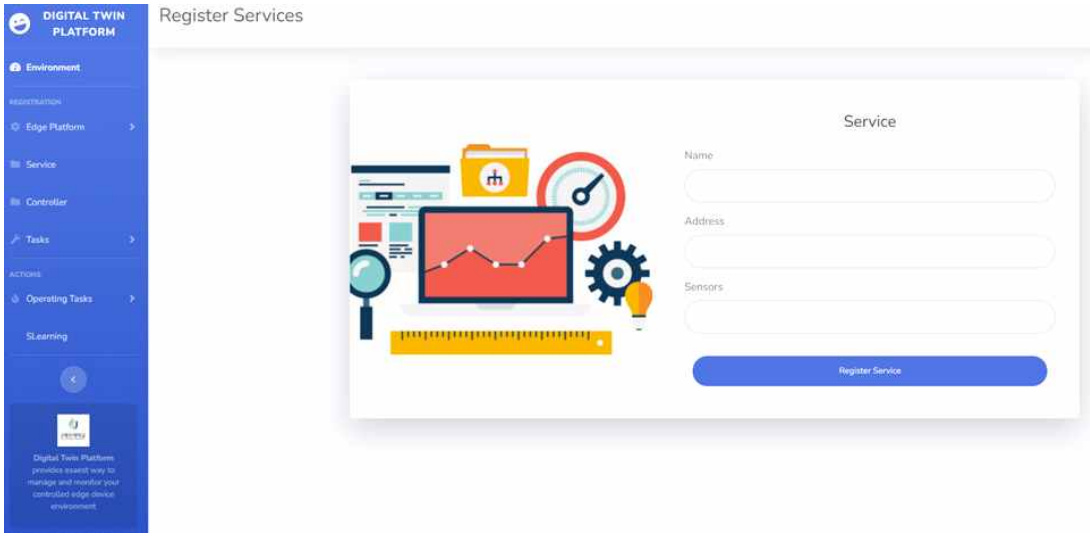


그림 3.42 PMV 예측 서비스 등록화면

작업 생성화면은 그림 3.43과 같다. 작업은 사물인터넷 디바이스에서 데이터를 수집하거나 수집한 데이터를 서비스에 적용하여 최적화 결과를 생성하는 등의 프로세스이다.

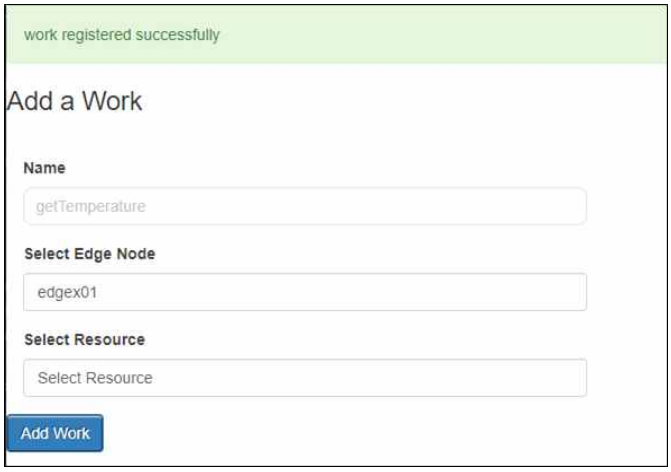


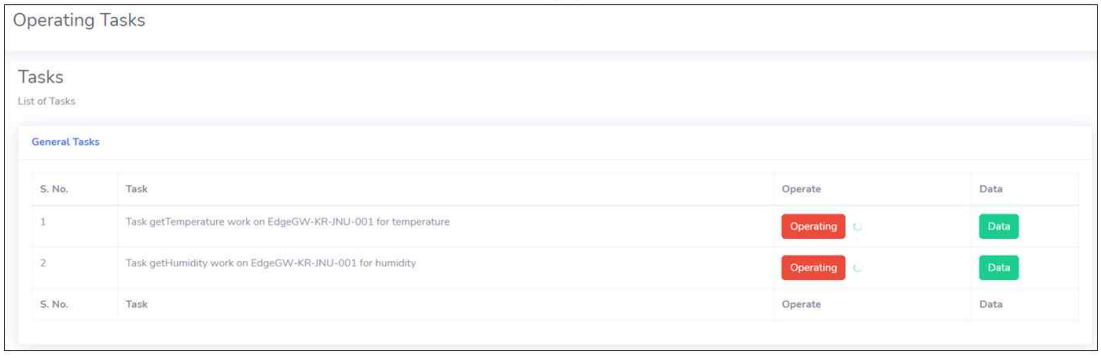
그림 3.43 온도 데이터 수집 작업 등록화면

사용자는 작업 관련 이름을 정의할 수 있으나 에지 게이트웨이와 자원 정보는 위에서 등록한 정보에 근거하여 선택할 수 있다. 이름은 작업을 구분하는 식별자로 사용되고 에지 게이트웨이와 자원은 각각 사전에 등록한 에지 게이트웨이와 해당 게이트웨이와 연결을 생성한 사물인터넷 디바이스 정보를 기반으로 나열된다. 선택

된 게이트웨이 정보가 변화할 때마다 데이터베이스를 질의하여 디바이스 정보도 따라서 변화한다. 에지 게이트웨이는 생성한 작업이 실행하는 위치를 가리킨다. 정상적으로 등록이 되면 그림과 같이 성공 문구가 위쪽에 표시된다.

작업 실행화면은 그림 3.44와 같다. 테이블로 작업에 관한 정보를 작업 (task), 실행 (operate), 데이터 컬럼으로 구분하여 표현한다. 작업 컬럼은 작업명에 관한 정보와 실행하는 에지 게이트웨이 이름 그리고 수집하는 데이터로 되어있다. 실행 컬럼은 버튼으로서 클릭하여 작업을 실행하고 종료할 수 있다. 그림 b는 작업의 실행상태를 나타내고 있다. 작업이 실행되면 버튼이 빨간색으로 변하고 값이 “operatirng” 으로 변한다. 작업이 실행되지 않을 때는 그림 a와 같이 파란색으로 변하고 값은 Operate로 변한다. 마지막으로 데이터 컬럼은 데이터 가시화 화면으로 이동하는 링크 버튼이다. 데이터를 클릭하면 해당 작업이 실행하는 동안 수집한 데이터를 조회하여 도표로 표현한다.

(a) 작업 대기 상태



(b) 작업 실행 상태

그림 3.44 온도/습도 데이터 수집 작업 실행화면

그림 3.45는 사물인터넷 디바이스로부터 온도 정보를 수집한 데이터를 시각화한

결과이다. 아래 데이터는 에지 게이트웨이 EdgeGW-KR-JNU-001을 통하여 사물인터넷 디바이스 IoTDevice-KR-JNU-001로부터 온도 데이터를 수집한 것을 시각화한 것이다.

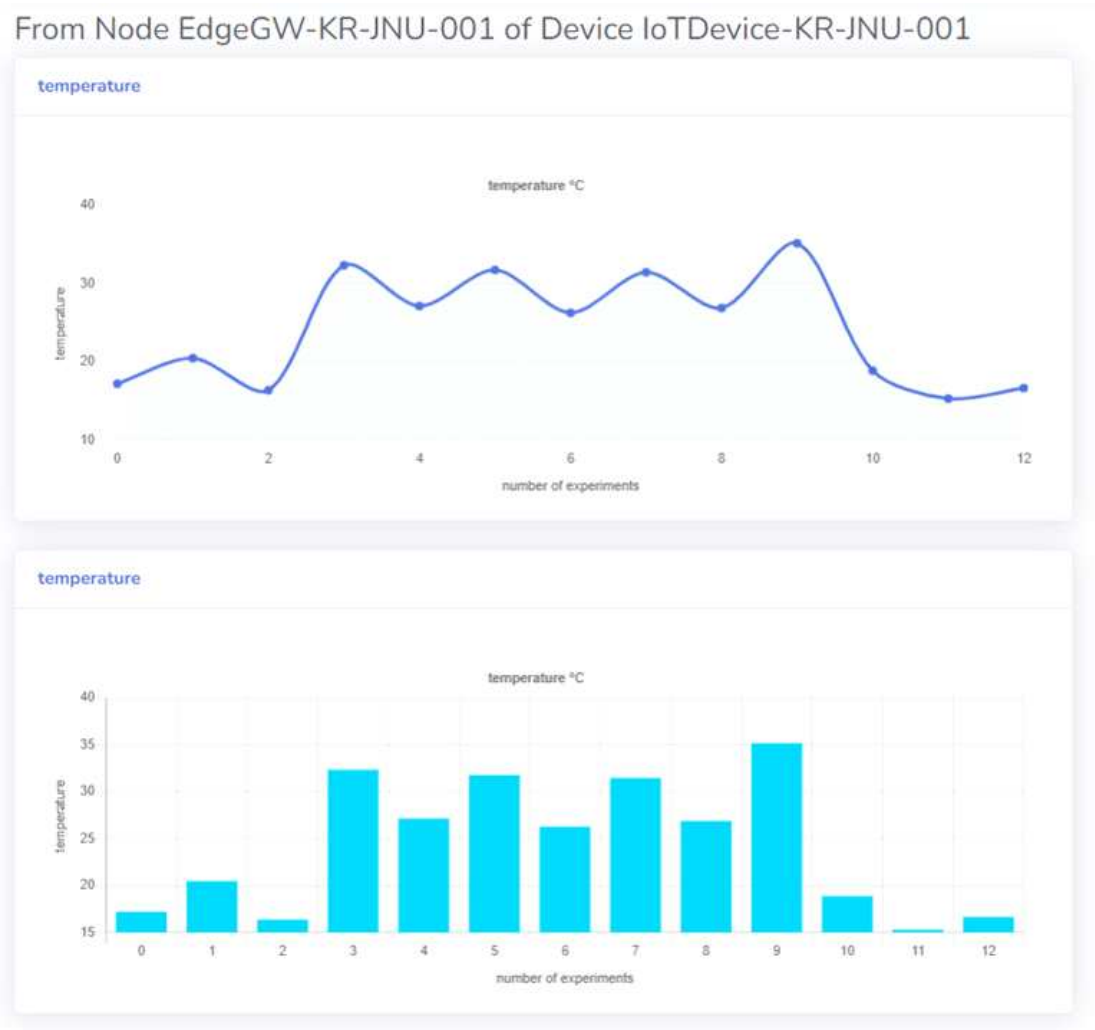


그림 3.45 OCF IoTivity 기반 온도 데이터 수집 작업 결과 가시화 화면

왼쪽 위에는 에지 게이트웨이와 사물인터넷 디바이스의 이름이 나타난다. 데이터는 선형, 그리고 막대기 도표 형태로 표현되고 있다. X좌표는 회 차 정보를 나타내고 Y좌표는 데이터 값 정보를 표현하고 있다. 막대기 좌표에서는 자원의 정보도 함께 표현되고 있다. 온도 데이터는 10에서 40사이의 값을 나타내고 있다. 12회의 데이터 수집 결과를 보여주고 있다. 아래 데이터는 IoTivity 프레임워크를 통하여 수집된 데이터 정보이다. 사물인터넷 디바이스 IoTDevice-KR-JNU-001은 IoTivity 서버를

실행하고 있고 온도 데이터를 자원으로 등록하여 있다. 사물인터넷 디바이스 IoTDevice-KR-JNU-001으로 온도 데이터를 요청하면은 랜덤 온도 값을 반환한다. 온도 데이터의 단위는 섭씨도 “° C”로 되어있다.

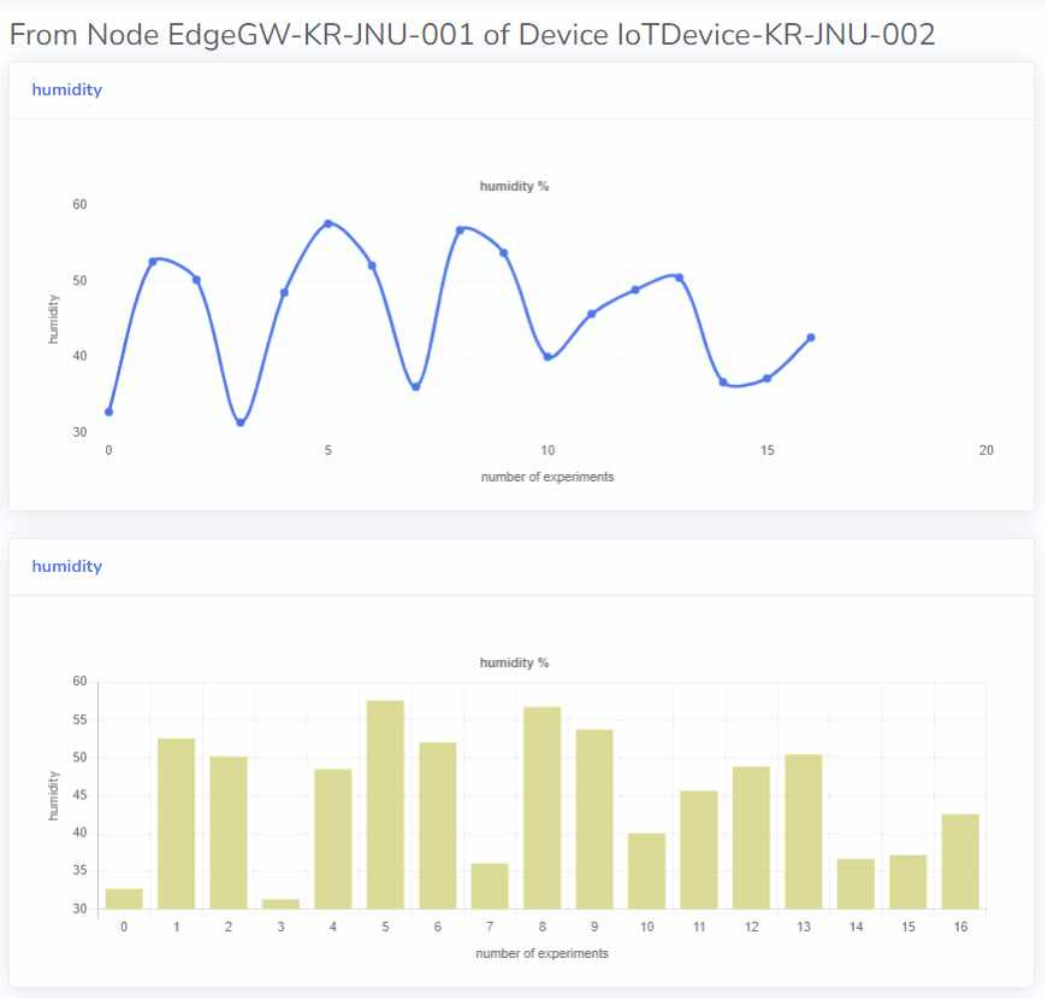


그림 3.46 HTTP 기반 습도 데이터 수집 작업 결과 가시화 화면

그림 3.46은 사물인터넷 디바이스로부터 습도 정보를 수집한 데이터를 시각화한 결과이다. 사물인터넷 디바이스 IoTDevice-KR-JNU-002는 spring boot 프레임워크를 이용하여 구현한 서버이다. 디바이스는 랜덤의 습도 값을 HTTP 프로토콜을 통하여 반환한다. 아래 데이터는 에지 게이트웨이 EdgeGW-KR-JNU-001을 통하여 사물인터넷 디바이스 IoTDevice-KR-JNU-002로부터 습도 데이터를 수집한 것을 시각화한 것이다. 왼쪽의 도표는 데이터의 값과 수집 회수를 선형으로 표현하였고 오른쪽 도표

에서는 막대기 형식으로 표현한다. 같은 데이터를 서로 다른 표현형식으로 나타냈다. 도표에서 알 수 있듯이 데이터는 30에서 60사이의 값이고 총 16번의 데이터 수집 회수의 결과를 표현한다. 습도 데이터의 단위는 “%”로 되고 있다.

6. EdgeX와 OCF IoTivity 기반 디지털 트윈 에지 컴퓨팅 성능 분석

그림 3.47는 IoTivity와 HTTP를 통하여 구현한 사물인터넷 디바이스와 통신을 하여 데이터를 수집하는 작업 시간을 측정한 결과이다. 시간은 밀리 세컨드를 단위로 측정한다. 막대기 도표를 보았을 때 왼쪽은 IoTivity 프레임워크의 시간이고 오른쪽은 HTTP 프로토콜의 시간임을 알 수 있다.

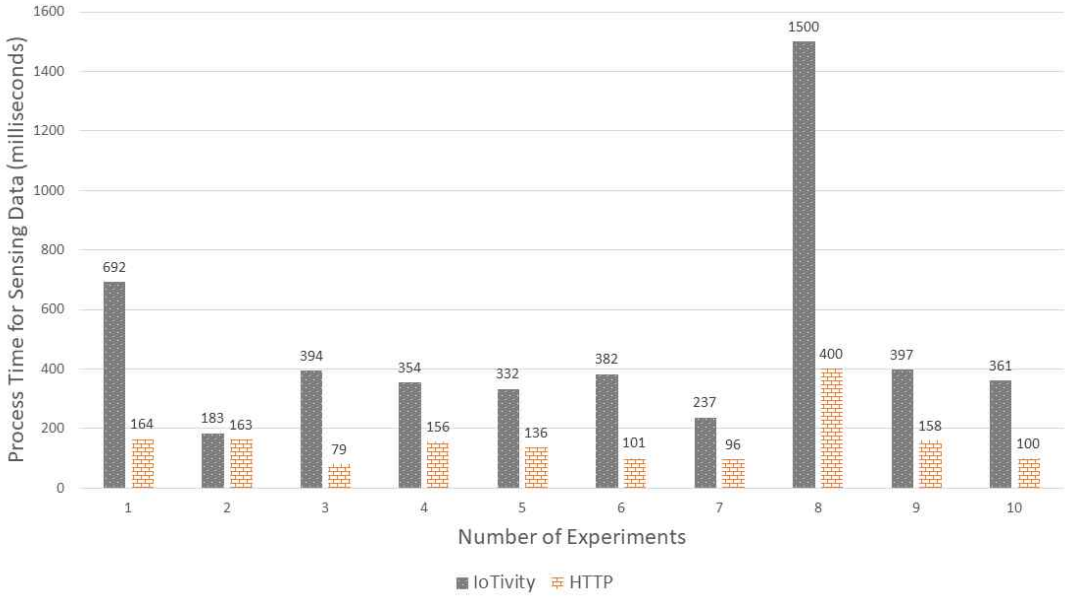


그림 3.47 OCF IoTivity와 HTTP 기반 사물인터넷 디바이스와 에지 게이트웨이 간의 데이터 수집 작업 소요 시간 비교

총 10회의 테스트 결과를 보여주었고 IoTivity 프레임워크가 최고 값 1500을 빼고는 모두 1초 미만인 것을 알 수 있다. HTTP 프로토콜을 구현한 사물인터넷 디바이스로부터 데이터를 수집하는 데 사용하는 시간보다 IoTivity 프레임워크를 구현한 사물인터넷 디바이스로부터 데이터를 수집하는 시간이 더욱 많이 필요하다. IoTivity

프레임워크는 클라이언트와 서버로 구성되어있다. 서버는 클라이언트에서 참조할 수 있는 자원을 등록하여 클라이언트가 참조하기를 기다린다. 클라이언트는 서버의 네트워크 주소를 모르는 상태에서 네트워크에 등록되어있는 자원의 식별자로부터 서버를 찾고 접근하여 데이터를 반환받는다.

그림 3.48은 위의 결과를 통계한 도표이다. 데이터 수집 작업 신간을 최저 값 (min), 최대 값 (max), 평균 값 (avg) 그리고 표준 편차 (std)로 통계를 한다. 통계 결과를 보았을 때 HTTP 프로토콜을 사용하였을 때 작업 시간은 평균이 155.3 ms이고 최고 값이 400 ms로 0.5초 미만의 시간을 사용하는 것을 알 수 있다. IoTivity 프레임워크를 구현한 디바이스는 최고 값이 1500이지만 평균이 483.2로 대부분의 상황에서 0.5초 미만의 시간을 사용한다. 마지막으로 표준편차를 보았을 때 IoTivity 프레임워크 보다 HTTP가 보다 안정적인 작업 시간을 사용하는 것을 알 수 있다.

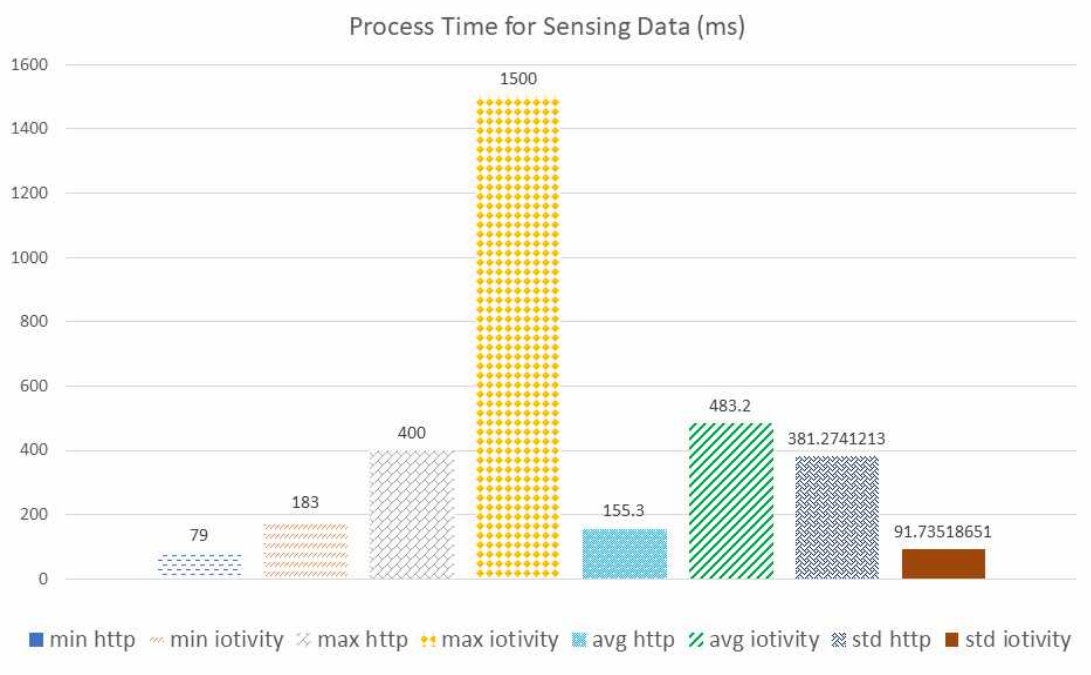


그림 3.48 OCF IoTivity와 HTTP 기반 사물인터넷 디바이스와 에지 게이트웨이 간의 데이터 수집 작업 소요 시간 통계

IV. PMV 예측을 위한 개선된 스웜학습 전략

1. 심층 신경망 선형 회귀 기반 개선된 스웜학습 전략

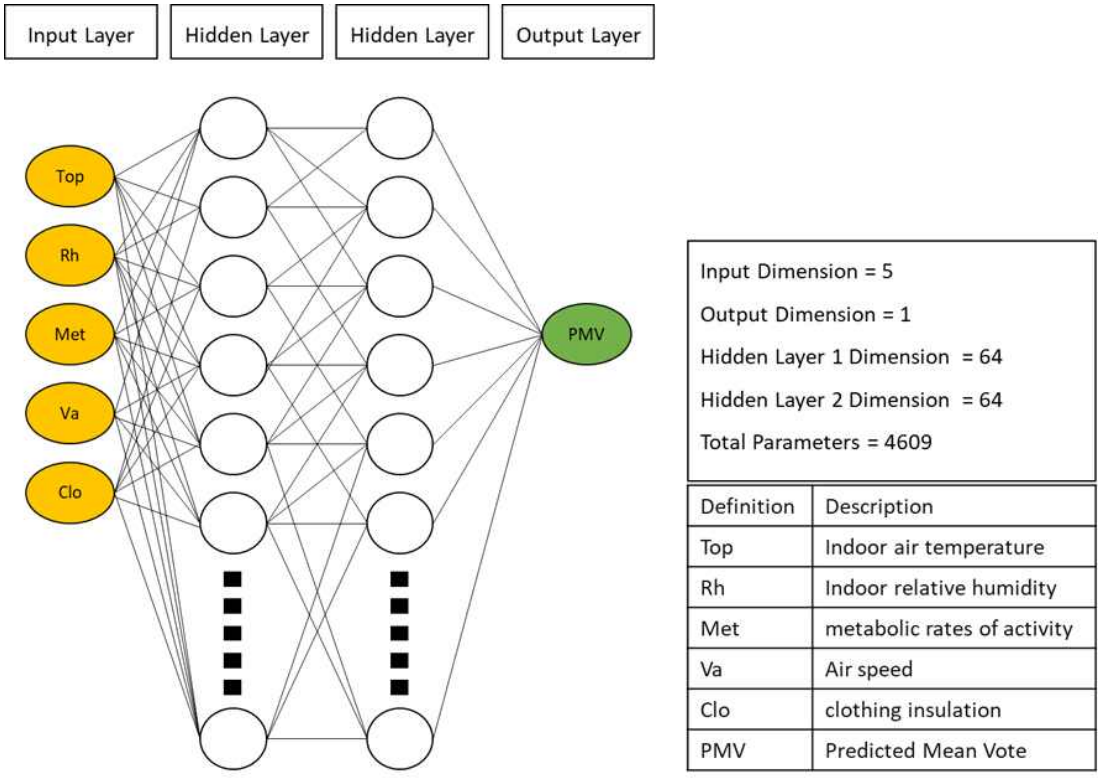
사람들은 편안함을 느끼는 열 환경에서 살기를 원한다. 날씨나 계절의 변화로 인해 기온이 떨어지면 사람들은 따뜻한 환경에서 생활하고 싶어하고, 기온이 오르면 더 시원한 환경에 머물고 싶어한다. 환경은 사람들의 작업효율, 기분, 심지어 건강상태에도 영향을 준다. 거주자의 열 쾌적도를 반영하는 PMV (Predicted Mean Vote)는 1970년에 개발되었으며 ASHRAE [101]의 표준 55로 채택되고 있다. 인간의 열적 안락감과 물리적 환경 변수 간의 관계를 표현하고 환기 시스템을 설계하고 허용 가능한 실내 공기 품질을 평가하는 표준이다. 열 쾌적도의 레벨은 -3에서 3사이로 춥다 (cool), 차갑다 (cold), 약간 차갑다 (slightly cold), 일반 (neutral), 약간 따뜻함 (slightly warm), 따뜻함 (warm), 그리고 더움 (hot)과 같은 거주자의 열 쾌적도를 표현한다. PMV 레벨은 온도 (air temperature), 평균 복사 온도 (mean radiant temperature), 상대 습도 (relative humidity), 풍속 (air velocity), 대사 율 (metabolic rate) 그리고 의류 단열 (clothing insulation) 등과 같이 환경 요인과 여러 중요 요인의 영향을 받는다.

표 4.1 PMV에서 사용하는 데이터 특성

Definition	Description
Top	Indoor air temperature
Rh	Indoor relative humidity
Met	Metabolic rates of activity
Va	Air speed
Clo	Clothing insulation

이 논문에서 우리는 모델을 훈련하기 위해 도표 4.1과 같은 특성을 선택한다. 실내 공기 온도(Top), 상대 습도(Rh), 풍속(Va)과 같은 환경적 특성과 의류 단열(Clo) 및 활동 대사율(Met)과 같은 개인 특성을 포함하고 있다.

그림은 PMV를 예측하는 심층 신경망 회귀 (Deep Neural Network Regression, DNNR) 모델이다. 본고에서 제안한 심층 신경망 회귀모델은 지도 학습 전략으로 수집된 데이터 기반으로 학습을 진행하여 모델을 구축한다. 거주자 열 쾌적도를 예측하는 심층 신경망 회귀모델은 다중 계층을 가진 신경망으로 은닉층은 2개의 계층 64개의 뉴론을 가지고 입력층과 출력층을 포함한다. 모델에 사용되는 입력 파라메터는 본고에서 선택한 실내 공기 온도(Top), 상대 습도(Rh), 풍속(Va), 의류 단열(Clo) 및 활동 대사율(Met)이다. 출력층은 PMV 파라미터값이다. 총 파라미터는 4609개이다.



Definition	Description
Top	Indoor air temperature
Rh	Indoor relative humidity
Met	metabolic rates of activity
Va	Air speed
Clo	clothing insulation
PMV	Predicted Mean Vote

그림 4.1 PMV를 예측하는 심층 신경망 회귀모델

에지 컴퓨팅 환경에서 에지 게이트웨이가 사물인터넷 디바이스와 통신하여 데이터를 수집하고 제어하는 역할을 한다. 민감한 데이터를 로컬 네트워크에서 처리하고 저장함으로써 데이터에 대한 안전성을 보장할 뿐만 아니라 데이터 처리 속도도 향상시킨다. 에지 컴퓨팅의 자원을 충분히 이용하고 데이터의 안전성도 보장하며 데이터의 모든 지식을 학습할 수 있도록 본고에서는 그림 4.2와 같은 방법으로 모델을

훈련했다. 에지 컴퓨팅 네트워크에 참여하는 모든 에지 게이트웨이가 분산 에지 컴퓨팅 네트워크 환경을 구성한다. 네트워크에 있는 모든 데이터를 서로 전달하지 않고 모델을 공유하여 성능이 좋은 모델을 찾는다. 네트워크에 있는 모든 게이트웨이는 실행과 함께 로컬 데이터를 통하여 모델을 훈련한다. 그리고 네트워크에 있는 게이트웨이 중에서 하나의 게이트웨이를 지정하여 모델의 전달을 시작한다. 마지막으로 다시 시작하는 게이트웨이로 모델이 돌아오게 되면 로컬모델과 비교하여 성능이 좋은 모델을 로컬모델로 교체하고 향후의 예측에 사용한다. 선택된 모델은 네트워크에 있는 모든 게이트웨이로 전달한다. 마지막으로 게이트웨이들이 로컬모델과 전달된 모델의 성능 비교를 진행하여 성능이 좋은 모델로 로컬모델을 교체하여 항상 최적의 성능으로 훈련된 모델이 서비스를 제공하도록 한다.

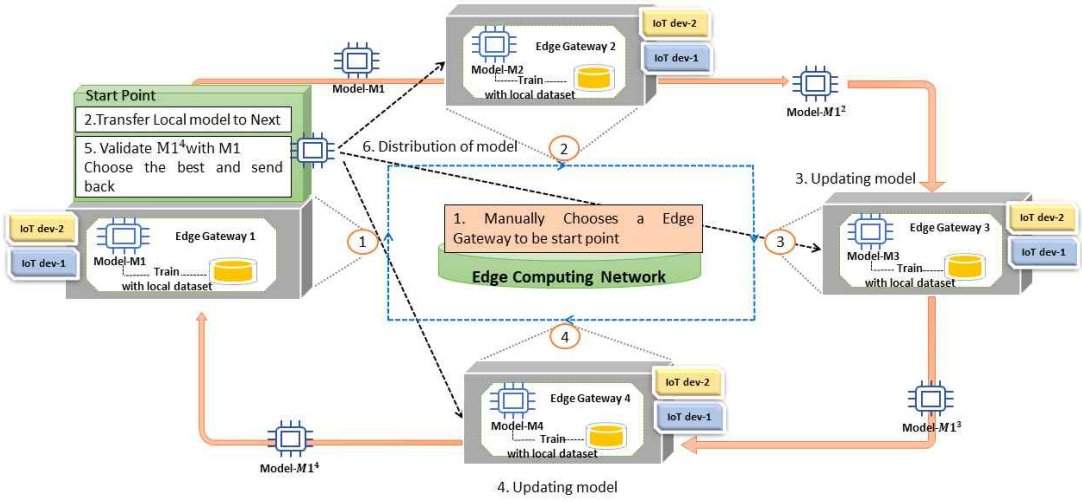


그림 4.2 개선된 스웜학습 모델 훈련 개념도

표 4.2에서 본고에서 개선한 스웜학습 전략과 기존 에지 컴퓨팅에서 모델학습 전략을 비교한다. 기존의 연합 학습과 스웜학습은 중앙 서버에 대한 높은 컴퓨팅 파워를 요구한다. 그리고 중앙서버에서 글로벌 모델을 가중치 평균의 방식으로 업데이트하여 에지 컴퓨팅 네트워크에 있는 데이터의 패턴을 모두 학습하지 못한다.

하지만 본고에서 개선한 학습 전략은 에지 컴퓨팅 네트워크에 있는 다른 에지 노드와 같은 컴퓨팅 파워만 요구하고 순서대로 에지 컴퓨팅 네트워크에 있는 모든 노드의 데이터를 이용하여 모델을 업데이트하여 데이터의 모든 패턴을 학습한다.

그림 4.2 개선한 스웜학습 전략과 기존 연합/스웜 모델학습 전략 비교

	Federated Learning	Swarm Learning	Proposed Swarm Learning
Central Server	A dedicated server	Dynamically selected	Manually choose
Computing Power	High	High	Same as other node on edge computing network
Data	Local edge node	Local edge node	Local edge node
Model training	Parallel	Parallel	Sequential
Updating mechanism	Weight averaging	Weight averaging	SGD

그림 4.3은 에지 게이트웨이의 스웜학습 기능 블록을 나타낸다. 스웜 학습 서버 Swarm Learning Server, SLS)는 에지 게이트웨이에서 TMA와 같이 실행하여 스웜학습을 실행한다.

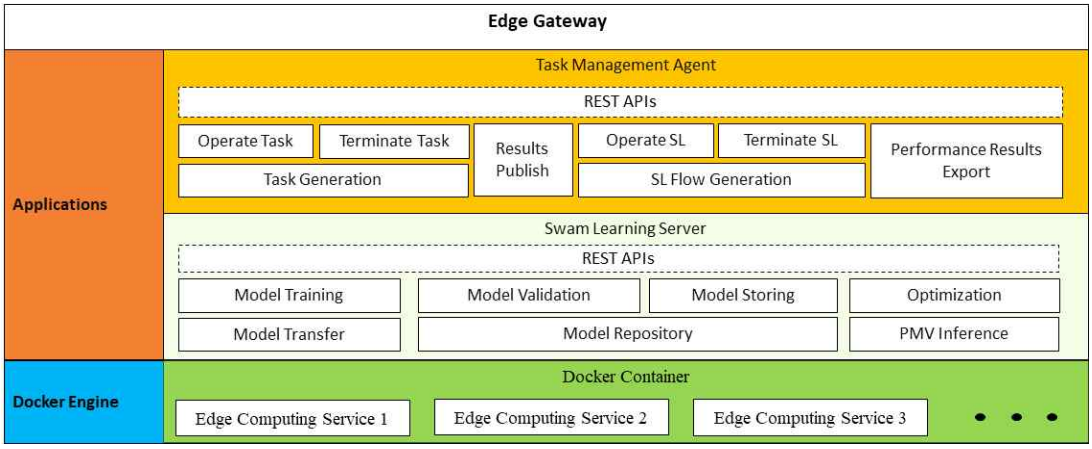


그림 4.3 에지 게이트웨이의 개선된 스웜학습 전략 기능 블록

SLS는 모델을 훈련, 전달, 검증, 저장하는 기능과 모델을 사용하여 최적화와 PMV를 예측하는 기능을 포함하고 있다. 사용자가 ECS로 스웜학습을 실행하도록 요청을 전달하면 ECS는 데이터베이스로부터 에지 게이트웨이들의 정보를 정리하여 리더로 선택된 게이트웨이의 TMA로 전달한다. TMA는 전달된 정보를 근거로 모델 전달의 순서를 정한다. 정해진 순서에 따라 SLS에 스웜학습을 실행하도록 한다. SLS는 모델 전달기능을 통하여 정해진 다음의 에지 게이트웨이로 모델을 전달한다. 모델이 에지 컴퓨팅 네트워크를 돌고 다시 시작점으로 돌아오면 검증 기능을 통하여 성능이 좋은 모델을 선택한다. 선택된 모델은 다시 다른 게이트웨이로 전달하여 다른 게이트

웨이도 성능 좋은 모델로 교체하도록 한다.

2. 기존 연합/스웜 학습과 개선된 스웜학습을 이용한 PMV 예측 메커니즘

PMV 모델을 학습하는 데이터는 그림 4.4와 같이 달별로 수집된 데이터이다. 1월부터 12월까지의 데이터가 수집되고 있다. 총 4년의 데이터를 선택하였고 그중 3년의 데이터는 훈련데이터, 나머지 1년의 데이터는 테스트 데이터로 사용한다. 스웜학습을 이용하여 훈련한 모델의 성능을 측정하기 위하여 훈련데이터를 연도별로 나누었다. 스웜학습에서 에지 노드를 3개를 사용하여 3개의 데이터세트는 각각의 노드가 하나씩 부여받는다. 그리고 각 노드의 데이터세트를 12등분으로 나누었다. 첫 번째 데이터세트는 두번째 데이터세트에 포함되고 있으면 12번째는 모든 데이터세트를 포함하고 있게 하여 시간이 갈수록 데이터가 각각의 에지 게이트웨이에 쌓이도록 표현한다.

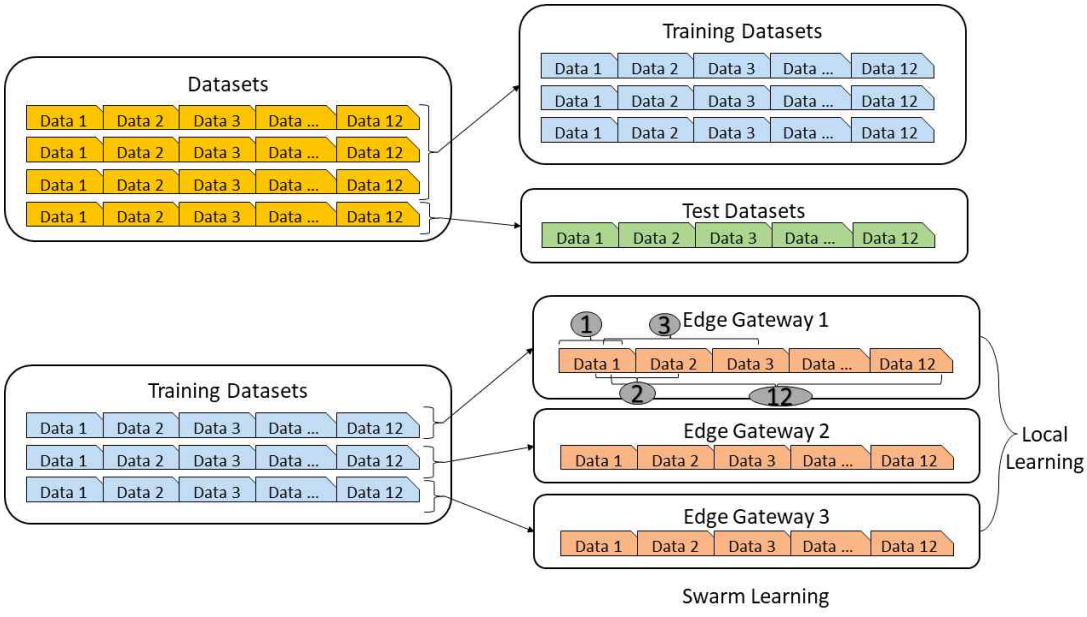


그림 4.4 PMV 모델 데이터세트 구성

월별, 연별 데이터의 수는 서로 다른 분포를 띄고 있어 데이터는 Non-IID 분포를 가지고 있다. 로컬에서 학습한 모델의 성능을 측정하기 위하여 훈련데이터 세트를

위와 같은 방식으로 3년의 데이터를 12 등분으로 나누었다. 마찬가지로 테스트 데이터도 12등분으로 나눈다.

스웜학습 혹은 연합학습은 글로벌 모델을 업데이트하는 방식으로 가중치 평균의 방식 FedAVG (FederatedAveraging) [20]를 사용한다. N은 노드 개수, E는 에포크(epoch), 반복 회수는 R로 표시한다.

본고에서 개선한 스웜학습의 성능을 평가하기 위하여 로컬에서 모델을 훈련하는 방식과 FedAVG를 사용하여 모델을 훈련하는 연합학습 및 기존 스웜학습 방식의 성능들을 측정하여 비교한다. 모델학습에 관한 설정은 표 4.3과 같이 설정한다. 분산 환경에서의 모델학습 전략은 1, 10, 100번의 반복 회수 그리고 에포크를 10으로 설정하였고, 로컬 환경에서 모델학습 전략은 10, 100, 1000의 에포크를 설정하여 모든 실험에서 모델에 대한 업데이트 회수를 동일하게 하여 측정한다.

표 4.3 기존 연합/스웜 학습 전략 및 개선된 스웜학습 전략 환경 설정

	Optimization	Desktop	N	R	E
Proposed SL	Sequential SGD	-	3	1, 10, 100	10
FL	FedSGD	1	3	1, 10, 100	10
SL	FedSGD	-	3	1, 10, 100	10
Local	SGD	1	-	-	10, 100, 1000

연합학습의 개발환경은 표 4.4와 같다. 연합학습은 글로벌 모델의 업데이트를 담당하는 서버와 로컬 모델을 학습하는 클라이언트로 구성되고 있다. 본고에서는 서버를 윈도우 10 데스크톱에서 구현한다. 기계학습을 구현하기 위하여 TensorFlow 플랫폼을 사용하였고, 서버 기능을 제공하기 위하여 Flask 프레임워크를 사용한다. 데이터를 처리하는 기능은 Pands 그리고 NumPy 파이선 라이브러리를 사용하였고 기본 언어는 파이선을 사용한다. 본고에서 사용하는 데이터는 위에서 언급했듯이 미리 수집된 데이터이기 때문에 파일 시스템을 이용하여 데이터를 입력한다. 라즈베리파이 3대를 이용하여 클라이언트 기능을 구현한다. 서버에서의 로컬 모델 훈련 명령을 받을 수 있도록 Flask를 이용하여 서버로 구현하여 REST APIs를 제공한다. 그리고 기계학습 모델을 훈련하기 때문에 TensorFlow를 우분투 20.04에 설치한다. 개발 툴은 두 하드웨어에서 모두 Visual Studio Code를 사용한다.

표 4.4 기존 연합학습 전략 개발환경

Hardware			Software	
Desktop	OS	Windows 10	Programming Language	Python 3.8
	CPU	Intel® core ™ i5-8500 3GHz	Libraries	Pandas, NumPy
	Memory	32GB	Application	Visual Studio Code
	Hard Disk	500GB		
Raspberry Pi 4	OS	Ubuntu 20.04 64bit	Framework & Platform	Flask
	CPU	Quad Core 1.5GHz 64bit		TensorFlow 2.6
	Memory	4GB	Database	File System
	MicroSD Card	16GB		

연합 학습은 그림 4.5와 같이 구성되고 있다. 연합 학습은 서버와 클라이언트로 구분하여 구성되고 있다. 데스크톱은 연합 학습의 서버 역할을 하고 모델 수집 및 업데이트 기능 그리고 업데이트한 클로벌 모델을 에지 게이트웨이 노드로 전달하는 기능을 제공한다. 3대의 에지 게이트웨이를 이용하여 연합 학습의 클라이언트를 구현한다. 서버에서 전달하는 글로벌 모델과 로컬 데이터세트를 이용하여 모델을 훈련한다. 훈련한 모델은 다시 서버로 전달한다.

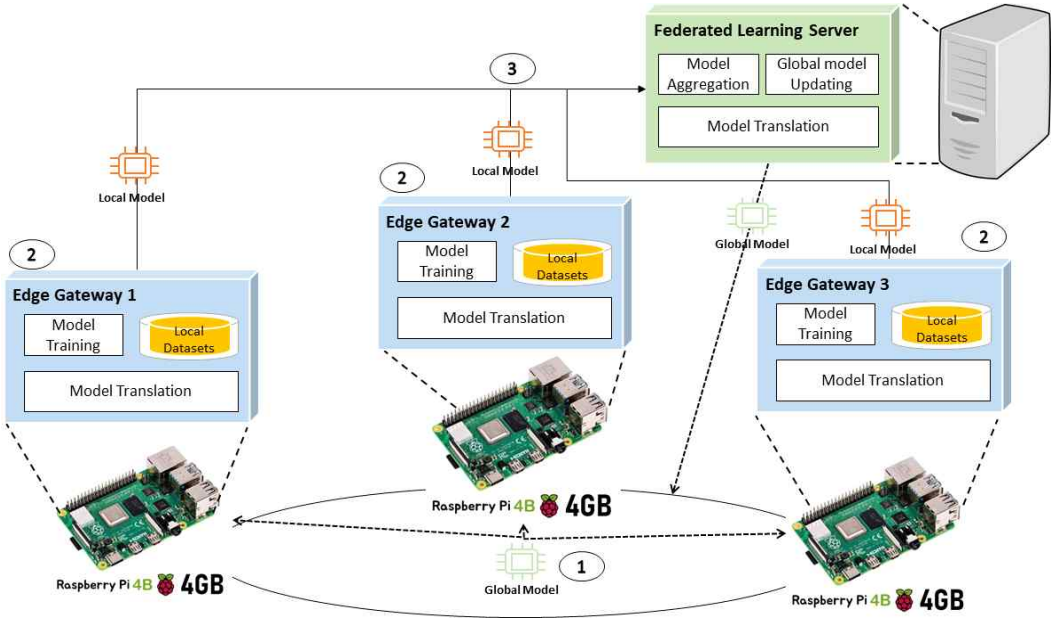


그림 4.5 PMV 예측을 위한 연합학습 전략 구성도

연합 학습의 순서도는 그림 4.6과 같다. 연합 학습 서버는 시작과 함께 반복 회수, 글로벌 모델 가중치, 로컬 모델 가중치 및 에지 게이트웨이들의 정보를 초기화 한다.

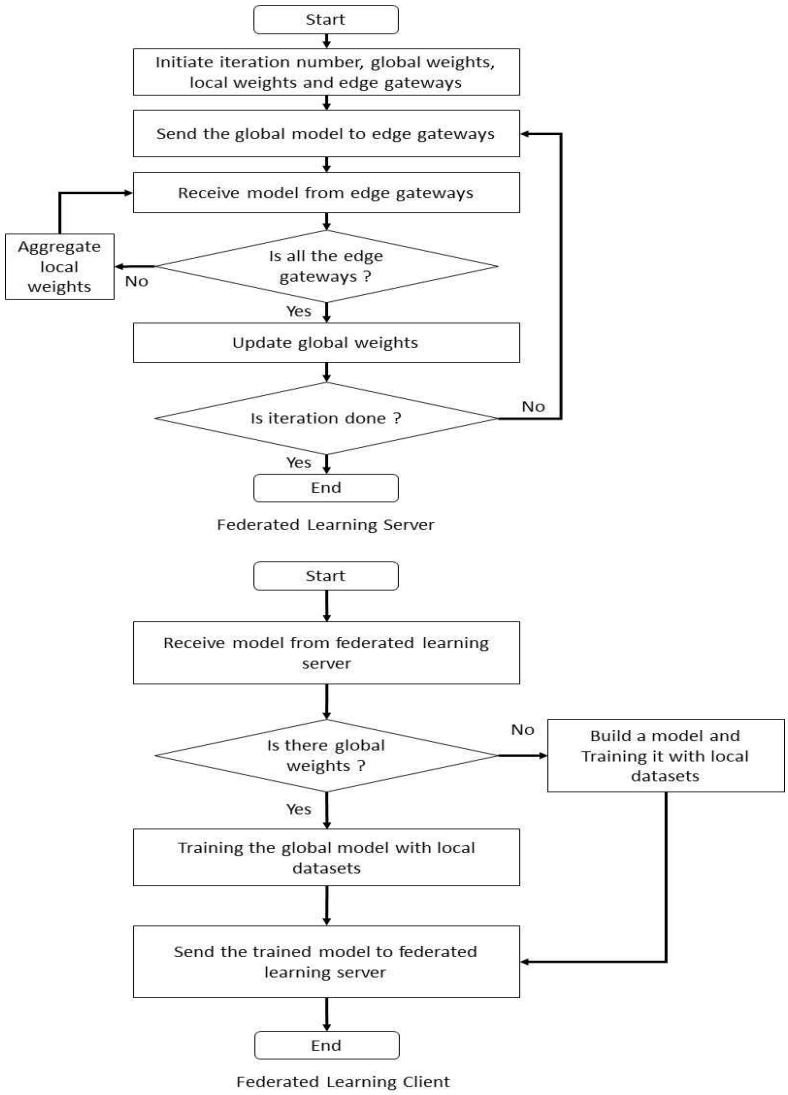


그림 4.6 기존 연합학습 전략 순서도

모델을 훈련을 시작하면 서버는 글로벌 모델을 클라이언트로 전달하여 훈련을 시작한다. 서버는 모델 학습에 참가한 모든 클라이언트들이 로컬 모델을 전달할 때까지 로컬 모델 가중치를 수집한다. 모든 클라이언트들의 로컬모델이 전달되면 글로벌 모델을 업데이트하고 반복 회수를 확인한다. 반복 회수만큼 훈련이 완성되면 훈련을 종료하고 아니면 다시 글로벌 모델을 모든 클라이언트로 전달한다. 각각의 로

컬 클라이언트들은 서버로부터 글로벌 모델이 전달되면 훈련을 시작한다. 훈련에 앞서 먼저 글로벌 모델이 전달되었는지를 확인한다. 널값의 글로벌 모델이 전달되면 모델을 생성하고 로컬 데이터세트를 이용하여 모델을 훈련한다. 그렇지 않으면 글로벌 모델과 로컬 데이터세트를 이용하여 모델을 훈련한다. 훈련된 모델은 마지막으로 서버로 다시 전달된다.

연합 학습 전략으로 반복 회수를 1, 10 그리고 100번으로 설정하여 모델을 훈련하여 성능을 측정한다. 그림 4.7은 연합 학습 성능 측정결과를 보여준다. 반복 회수가 많을수록 그리고 데이터세트가 클수록 모델의 성능이 점점 좋아진다. 10번의 반복 회수는 데이터세트가 클수록 안정적으로 성능이 좋아지고 있지만 100번의 반복 회수가 결국은 최고의 성능을 보여주었다. 1번의 반복 회수와 10번의 반복 회수를 비교하였을 때 데이터가 적을 때는 1번의 반복 회수로 훈련된 모델이 좋은 성능을 나타내지만 데이터의 수가 많을수록 10번의 반복 회수가 좋은 성능을 보여주고 있다. 100번의 반복 회수로 훈련한 모델은 1번과 2번의 실험에서 가장 나쁜 성능을 보여주고 있지만, 나머지 실험에서 모두 가장 좋은 성능을 보여주고 있다.

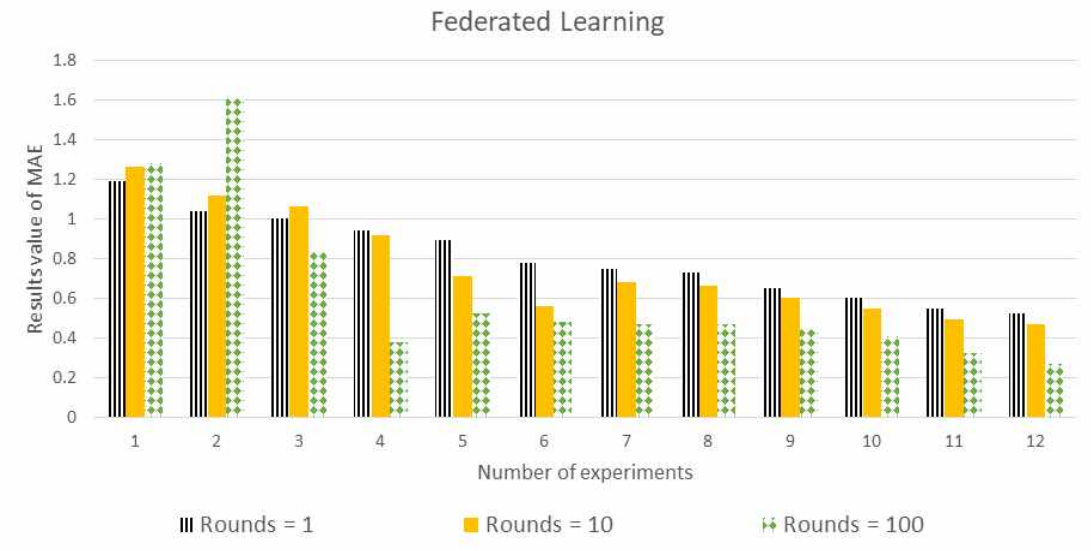


그림 4.7 기존 연합학습 전략 정확도 (MAE) 성능 결과

표 4.5는 스웜학습 개발환경을 상세히 나열한 것이다. 스웜학습은 에지 컴퓨팅 네트워크의 노드들을 이용하여 기계학습을 실현하는 방법이다. 본고에서는 라즈베리파이 하드웨어에 우분투 20.04 버전을 설치하여 에지 게이트웨이를 구축한다. 3대의

라즈베리파이가 에지 컴퓨팅 네트워크 환경을 구성하였고, 에지 네트워크에서 기계학습을 실현할 수 있도록 서버로 구현한다.

표 4.5 스웜학습 개발환경

Hardware			Software	
Raspberry Pi 4	OS	Ubuntu 20.04 64bit	Programming Language	python 3.8
			Libraries	Pands, NumPy
	CPU	Quad Core 1.5GHz 64bit	Application	Visual Studio Code
	Memory	4GB	Framework and Platform	Flask
				TensorFlow 2.6
	MicroSD Card	16GB	Database	File system

기계학습 기능을 제공하기 위하여 TensorFlow 플랫폼을 사용하였고, 서버를 구축하기 위하여 파이선 언어를 사용하는 Flask 프레임워크를 사용한다. 파일 시스템에 저장되어있는 로컬 데이터를 처리할 수 있도록 Pands 그리고 NumPy 파이선 라이브러리를 사용한다. 개발 툴은 Visual Studio Code를 선택한다.

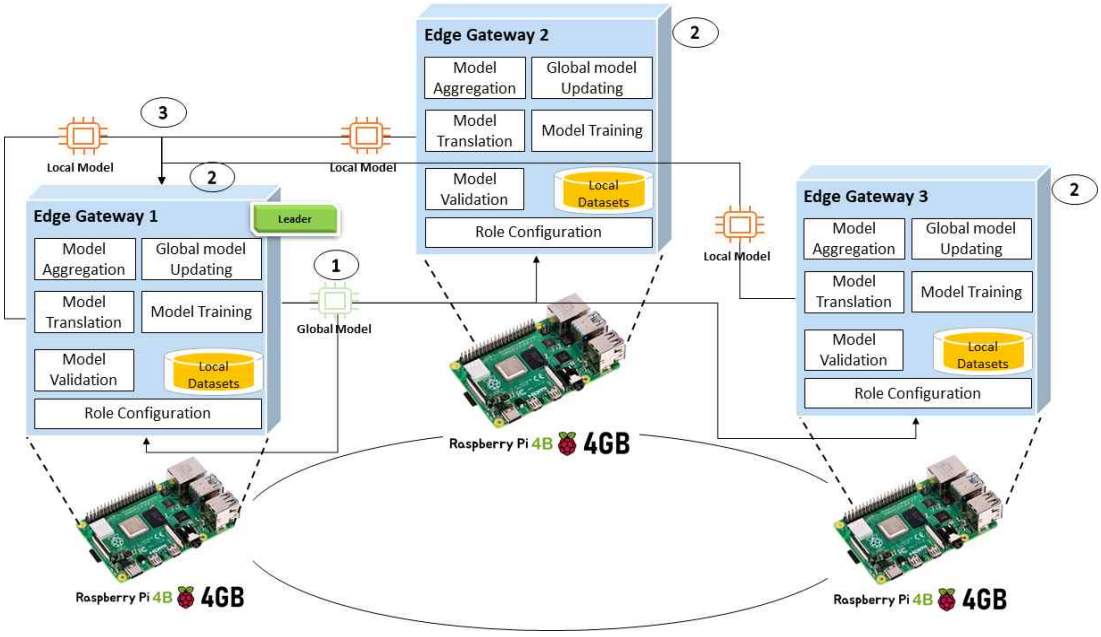


그림 4.8 PMV 예측을 위한 기존 스웜학습 전략 구성도

기존 스웜학습은 그림 4.8과 같이 구성한다. 기존의 스웜학습은 리더와 에지 게이트웨이들로 구분하여 구성되고 있다. 기존의 스웜학습에서 리더를 에지 네트워크

에서 게이트웨이들이 자동으로 선택하지만 성능 테스트 편의상 본고에서는 임의로 에지 게이트웨이중의 하나를 리더로 선택한다. 3대의 에지 게이트웨이들을 이용하여 기존 스웜학습 환경을 구성한다. 모든 에지 게이트웨이가 모델 훈련, 모델 수집, 업데이트 및 전달 기능을 제공하지만, 리더로 선택된 에지 게이트웨이만이 모델 수집 및 업데이트 기능을 수행한다.

기존 스웜학습의 순서도는 그림 4.9와 같다. 선택된 에지 게이트웨이가 먼저 반복 회수, 글로벌 모델 가중치, 로컬 모델 가중치 및 에지 게이트웨이들의 정보를 초기화한다. 모델 훈련을 시작하기 위하여 등록된 모든 에지 게이트웨이 (리더로 선택된 에지 게이트웨이도 포함)로 글로벌 모델을 전달한다. 리더는 모든 에지 게이트웨이가 로컬 모델을 전달할 때까지 로컬 모델 가중치를 수집한다. 로컬모델이 모두 전달되면 글로벌 모델을 업데이트하고 반복 회수를 확인한다. 반복 회수만큼 훈련이 완성되면 훈련을 종료하고 아니면 다시 글로벌 모델을 모든 에지 게이트웨이 전달한다. 에지 게이트웨이는 연합 학습의 클라이언트와 같은 작업을 진행한다.

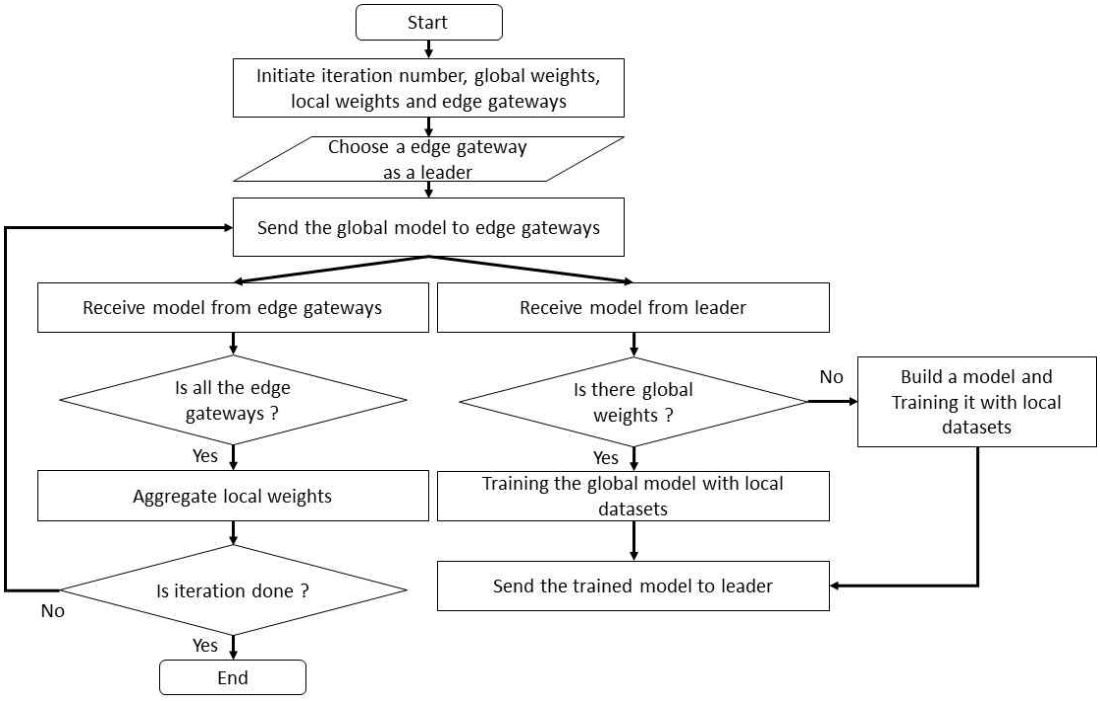


그림 4.9 기존 스웜학습 전략 순서도

기존 스웜학습 전략으로 반복 회수를 1, 10 그리고 100번으로 설정하여 모델을

훈련하여 성능을 측정한다. 그림 4.7은 기존 스웜학습 성능 측정결과를 보여준다. 연합 학습과 같은 방법으로 글로벌 모델을 업데이트하기 때문에 성능이 비슷한 추세를 보이고있다. 마찬가지로 10번의 반복 회수는 데이터세트가 클수록 안정적으로 성능이 좋아지고 있지만 100번의 반복 회수가 결국은 최고의 성능을 보여주었다. 1번의 반복 회수와 10번의 반복 회수를 비교하였을 때 2에서 4까지의 실험에서 1번의 반복 회수로 훈련된 모델이 좋은 성능을 나타내지만, 나머지 실험에서 10번의 반복 회수로 훈련한 모델이 좋은 성능을 보여주고 있다. 100번의 반복 회수로 훈련한 모델은 1번과 2번의 실험 외에 나머지 실험에서 모두 가장 좋은 성능을 보여주고 있다. 총체적으로 보았을 때 반복 회수가 많을수록 그리고 데이터세트가 클수록 모델의 성능이 점점 좋아지고 있다.

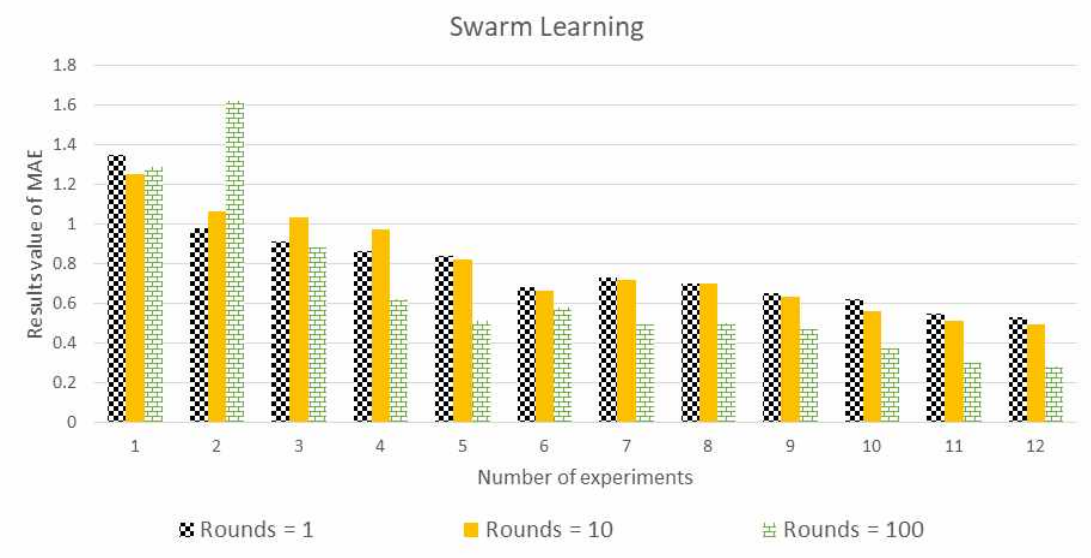


그림 4.10 기존 스웜학습 전략 정확도 (MAE) 결과

본고에서 개선한 스웜학습은 그림 4.11과 같이 구성한다. 개선한 스웜학습은 시작점으로 선택된 에지 게이트웨이와 다른 에지 게이트웨이들로 구분하여 구성되고 있다. 개선한 스웜학습은 디지털 트윈 에지 컴퓨팅과 통합되어 임의로 에지 게이트웨이 중의 하나를 시작점으로 수동으로 선택할 수 있다. 3대의 에지 게이트웨이들을 이용하여 제안한 스웜학습 환경을 구성한다. 모든 에지 게이트웨이가 모델 훈련, 모델 검증, 전달기능을 똑같이 제공한다. 시작점에서 훈련한 모델이 인접한 다른 에지 게이트웨이로 전달되어 그 에지 게이트웨이의 로컬 데이터세트를 이용하여 훈련하

고 또 다시 인접한 에지 게이트웨이로 전달되어 시작점으로 최종적으로 전달되어 한 번의 모델 훈련을 완성한다. 모든 에지 게이트웨이는 라즈베리파이로 구성되었고 모델을 훈련하고 전달하고 검증할 수 있는 기능을 제공할 수 있도록 서버를 실행하고 있다. 각각의 에지 게이트웨이는 모델 훈련에 사용하는 로컬 데이터세트를 보유하고 있고 데이터세트는 서로 다른 분포를 가지고 있다.

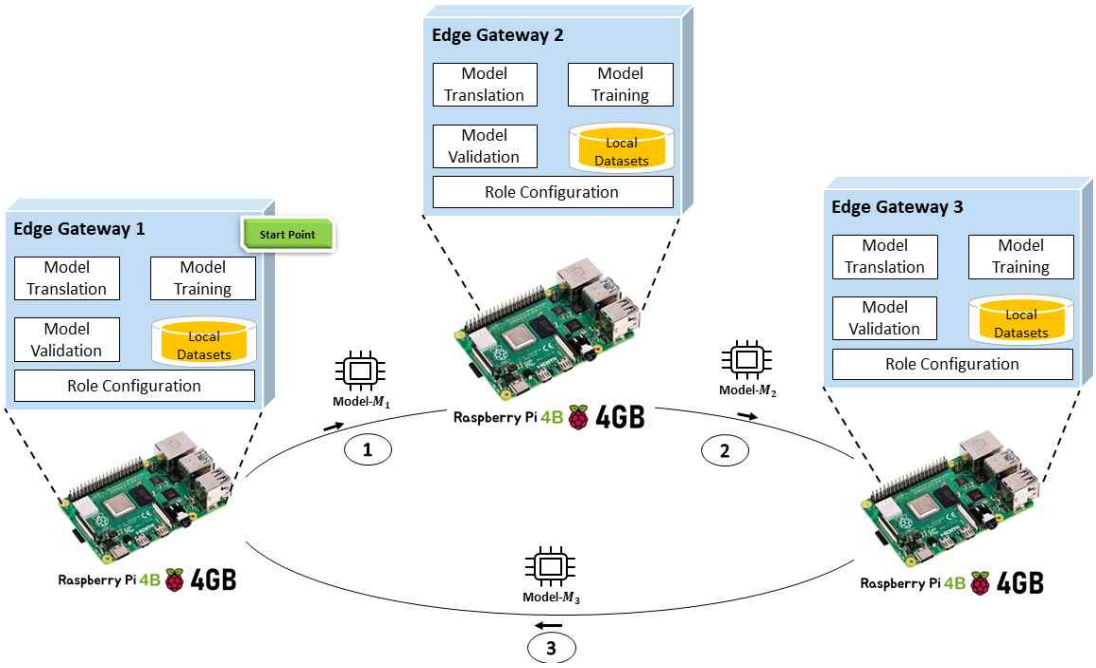


그림 4.11 PMV 예측을 위해 개선한 스웜학습 전략 구성도

저희가 개선한 스웜학습의 순서도는 그림 4.12과 같다. 모든 에지 게이트웨이는 시작과 함께 로컬 데이터세트를 이용하여 로컬 모델을 훈련한다. 선택된 시작점인 에지 게이트웨이는 반복 회수 초기화와 함께 훈련된 로컬 모델을 다음의 에지 게이트웨이로 전달한다. 모델을 전달받은 에지 게이트웨이는 모델을 모컬 데이터세트를 이용하여 훈련을 하고 다음의 에지 게이트웨이로 전달한다. 시작점인 에지 게이트웨이는 모델을 전달받으면 기존의 로컬 모델과 성능을 비교하여 더 좋은 모델로 교체한다. 다음은 반복 회수를 체크하여 반복 회수만큼 훈련이 완성되면 훈련을 종료하고 아니면 다시 로컬 모델을 다음의 에지 게이트웨이로 전달한다. 훈련이 종료 되면 최종 모델을 모든 에지 게이트웨이로 전달한다. 모델을 전달받은 에지 게이트웨이들은 각각 모델 검증을 통하여 성능이 좋은 모델로 로컬 모델을 교체한다.

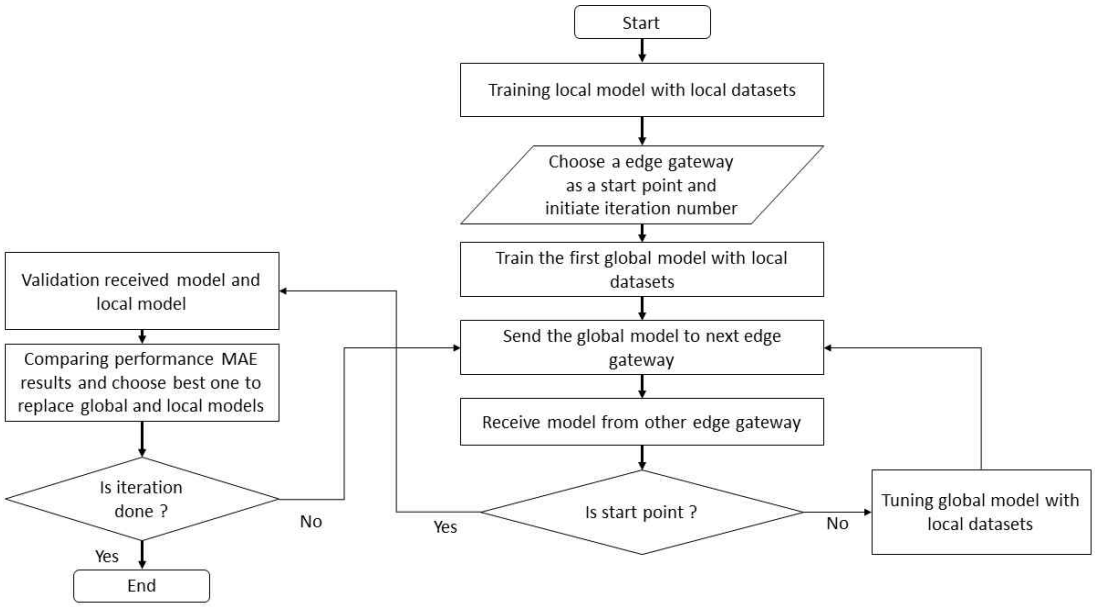


그림 4.12 개선한 스웜학습 전략 순서도

개선한 스웜학습 전략으로 반복 회수를 1, 10 그리고 100번으로 설정하여 모델을 훈련하여 성능을 측정한다. 그림 4.13은 개선한 스웜학습 성능 측정결과를 보여준다. 성능 측정결과를 보았을 때 1, 10, 혹은 100번의 반복 회수에도 불구하고 본고에서 개선한 스웜학습 전략으로 훈련한 모델의 성능이 더 안정적인 것을 확인할 수 있다.

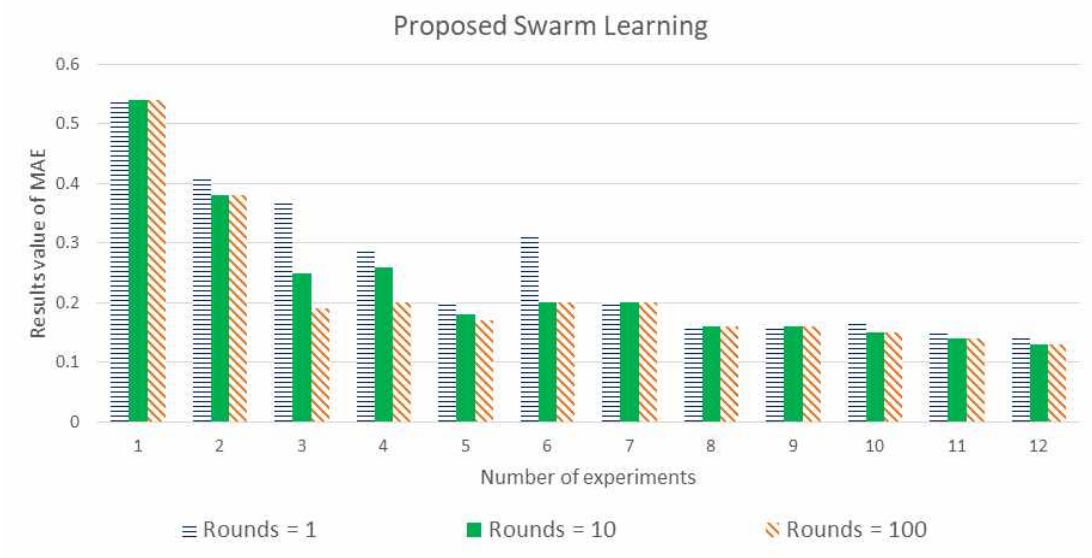


그림 4.13 개선한 스웜학습 전략 정확도 (MAE) 결과

첫 번째 실험에서 1번의 반복 회수로 훈련한 모델이 미세한 차이로 좋은 성능을 보여주고 있다. 그 외에서는 더욱 많은 반복 회수로 훈련한 모델이 모든 실험에서 좋은 성능을 보여주고 있다. 그중 10, 100번의 반복 회수로 훈련한 모델의 성능 차이는 실험의 차수가 적을수록 즉 데이터의 수가 커짐에도 불구하고 멀어지지 않는다. 1, 10 그리고 100번의 반복 회수로 훈련한 모델은 실험7 이후로는 0,2와 0.1사이의 성능을 보여주고 있고 점점 0,1에 가까워지고 있다.

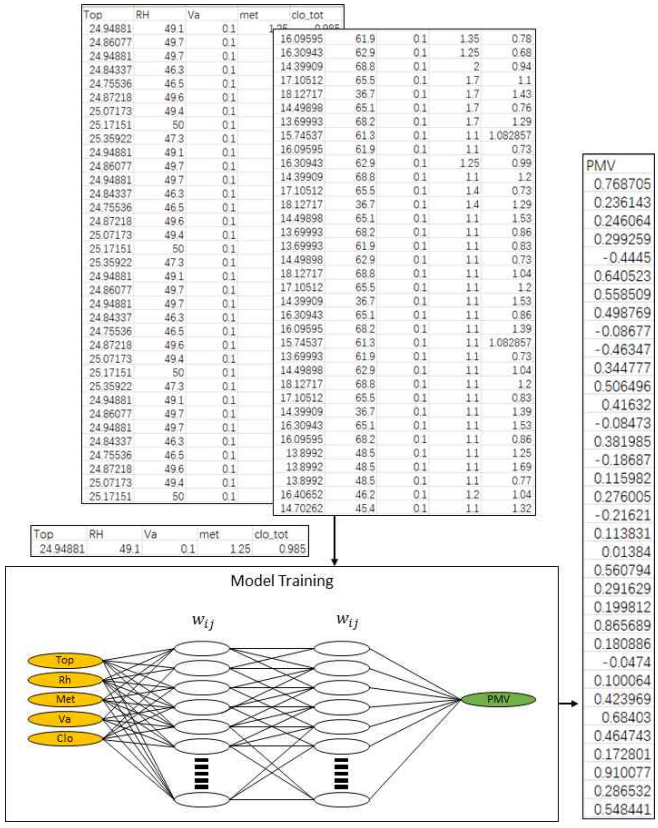


그림 4.14 PMV 예측을 위한 로컬학습방법

그림 4.14는 로컬모델 훈련방법을 표현한다. 데이터세트와 모델이 모두 하나의 데스크톱에 위치한다. 먼저 모델을 생성하고 데이터세트와 함께 훈련한다. 훈련을 통하여 모델의 가중치 값의 최적화를 에포크 회수만큼 업데이트한다. 모든 데이터가 한 대의 데스크톱에 위치하여 모델을 전달하는 비용이 없으므로 에지 컴퓨팅환경을 이용하여 모델을 훈련하는 방법보다 빠르다.

3. 기존 연합/스웜 학습과 개선된 스웜학습 성능 비교분석

그림 4.15는 로컬에서 모델을 학습하는 방식과 FedAVG를 사용하여 모델을 훈련하는 연합 학습 및 기존 스웜학습 방식 그리고 본고에서 제안한 스웜학습 전략을 통하여 모델을 학습한 성능의 측정결과이다. 스웜학습 전략을 사용하여 훈련한 모델의 성능을 나타내는 도표이다. 로컬에서 훈련한 모델의 성능과 스웜학습 전략으로 훈련한 모델의 성능을 수집하여 비교한다. 총 12회의 학습을 진행하였고 로컬에서 훈련한 모델은 에포크를 10으로 지정하였고, 분산 환경에서 훈련한 모델은 반복 회수를 1로 설정하고 로컬 에포크를 10으로 설정하여 측정을 한다. 저희가 개선한 학습 전략으로 훈련한 모델은 모든 실험에서 좋은 성능을 보이고 있고 2배, 심지어 3배 정도의 차이를 보이고 있다.

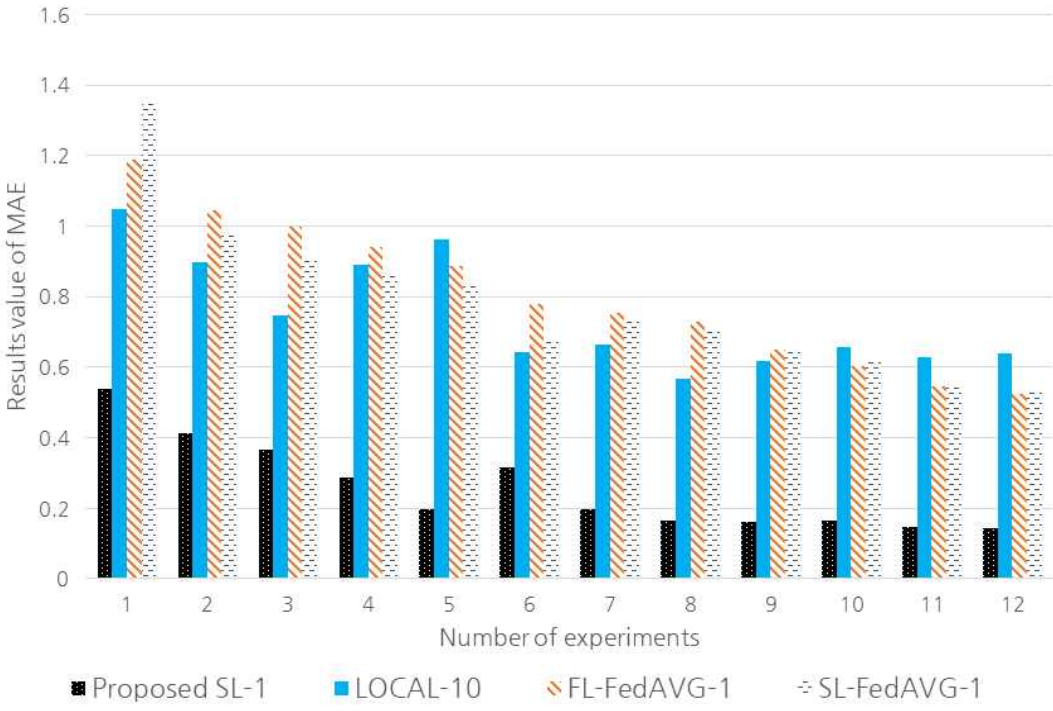


그림 4.15 분산 에지 지능 환경에서 기존 연합/스웜 및 개선된 스웜학습 전략 성능비교 (1회)

그림 4.16은 로컬에서 훈련한 모델의 에포크를 10으로 설정하고, 분산 환경에서

훈련한 모델의 반복 회수를 10으로 설정하고 로컬 에포크를 10으로 설정하였을 때의 성능 측정결과이다. 본고에서 개선한 학습 전략으로 훈련한 모델은 실험 1에서 0.4이상의 성능을 보였지만 나머지 실험에서 0.4이하의 성능 심지어 5번째 실험부터는 0.2 이하의 성능을 보여주고 있다. 그에 비해 다른 방법으로 훈련한 모델은 실험이 1이상의 성능으로부터 실험이 끝날 때까지 0.4이상의 성능을 보여주고 있다. 그림에서 저희가 개선한 학습 전략으로 훈련한 모델이 좋은 성능을 보여주고 있는 데 반해 다른 학습 전략은 뚜렷한 변화를 보여주고 있지 않다.

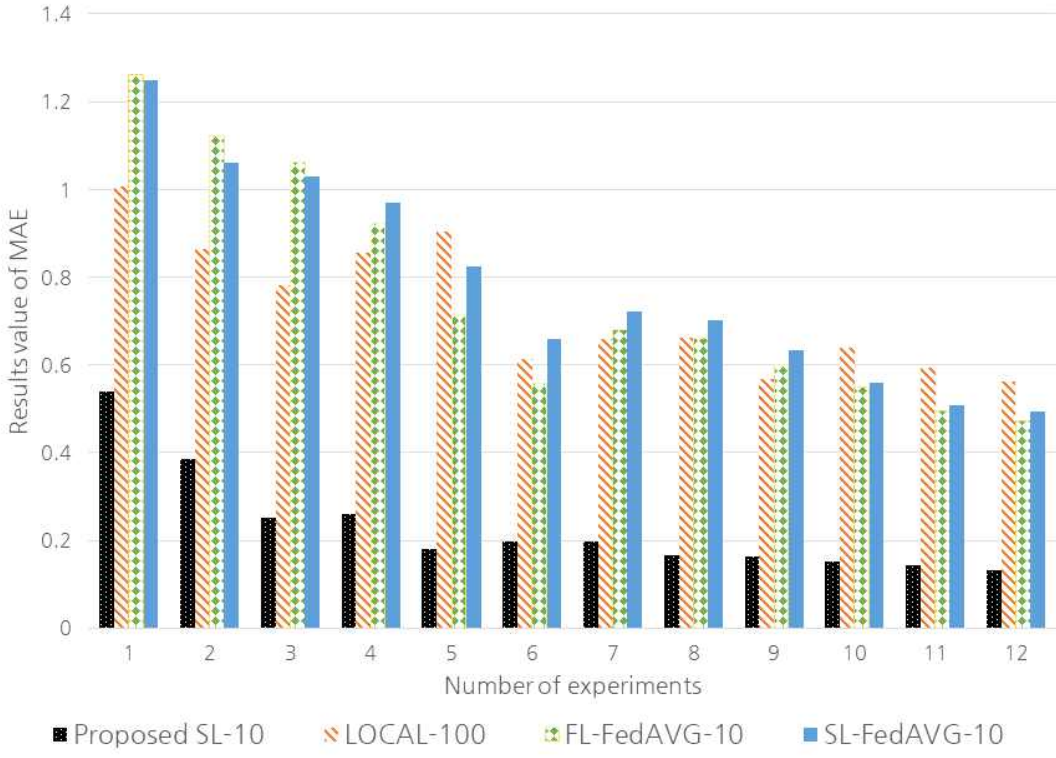


그림 4.16 분산 에지 지능 환경에서 기존 연합/스웜 및 개선된 스웜학습 전략 성능비교 (10회)

그림 4.17는 로컬에서 훈련한 모델의 에포크를 100으로 설정하고, 분산 환경에서 훈련한 모델의 반복 회수를 100, 로컬 에포크를 10으로 설정하였을 때의 성능 측정결과이다. 연합 학습 및 기존 스웜학습 전략이 데이터의 수가 많아질수록 좋은 성능을 보여주고 있다. 하지만 저희가 개선한 스웜학습 전략에 비해 선명한 차이를 보이

고 있다.

본고에서 개선한 스웜학습 전략이 모든 실험에서 좋은 성능을 보여주고 있다. 반복 회수를 각각 1, 10, 100으로 설정하여 테스트를 진행하였지만 다른 학습 전략을 통하여 학습한 모델의 성능이 저희가 개선한 스웜학습 전략으로 1번의 훈련으로 학습한 모델보다도 나쁜 성능을 보여주고 있다.

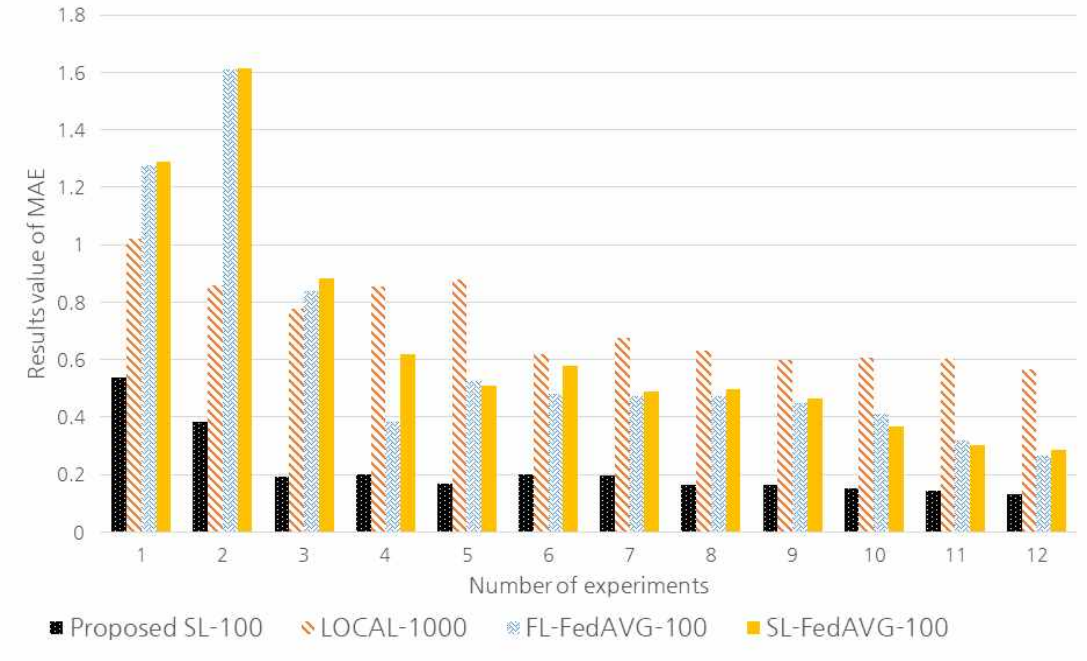


그림 4.17 분산 에지 지능 환경에서 기존 연합/스웜 및 개선된 스웜학습 전략 성능비교 (100회)

그림 4.18는 분산 환경에서 모델을 한 번 업데이트 하는 데 사용한 시간을 측정한 결과이다. 기존 스웜학습 전략은 리더 선택부분을 생략하여 모델만 학습하는 데 사용한 시간을 측정한다. 연합 학습 전략과 기존 스웜학습 전략이 모든 실험에서 1초 좌우에서 한번의 모델 업데이트를 완성한다. 반면 본고에서 개선한 스웜학습 전략은 대략 3배 많은 시간을 사용하고 있다.

본고에서 개선한 스웜학습 전략은 하나의 모델을 에지 네트워크 컴퓨팅에 있는 에지 게이트웨이들이 순서대로 업데이트하여 학습하는 방식을 사용하여 병렬로 학습하는 연합 학습 및 기존 스웜학습에 비해 많은 학습시간을 사용하지만 좋은 성능의 모델을 학습할 수 있다.

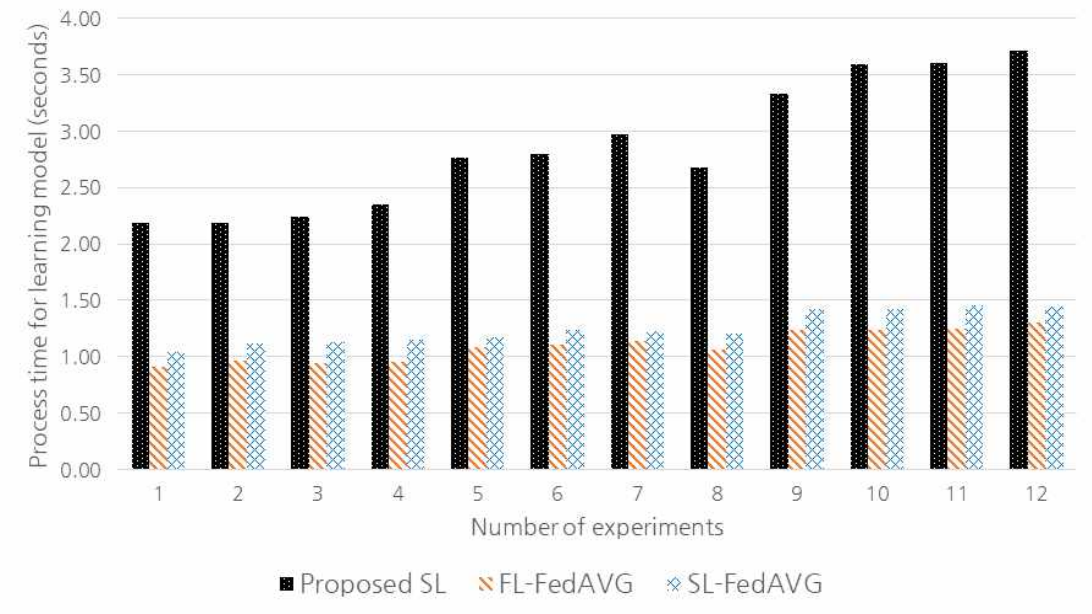


그림 4.18 분산 에지 지능 환경에서 모델 훈련 성능 비교 (지연 시간)

V. PSO 알고리즘 기반 최적화 메커니즘

1. PMV 최적화를 위한 PSO 알고리즘

집은 사람에게 육체적, 정신적 필요를 충족시키기 위해 안전한 생활 환경과 안락함을 제공하는 가장 중요한 장소이다. 사람들이 가족, 친구와 함께 모일 뿐만 아니라 휴식을 취하는 활동을 하고 오락과 즐거움을 얻고 잠을 잘 수 있는 곳이다. 사물인터넷의 지속적인 발전으로 인해 사람들의 생활환경이 크게 영향을 받고 있다. 특히 스마트홈에 대한 연구는 집을 단순한 생활환경이 아닌 보다 지능적이고 스마트한 환경으로 변화시키고 있다. 스마트 홈을 통해 사람들은 대중적인 주거 환경에서 맞춤형 주거 환경의 혜택을 받고 있다.

거주자의 열 쾌적성에 맞춤화된 주거 환경을 제공 할 수 있도록 입자 군집 최적화 (Particle Swarm Optimization, PSO) 알고리즘을 에지 컴퓨팅에 적용할 것을 제안한다. PSO에서는 하나의 에이전트가 최적해를 찾는 것에 비해 최적해를 찾는 에이전트가 군집을 이루어 서로 정보를 교환하면서 최적해를 찾아 지역적인 최적해에 빠지더라도 전체적으로 글로벌 최적해에 수렴할 수 있다.

저희는 열 쾌적성을 최적화하는 스마트 홈 환경을 유지하기 위하여 PMV 모델과 PSO 알고리즘을 통합한다. PSO는 그림 5.1과 같이 PMV 모델 객체 및 입자의 파라미터를 생성하는 초기화 단계 라인 3 ~ 9, 입자를 최적해의 위치로 가도록 하는 속도 변수 계산 단계 라인 18, 입자의 위치를 업데이트 하는 단계 라인 18, 최적해 평가 단계 12 ~ 15, 마지막으로 종료 조건 확인 단계 라인 16 ~17로 구성되고 있다.

초기화 단계에서 PMV 모델 객체, 목표하는 PMV 값과 허용하는 오차를 설정하고 군집에 속하는 입자의 개수를 S 로 지정한다. 그리고 각각의 입자의 위치 값 x_i 와 속도 값 v_i 변수를 생성한다. x_i는 스마트 홈에서 수집되는 현재 온도 및 습도의 값을 기반으로 임의 벡터로 초기화된다. p_i 는 입자 i의 최적해 값이고 g는 전체 군집에서의 최적해이다. 입자는 최적해 즉 쾌적한 온도와 습도를 찾는 최적해 에이전트

이다.

초기화 단계를 마치면 최적해 평가 단계를 실행한다. 최적해 평가 단계는 종료 조건 즉 반복 횟수 또는 글로벌 최적해와 목표하는 최적해 값의 차이가 허용하는 오차 범위 내에 있으면 종료한다. 최적해 평가 단계에서 로컬 최적해 즉 p_i 또는 글로벌 최적해 g를 찾는다. 각 입자의 x_i 를 PMV 모델에 입력하여 예측한 열적 쾌적도 수준(Thermal Comfort Level, TCL)과 현재의 로컬 및 글로벌 최적해가 각각 목표하는 값과의 차이가 가장 작은 값으로 대체한다. 군집에 생성된 입자의 수 만큼 반복문을 실행한다. 반복 과정에서 각 입자의 속도 변수도 업데이트된다. 모든 반복이 완료되면 입자의 속도 변수에 근거하여 위치를 업데이트한다.

```
1.  Input: current temperature and humidity.
2.  Output: optimized temperature and humidity.
3.  Instantiation PMV model as pmv
4.  Initialize target PMV value and target error as target and target-error
5.  for particle i = 1, ..., S do
6.      Initialize the particle's position with random vector from input: x_i
7.      Initialize the particle's best known position to its initial position: p_i ← x_i
8.      Initialize the particle's velocity: v_i
9.  end for
10. while a termination criterion is not met do:
11.     for each particle i = 1, ..., S do
12.         if abs(pmv(x_i)-target) < abs(pmv(p_i)-target) then
13.             Update the particle's best known position: p_i ← x_i
14.         if abs(pmv(p_i)-target) < abs(pmv(g)-target) then
15.             Update the swarm's best known position: g ← p_i
16.         if abs(pmv(g)-target) <= target-error then
17.             break
18.         Update the particle's velocity.
19.     end for
20.     Update the particle.
21. end while
22. Output set optimized temperature and humidity from g.
23. return output.
```

그림 5.1 PMV를 통합한 PSO 슈도코드

그림은 5.2는 최적화 엔진의 실행과정을 기능 블록으로 표현한다. 현재 온도와

습도가 입력되면 초기화 단계를 실행하는데 Search space가 space 인스턴스를 생성하고 온도, 습도, 희망한 PMV 값, 허용 범위, 그리고 입자 개수를 입력 파라미터로 한다. 입자 개수만큼 온도와 습도를 기반으로 초기 값이 생성되어 position 즉 위치의 벡터 값으로 저장된다. 최적화된 온도 습도 생성을 위하여 Pbest 선택, Gbest 선택, 최적해 평가 그리고 입자들 업데이트 단계를 반복 회수만큼 반복적으로 실행한다. Pbest는 입자들을 하나하나 찾아서 자신이 가진 Pbest 값과 현재 가진 위치 값의 PMV 값 중에서 희망한 PMV 값과의 차이가 적은 것으로 변경한다. Gbest도 마찬가지로 Gbest 값 변경을 변경한단. Fitness 는 현재 입자가 가진 위치 값 즉 온도와 습도를 입력으로 받아 PMV 값을 예측한다. 최적해 평가는 반복 횟수 또는 Gbest 값과 희망하는 최적해 값의 차이가 허용하는 오차 범위 내에 있으면 마치고 Gbest위치의 온도와 습도를 최적의 해로 반환한다. 입자들 업데이트는 입자들을 하나하나 찾아서 속도를 업데이트하고 업데이트된 속도를 이용하여 위치 값 즉 온도와 습도 값을 업데이트한다.

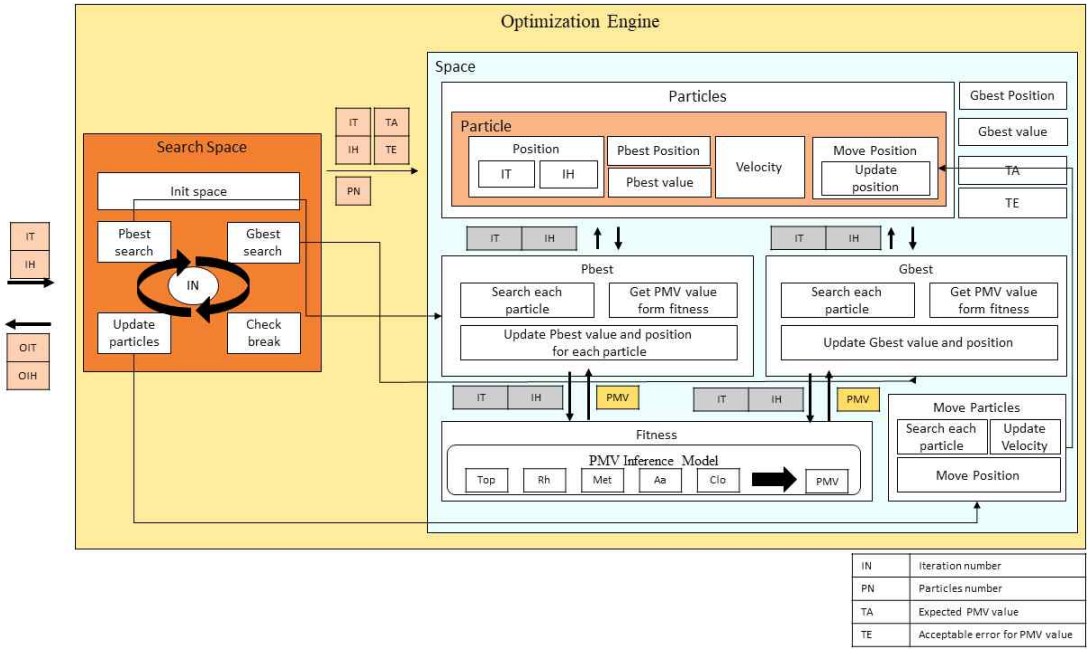


그림 5.2 PSO 최적화 엔진의 기능 블록

2. 스마트 홈에 PMV 최적화 메커니즘 구현

그림 5.3은 본고에서 제안하는 PMV 기반의 PSO 알고리즘을 테스트하는데 필요한 환경 구조를 설명한다. 에지 게이트웨이와 스마트 홈 에뮬레이터가 서로 데이터를 주고 받으면서 PSO 최적화 알고리즘의 성능을 측정한다.

에지 게이트웨이는 스마트 홈 에뮬레이터에게 데이터를 요청하고 PSO 알고리즘을 이용하여 최적의 실내 온도 습도 값을 생성하여 다시 컨트롤러로 데이터를 전달한다. 스마트 홈 에뮬레이터는 스마트 홈 환경 데이터세트를 기반으로 실내 온도 (IT), 실내 습도 (IH), 실외 온도 (OT), 실외 습도 (OH), 히터 전력 (HP) 데이터를 생성한다. 에지 게이트웨이에서 생성한 최적의 실내 온도 습도 값을 기반으로 컨트롤러를 이용하여 실내 환경 변화를 반영한다. 변화된 실내 환경 파라미터는 에지 게이트웨이로 다시 전달되어 테스트에 필요하는 데이터를 시뮬레이션한다.

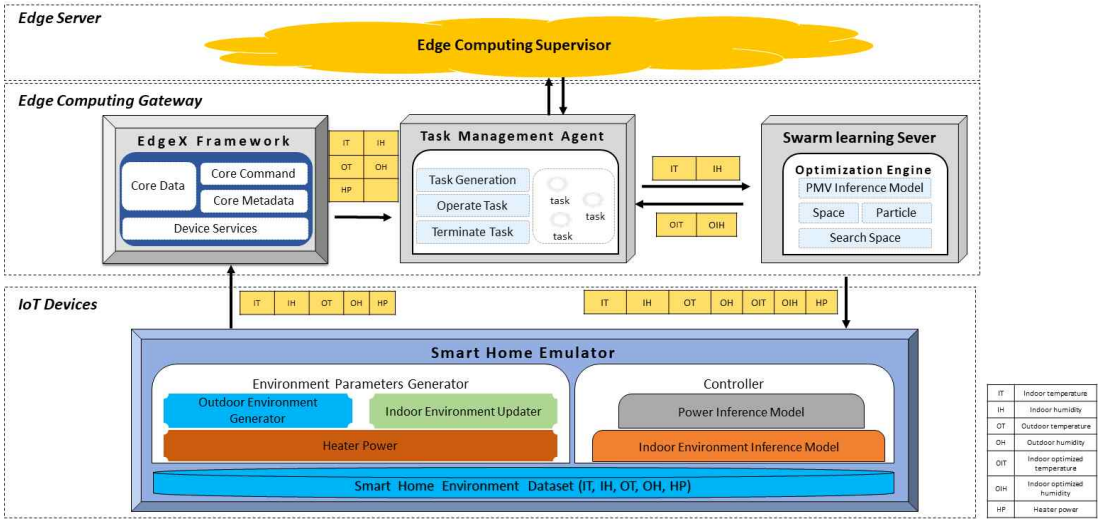


그림 5.3 PSO 알고리즘 기반 스마트 홈 에지 지능 실험 환경 구성

실험은 그림 5.4와 같은 순서를 따라 진행된다. 먼저 실험에 필요한 서비스, 컨트롤러 그리고 작업을 생성한다. 등록된 정보에 근거하여 최적화 작업에 관한 객체가 웹 페이지에 생성된다. 작업 객체를 생성하기 위하여 최적화 작업에 관련된 에지 게이트웨이, 사물인터넷 디바이스, 서비스 그리고 컨트롤러 정보를 데이터베이스로부

터 수집한다. 작업을 에지 게이트웨이에 배포할 수 있다. 데이터베이스로부터 수집된 정보들을 근거로 최적화 작업에 필요한 데이터들을 수집하는 명령을 생성한다. 그리고 수집된 데이터를 최적화 서비스에 입력하여 최적화된 출력을 얻도록 명령어를 생성한다. 마지막으로 출력된 최적화 데이터를 컨트롤러로 입력하는 명령어를 생성한다. 최적화 작업 정보와 생성된 명령어들은 에지 게이트웨이에 배포되고 반복적으로 호출하여 실행한다. 최적화 작업이 실행되는 과정에서 수집된 데이터 및 최적화 파라미터는 데이터베이스로 저장한다. 사용자는 가시화된 데이터에 근거하여 최적화 작업의 적용상태를 모니터링 할 수 있다.

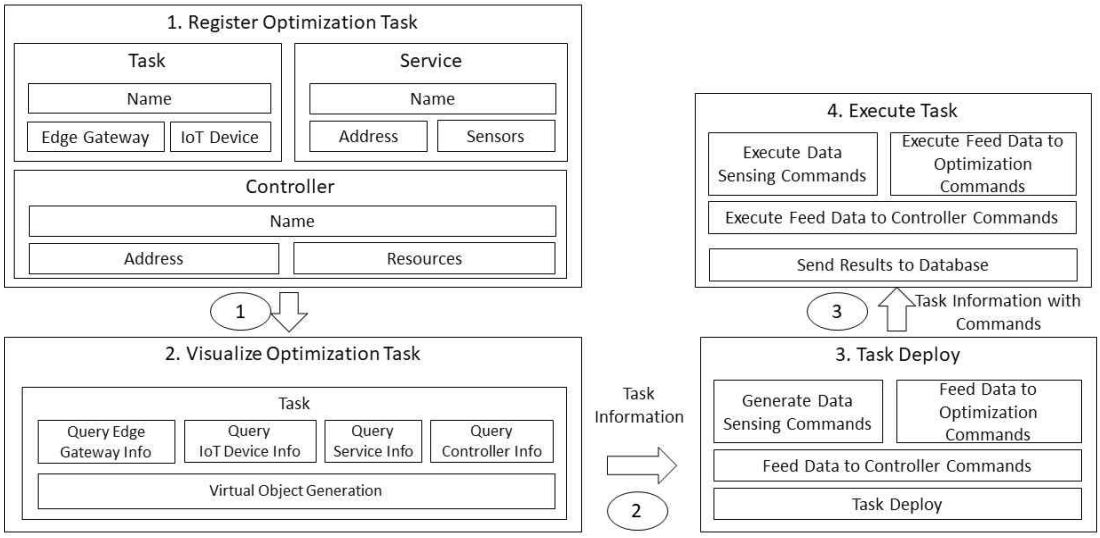
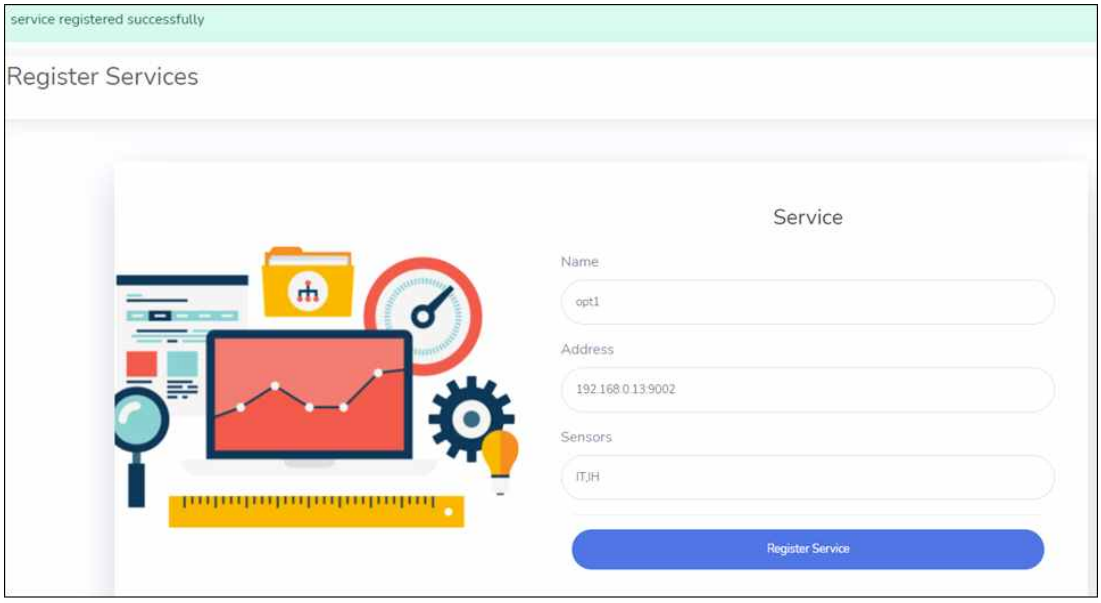


그림 5.4 PSO 알고리즘을 이용한 PMV 최적화 실험 과정

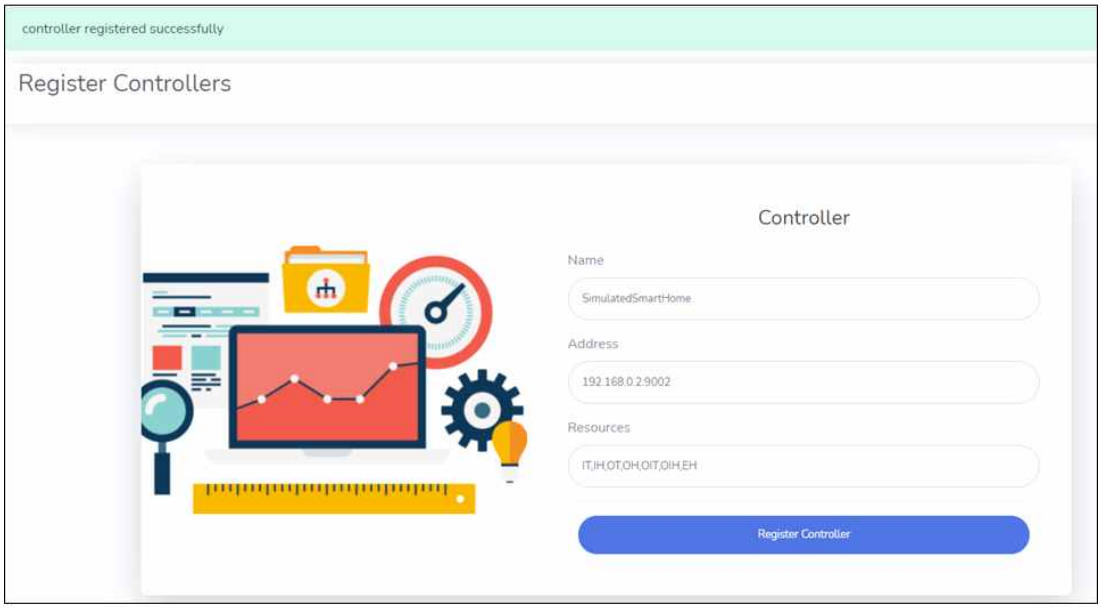
알고리즘의 성능을 테스트 하기 위하여 다음과 같은 사전 준비를 한다. 그림 5.5는 PSO 알고리즘을 서비스와 컨트롤러를 ECS에 등록하는 화면이다. 그중 그림 a는 최적화 서비스 등록화면이다. 서비스 이름은 “opt1”, 접속 주소는 “192.168.0.13:9002”, 그리고 서비스에 필요한 센서 데이터 정보는 IT, IH로 지정한다.

그림 b는 컨트롤러 등록화면이다. 컨트롤러 이름은 “SimulatedSmrtHome”이고 접속 주소는 “192.168.0.2:9002” 이고 필요한 센서 데이터 정보는 실내 온도 (IT), 실내 습도 (IH), 최적화 실내 온도 (OIT), 최적화 실내 습도 (OIH), 실외 온도 (OT), 실외 습도 (OH), 히터 전력 (HP)로 지정한다. 왼쪽 위의 메시지를 통하여 최적화 서

비스와 컨트롤러가 정상적으로 등록되었음을 확인할 수 있다.



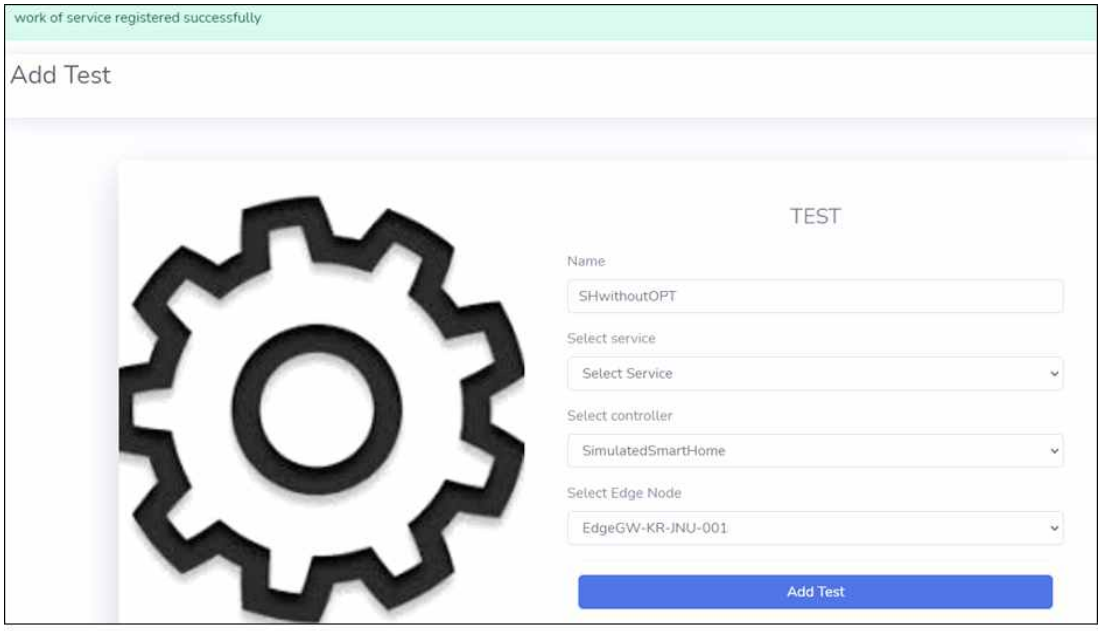
(a) PMV 최적화 서비스 등록화면



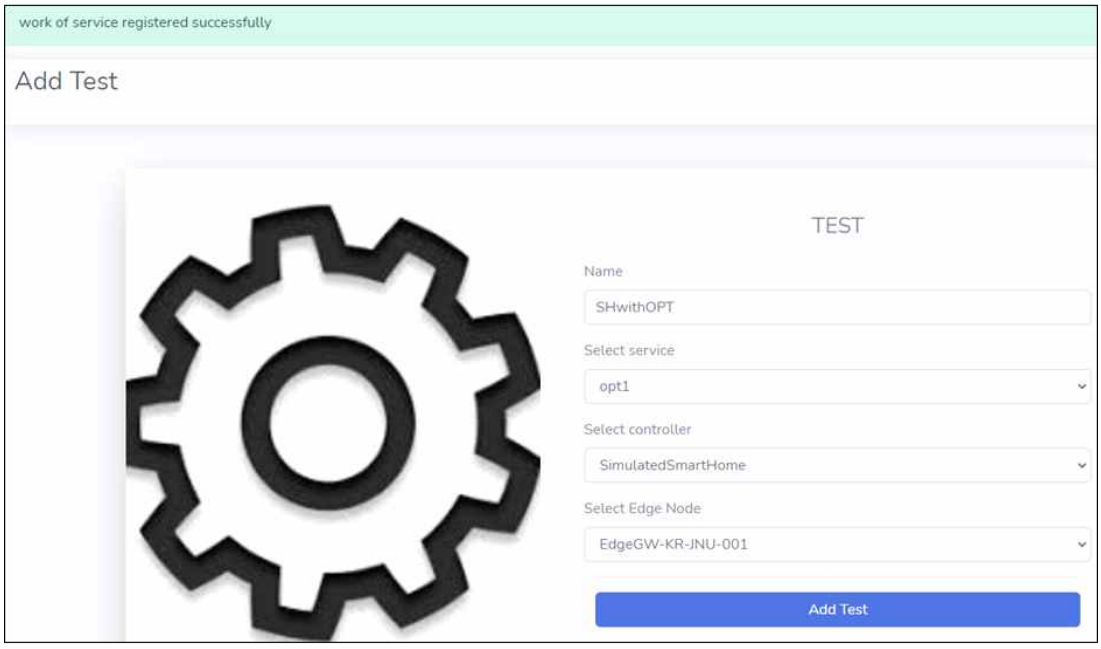
(b) 컨트롤러 등록화면

그림 5.5 PSO 알고리즘 서비스와 컨트롤러 등록화면

본고에서 제안하는 PMV 기반의 PSO 최적화 알고리즘의 성능을 비교하기 위하여 그림 5.6와 같이 최적화 알고리즘을 사용하는 테스트와 사용하지 않는 테스트 두 개의 테스트를 등록한다.



(a) PMV 최적화 미적용



(b) PMV 최적화 적용

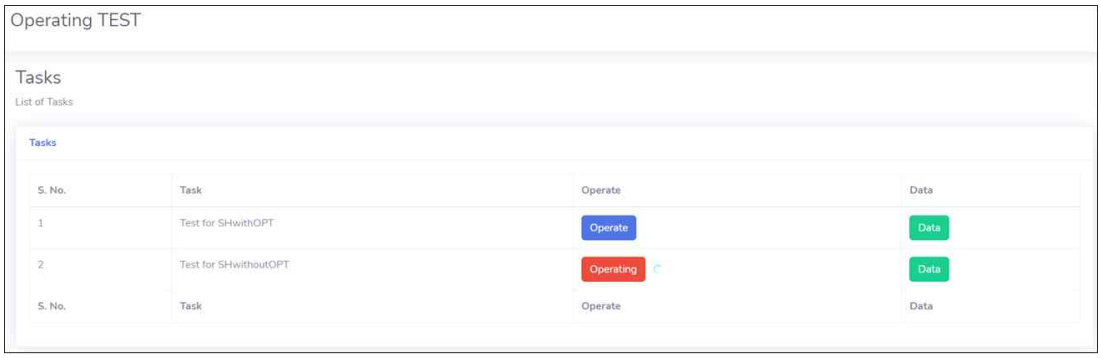
그림 5.6 PSO 기반 PMV 최적화 실험 등록화면

그림 a는 최적화 서비스를 사용하지 않도록 등록된 테스트이다. 이름은 “SHwithoutOPT” 로 지정하였고 서비스 선택사항에서 아무것도 선택하지 않았다. 그리고 컨트롤러는 위에서 등록한 것을 선택한다. 마지막으로 테스트가 실행할 에지

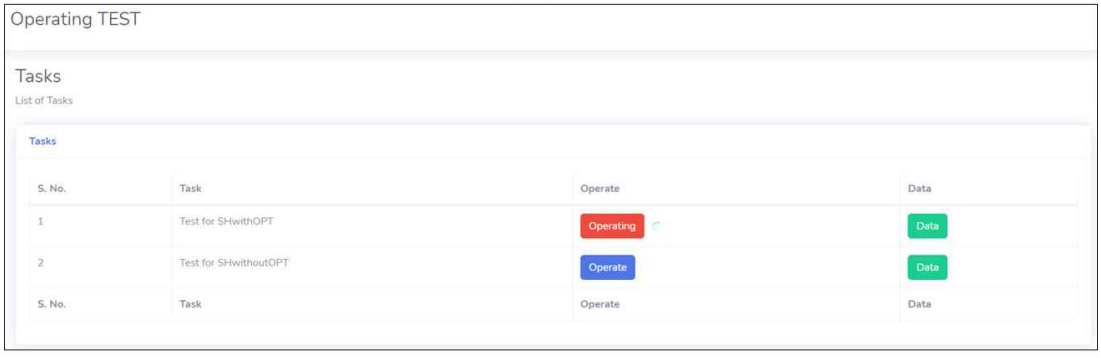
게이트웨이는 “EdgeGW-KR-JNU-001”를 선택한다.

그림 b는 최적화 서비스를 사용하는 테스트이다. 이름은 “SHwithOPT”로 등록하였고, 서비스는 “opt1”을 선택한다. 그리고 나머지는 위의 테스트와 같은 정보를 지정한다. 이로 인하여 최적화 알고리즘을 테스트하는 준비를 완료한다.

최적화 기법을 테스트하는 화면은 그림 5.7과 같다. 테이블에 최적화 기법을 사용하는 작업과 사용하지 않는 작업 두 개의 테스트가 표현되고 있다. 작업들은 작업(task), 실행 (operate), 데이터 컬럼으로 구분하여 표현한다. 작업 컬럼은 테스트 이름을 나타내고 있다. 실행 컬럼은 버튼으로서 클릭하여 작업을 실행하고 종료할 수 있다. 그림 a는 최적화 기법을 스마트 홈에 적용하지 않는 테스트를 실행하는 상태를 나타내고 그림 b는 최적화 기법을 스마트 홈에 적용하는 테스트를 실행하는 상태를 나타낸다. 데이터 컬럼은 테스트에서 수집한 데이터를 표현하는 화면으로 이동하는 링크 버튼이다.



(a) PMV 최적화 미적용 실험



(b) PMV 최적화 적용 실험

그림 5.7 PSO 기반 PMV 최적화 실험 실행화면

그림 5.8은 최적화 기법을 스마트 홈에 적용하는 테스트에서 수집한 데이터를 시

각화한 결과이다.

(a) 실내 온도 (b) 실내 습도

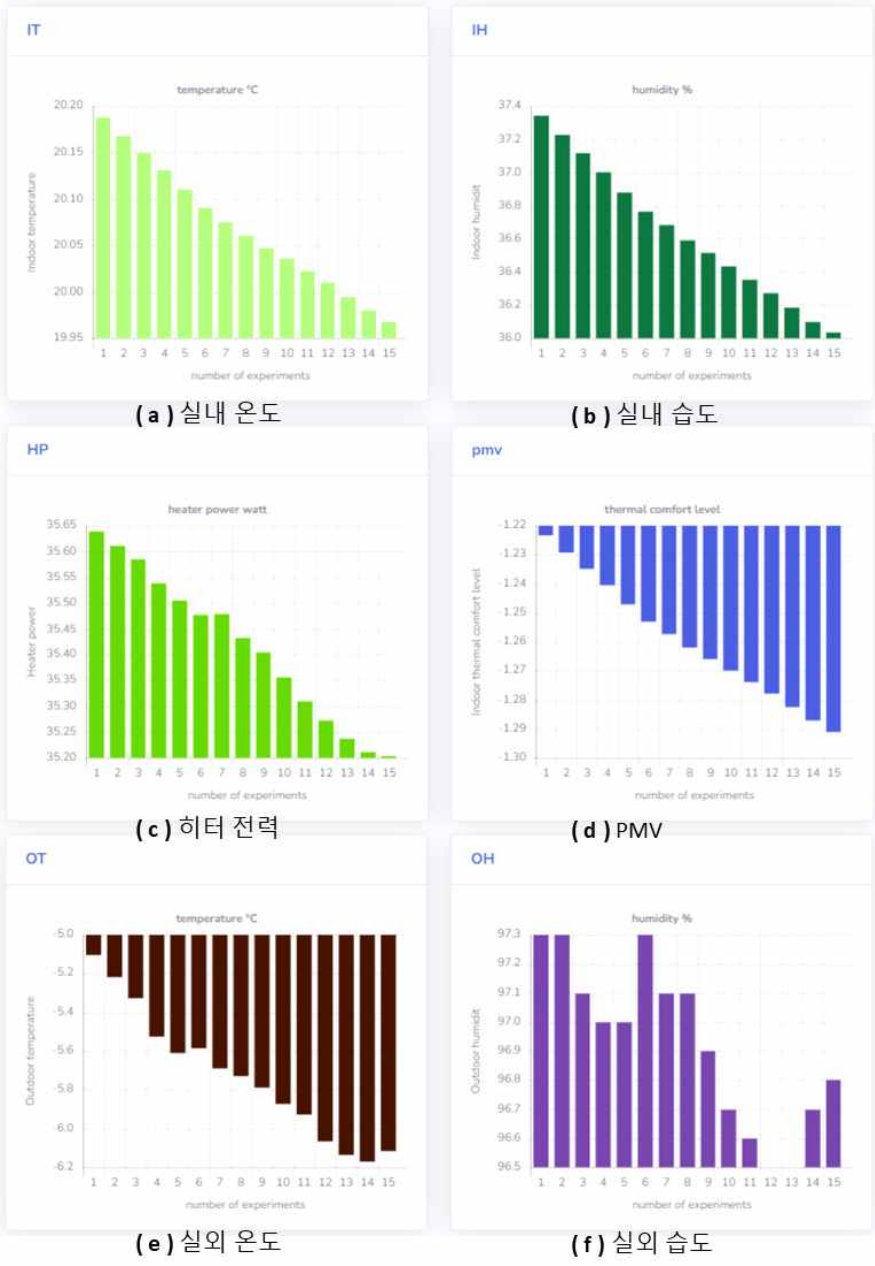
(c) 히터 전력 (d) 최적화 실내 온도

(e) 실외 온도 (f) 실외 습도

(g) 최적화 실내 습도 (h) PMV

그림 5.8 스마트 홈에서 PMV 최적화 적용 실험 결과

왼쪽 위에는 테스트 이름이 표시되고 있다. 실내 온도 (IT), 실내 습도 (IH), 실외 온도 (OT), 실외 습도 (OH), 히터 전력 (HP), 최적화 실내 온도 (opt_IT), 최적화 실내 습도 (opt_IH), 열 쾌적 레벨 (PMV) 등의 데이터가 막대기 도표 형태로 표현되고 있다. X좌표는 회 차 정보를 나타내고 Y좌표는 데이터 값 정보를 표현하고 있다. 총 15회 차의 정보를 보여 주고 있다.



(a) 실내 온도 (b) 실내 습도

(c) 히터 전력 (d) PMV

(e) 실외 온도 (f) 실외 습도

그림 5.9 스마트 홈에서 PMV 최적화 미적용 실험 결과

그림 5.9는 최적화 기법을 스마트 홈에 적용하지 않는 테스트에서 수집한 데이터를 시각화한 결과이다. 실내 온도 (IT), 실내 습도 (IH), 실외 온도 (OT), 실외 습도 (OH), 히터 전력 (HP), 열 쾌적 레벨 (PMV) 등의 데이터가 막대기 도표 형태로 표현되고 있다. 위의 테스트와 마찬가지로 X좌표는 회 차 정보를 나타내고 Y좌표는 데이터 값 정보를 표현하고 있다. 실내 온도는 19에서 21사이의 값을 보여주고 있고 실내 습도는 36에서 38사이의 값을 보여주고 있다. 실외 온도는 -6에서 -5사이의 값을 보여주고 실외 습도는 96에서 98사이의 값을 나타내고 있다. 히터 전력은 35에서 36사이의 값을 보여주고 있으며 PMV값은 -1.3에서 -1.2사이의 값을 나타내고 있다. 각각의 도표를 선명하게 구별할 수 있도록 수치를 표시하는 막대기를 서로 다른 색상으로 표현한다.

3. 제안한 PMV 최적화 메커니즘 성능 분석

그림 5.10은 스마트 홈 에뮬레이터에 최적화 기법을 적용한 후 측정한 실내 온도 데이터 (OIT)의 변화와 최적화 기법을 적용하지 않는 실내 온도 (IT)의 변화를 비교하는 도표이다.

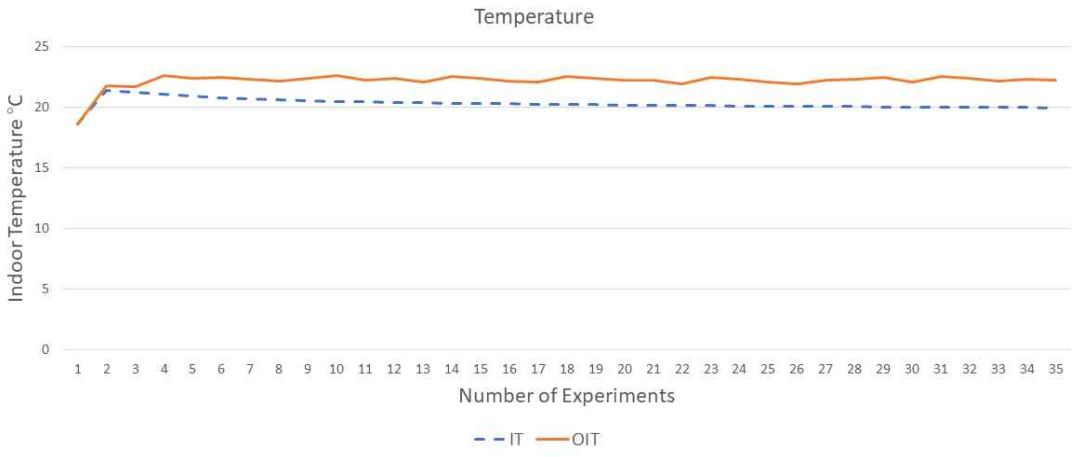


그림 5.10 PMV 최적화 적용 여부에 따른 실내 온도 수집 결과 비교

총 35회의 걸쳐서 수집한 데이터의 값을 도표로 나타냈다. 최적화 기법을 적용한

실내 온도는 실선으로 표현하였고 적용하지 않는 실내 온도 값은 점선으로 표현하여 구별한다. 최적화 기법을 적용하지 않은 실내 온도는 20이하에서 시작해서 점점 20의 값으로 접근하여 가고 있었다. 반대로 최적화 기법을 적용한 실내온도는 시작 온도를 제외하고 모두 22 좌우의 값으로 되어있었다. 최적화 기법을 사용한 스마트 홈 에뮬레이터는 최적화 기법을 적용하지 않을 때보다 실내 온도를 높은 값으로 유지하고 있는 것을 알 수 있다.

그림 5.11은 최적화 기법을 적용한 스마트 홈 에뮬레이터의 실내온도 데이터와 최적화 기법을 적용하지 않는 실내 온도 데이터를 통계한 결과이다. 각각 데이터의 최저 (min), 최고 (max), 평균 (avg), 표준 편차 (standard deviation) 값을 통계한다. 수치를 막대기 도표를 이용하여 표현하였고 구분하기 위하여 서로 다른 모양의 막대기로 표현한다. 최대치와 평균값에서 최적화 기법을 적용한 스마트 홈 에뮬레이터가 실내온도를 상대적으로 조금 높게 유지하고 있었다. 통계한 표준 편차 수치를 보았을 때 최적화 기법을 사용하지 않았을 때 스마트 홈 에뮬레이터가 실내 온도를 더욱 안정적으로 유지하고 있음을 알 수 있다.

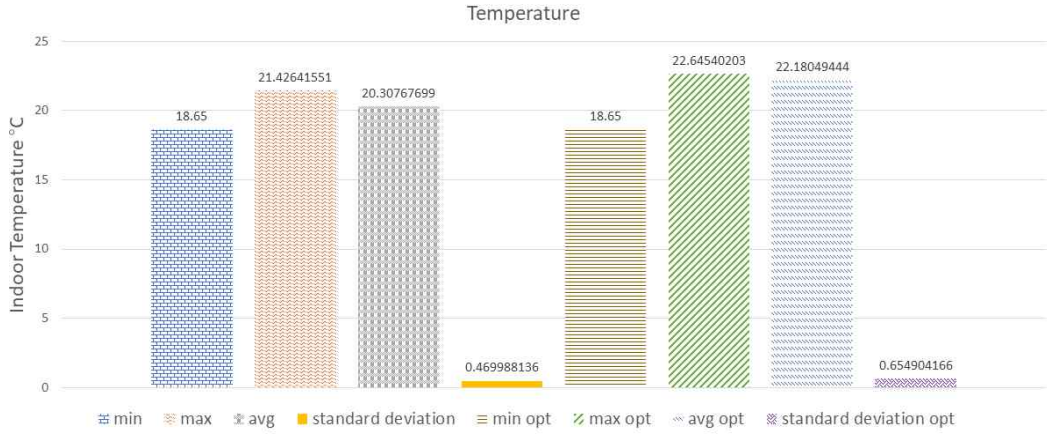


그림 5.11 PMV 최적화 적용 여부에 따른 실내 온도 수집 결과 통계

그림 5.12는 최적화 기법을 적용한 스마트 홈 에뮬레이터의 실내 습도 데이터 (OIH) 와 최적화 기법을 적용하지 않는 실내 습도 (IH) 데이터를 비교하는 도표이다. 실내 습도 데이터도 마찬가지로 총 35회의 데이터를 수집한다. 최적화 기법을 적용한 실내 습도는 실선으로 표현하였고, 적용하지 않는 실내 습도 값은 점선으로 표현하여 구별한다. 최적화 기법을 적용하지 않은 실내 습도는 40 이상의 값으로 시작해

서 점점 40 이하의 값으로 유지하고 있다. 반대로 최적화 기법을 적용한 실내습도는 시작을 제외하고 50 좌우의 값으로 되어있었다. 최적화 기법을 사용한 스마트 홈 에뮬레이터는 최적화 기법을 적용하지 않을 때보다 실내 습도를 같은 값으로 유지하고자 하는 것을 알 수 있다.

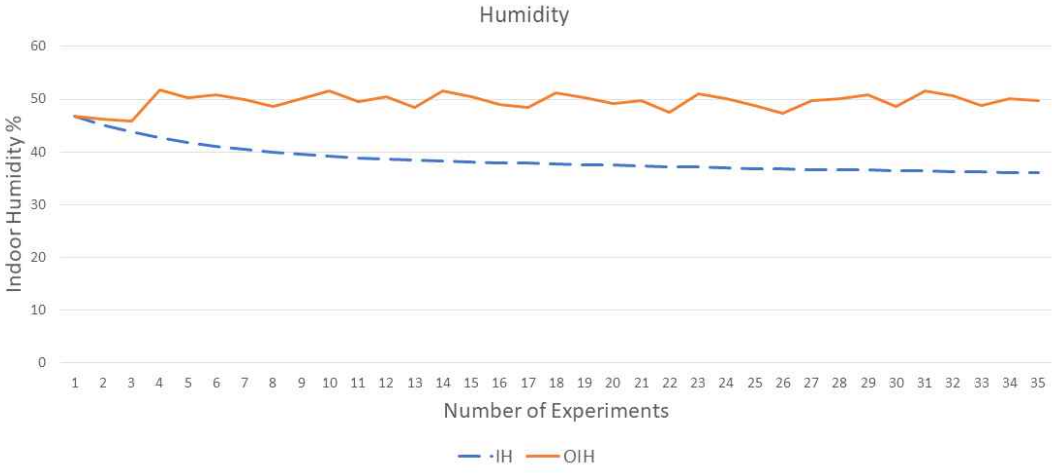


그림 5.12 PMV 최적화 적용 여부에 따른 실내 습도 결과 비교

그림 5.13은 최적화 기법을 적용한 실내 습도 데이터와 최적화 기법을 적용하지 않는 실내 습도 데이터를 통계한 결과이다.

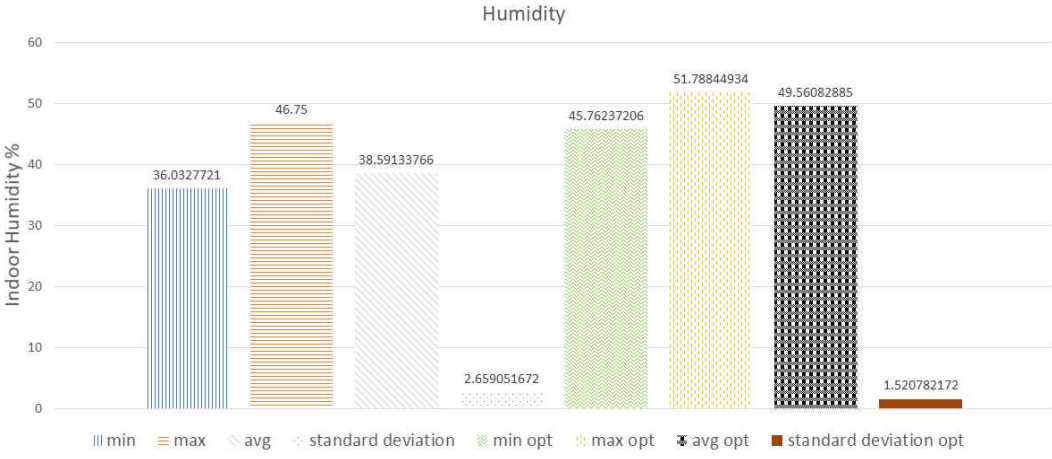


그림 5.13 PMV 최적화 적용 여부에 따른 실내 습도 결과 통계

데이터를 더욱 잘 이해하기 위하여 각각 데이터의 최저 (min), 최고 (max), 평균

(avg), 표준 편차 (standard deviation) 값을 통계한다. 수치를 막대기 도표를 이용하여 표현하였고 각각의 값을 구별하기 위하여 서로 다른 모양의 막대기로 나타냈다. 최저, 최대, 평균 등 모든 면에서 최적화 기법을 적용한 스마트 홈 에뮬레이터가 실내 습도를 상대적으로 높은 값으로 유지하고 있었다. 통계한 표준 편차 수치를 보았을 때 최적화 기법을 사용하지 않았을 때 스마트 홈 에뮬레이터의 실내 습도가 더욱 많이 변화되었다.

그림 5.14는 최적화 기법을 적용한 열 쾌적 레벨 (OPMV) 와 최적화 기법을 적용하지 않는 열 쾌적 레벨 (PMV)를 비교하는 도표이다. 최적화 기법을 적용한 열 쾌적 레벨은 점선으로 표현하였고 적용하지 않는 열 쾌적 레벨은 실선으로 표현하여 구별한다. 최적화 기법을 적용하였을 때 스마트 홈 에뮬레이터의 열 쾌적 레벨은 -0.6에서 -0.8사이를 유지하고 있는 반면 적용하지 않았을 때는 -1.2와 -1.4사이로 점차 변화하여 갔다. 최적화 기법을 적용하였을 때 실내 온도와 습도가 열 쾌적 레벨을 최적화 기법을 적용하지 않았을 때보다 따뜻하게 유지하고 있음을 알 수 있다. 총 35회에 수집된 데이터 평균을 보았을 때도 최적화 기법을 적용하였을 때 -0.7의 수치로 -1.2보다 더욱 따뜻한 열 쾌적 레벨에 유지 되고 있음을 알 수 있다.

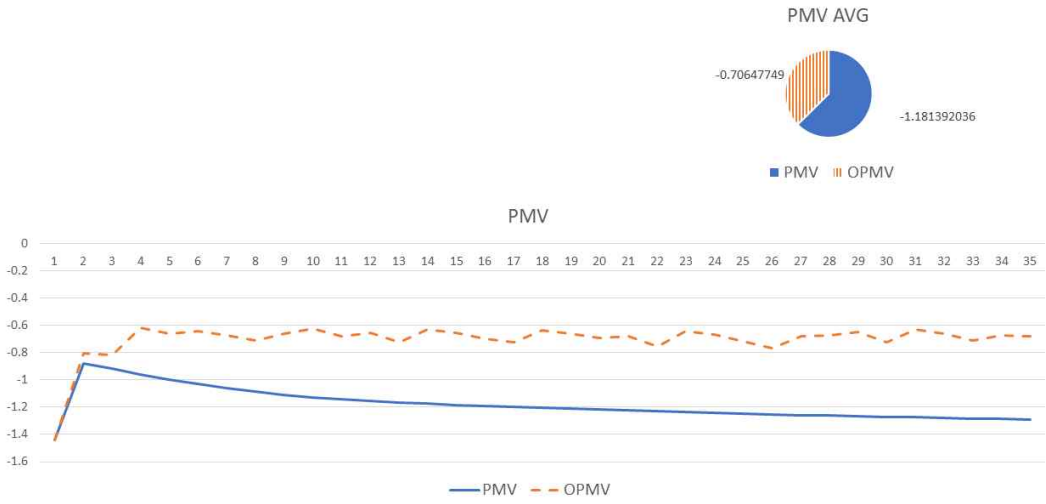


그림 5.14 PMV 최적화 적용 여부에 따른 열 쾌적도 결과 비교

그림 5.15는 최적화 기법을 적용한 히터 전력 (OHP)와 최적화 기법을 적용하지 않는 히터 전력 (HP)를 비교하는 도표이다. 최적화 기법을 적용하였을 때와 최적화

기법을 적용하지 않았을 때 거의 비슷한 에너지를 사용하고 있는 것을 알 수 있다. 수치를 막대기 도표를 이용하여 표현하였고 최적화 기법을 적용한 히터 전력의 값과 적용하지 않은 히터 전력의 값을 구별하기 위하여 서로 다른 모양의 막대기로 나타냈다. 실험 시작을 제외하고 모두 35에 가까운 값을 나타내고 있다. 총 35회에 수집된 데이터 평균을 보면 최적화 기법을 적용하였을 때 36,09 최적화 기법을 사용하지 않았을 때 35.93로 최적화 기법을 적용하였을 때 아주 미세한 차이로 에너지를 많이 사용하고 있다. 결과적으로 보았을 때 최적화 기법을 사용하였을 때 열 쾌적 레벨을 더 따뜻한 레벨로 유지하고 에너지 소비는 거의 변화가 없는 것을 알 수 있다.

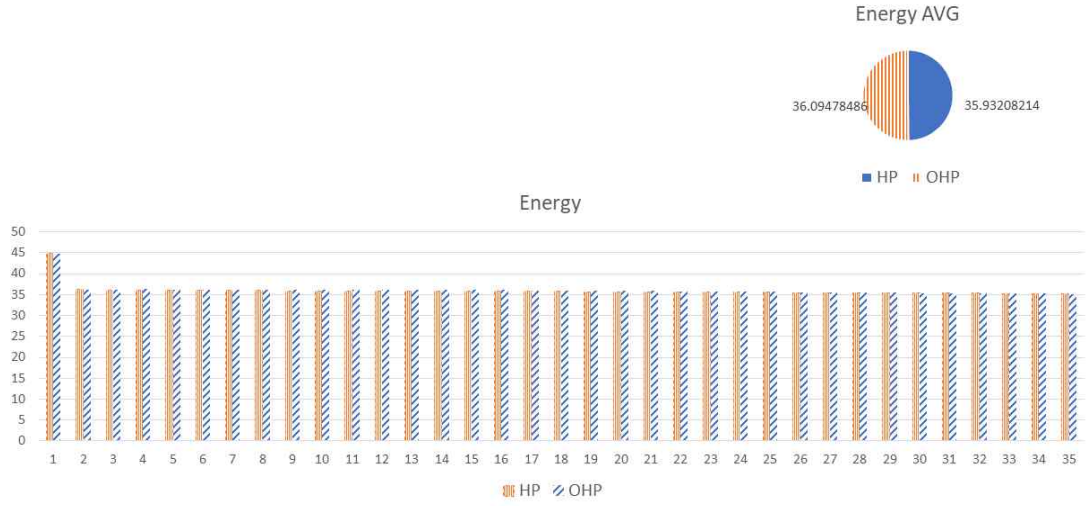


그림 5.15 PMV 최적화 적용 여부에 따른 히터 전력 결과 비교

VI. 결론

본 논문에서 먼저 분산 디지털 트윈 에지 컴퓨팅 환경을 구축하기 위하여 물리적인 에지 컴퓨팅을 가상화할 수 있도록 에지 컴퓨팅 슈퍼바이저, 에지 게이트웨이 관리 서포터를 구현하였다. 디지털 트윈 에지 컴퓨팅 환경을 통하여 사용자는 GUI 기반으로 에지 컴퓨팅의 구성자, 즉 에지 게이트웨이, 사물인터넷 디바이스의 상태를 실시간으로 모니터링할 수 있을 뿐만 아니라, 작업을 생성하여 지정 에지 게이트웨이에 배포할 수 있다. 분산 디지털 트윈 에지 컴퓨팅 환경을 이용하여 환경 데이터 수집 및 PSO 최적화 기법을 통한 최적의 온도와 습도의 결과를 도출할 수 있다. 그리고 에지 게이트웨이 스마트 홈 에뮬레이터를 컨트롤러로 등록하여 PSO 최적화 기법의 성능을 평가하는 시뮬레이션도 할 수 있다.

다음으로 기존 스웜학습 전략으로부터 개선한 모델 훈련 전략을 도입하여 에지 컴퓨팅 네트워크에 참여한 에지 게이트웨이들을 이용하여 PMV 모델을 훈련한다. 기존 연합/스웜 학습과 같이 중앙서버에서 모든 모델을 수집하여 글로벌 모델을 업데이트하지 않고 모든 에지 게이트웨이가 모델 훈련에 순서대로 참여하여 로컬 데이터를 이용하여 모델을 훈련하기 때문에 참여자의 개수에 관계없이 모델 훈련에 필요한 컴퓨팅 부담은 줄어든다. 그리고 에지 컴퓨팅 네트워크에 저장된 모든 데이터를 활용하여 모델의 성능을 향상하고 로컬 데이터를 중앙서버로 전달하지 않아서 데이터의 개인정보에 대한 안전도 강화한다.

그리고 분산 디지털 트윈 에지 컴퓨팅 환경을 통하여 IoTivity와 HTTP를 구현한 사물인터넷 디바이스에서 데이터를 수집하는데 소요하는 시간을 측정하였고 에지 컴퓨팅 네트워크에서 제안한 스웜 학습을 통하여 모델을 학습한 성능을 측정하였으며 마지막으로 PSO 최적화 알고리즘의 성능도 측정하였다. 데이터 수집 작업 시간은 10회의 테스트를 진행하였고 IoTivity 프레임워크가 최고 값 1500을 빼고는 두 통신 방법이 모두 1초미만의 작업 시간을 보여주었다. IoTivity는 오픈소스 프레임워크로서 사물인터넷 디바이스를 자원으로 등록하여 로컬 네트워크에서 IoTivity 클라이

언트로부터 참조할 수 있도록 한다. HTTP를 이용하여 사물인터넷 디바이스와 직접 통신을 하는 데 비해 IoTivity 클라이언트에서 같은 네트워크에 있는 IoTivity서버에 등록한 자원을 발견하도록 실행하여 더욱 많은 지연 시간이 발생하였다.

마지막으로 개선한 스웜학습 전략이 대부분의 실험에서 더 좋은 성능을 보여주고 있다. 반복 회수를 각각 1, 10, 100으로 설정하여 실험을 진행하였지만 기존 연합/스웜 학습 전략을 통하여 훈련한 모델의 성능은 0.3 좌우가 최고의 성능이지만 본고에서 개선한 스웜학습 전략은 1회에 훈련한 모델이 0.2 좌우의 성능을 보여주고 있다. 하지만 본고에서 개선한 스웜학습 전략은 하나의 모델을 에지 네트워크 컴퓨팅에 있는 에지 게이트웨이들이 순서대로 업데이트하여 학습하는 방식을 사용하여 병렬로 학습하는 기존 연합/스웜 학습에 비해 많은 훈련시간을 사용한다.

향후 디지털 에지 컴퓨팅 환경 서비스에 사용자 등록 기능을 추가하여 허용한 사용자만이 에지 컴퓨팅 네트워크에 접근할 수 있도록 한다. 사용자 등록 기능을 통하여 악의적인 사용자로부터의 시스템 접근을 차단하여 데이터의 안전을 강화한다. 목적 함수 기반의 최적화 기법을 사용하여 스마트 홈의 실내 환경을 자동으로 최적의 상태로 유지함과 동시에 에너지 소비도 고려하는 방안을 연구한다. 그리고 충분한 에지 게이트웨이 자원을 등록하여 군집 강화 학습의 성능을 측정할 필요가 있다.

참고문헌

[1] LeCun Y, Bengio Y, Hinton G. Deep learning. nature. 2015 May;521(7553):436-44.

[2] Deng L, Yu D. Deep learning: methods and applications. Foundations and trends in signal processing. 2014 Jun 30;7(3-4):197-387.

[3] Networking CV. Cisco global cloud index: Forecast and methodology, 2016-2021. White paper. Cisco Public, San Jose. 2016.

[4] Chen, Jiasi, and Xukan Ran. "Deep Learning With Edge Computing: A Review." Proc. IEEE 107.8 (2019): 1655-1674.

[5] Ray PP. A survey of IoT cloud platforms. Future Computing and Informatics Journal. 2016 Dec 1;1(1-2):35-46.

[6] Satyanarayanan, Mahadev. "The emergence of edge computing." Computer 50.1 (2017): 30-39.

[7] Jeans, David. "Related's Hudson Yards: Smart City or Surveillance City?." The country's largest development has officially arrived, but so too has the debate over Related's plans for user data." The Real Deal (2019).

[8] Satyanarayanan, Mahadev, et al. "The case for vm-based cloudlets in mobile computing." IEEE pervasive Computing 8.4 (2009): 14-23.

[9] Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: Vision and challenges. IEEE internet of things journal. 2016 Jun 9;3(5):637-46.

[10] Naveen, Soumyalatha, and Manjunath R. Kounte. "Key technologies and challenges in IoT edge computing." 2019 Third international conference on I-SMAC (IoT in social, mobile, analytics and cloud)(I-SMAC). IEEE, 2019.

[11] Wang X, Han Y, Wang C, Zhao Q, Chen X, Chen M. In-edge ai: Intelligentizing mobile edge computing, caching and communication by

federated learning. IEEE Network. 2019 Jul 24;33(5):156-65.

[12] Li E, Zhou Z, Chen X. Edge intelligence: On-demand deep learning model co-inference with device-edge synergy. InProceedings of the 2018 Workshop on Mobile Edge Communications 2018 Aug 7 (pp. 31-36).

[13] Ananthanarayanan G, Bahl P, Bodík P, Chintalapudi K, Philipose M, Ravindranath L, Sinha S. Real-time video analytics: The killer app for edge computing. computer. 2017 Oct 3;50(10):58-67.

[14] Ha K, Chen Z, Hu W, Richter W, Pillai P, Satyanarayanan M. Towards wearable cognitive assistance. InProceedings of the 12th annual international conference on Mobile systems, applications, and services 2014 Jun 2 (pp. 68-81).

[15] Cao J, Xu L, Abdallah R, Shi W. EdgeOS_H: a home operating system for internet of everything. In2017 IEEE 37th international conference on distributed computing systems (ICDCS) 2017 Jun 5 (pp. 1756-1764). IEEE.

[16] Li L, Ota K, Dong M. Deep learning for smart industry: Efficient manufacture inspection system with fog computing. IEEE Transactions on Industrial Informatics. 2018 Jun 1;14(10):4665-73.

[17] Svozil D, Kvasnicka V, Pospichal J. Introduction to multi-layer feed-forward neural networks. Chemometrics and intelligent laboratory systems. 1997 Nov 1;39(1):43-62.

[18] Dean J, Corrado G, Monga R, Chen K, Devin M, Mao M, Ranzato MA, Senior A, Tucker P, Yang K, Le Q. Large scale distributed deep networks. Advances in neural information processing systems. 2012;25:1223-31.

[19] Yang Q, Liu Y, Cheng Y, Kang Y, Chen T, Yu H. Federated learning. Synthesis Lectures on Artificial Intelligence and Machine Learning. 2019 Dec 19;13(3):1-207.

[20] McMahan, Brendan, et al. “Communication-efficient learning of deep networks from decentralized data.“ Artificial intelligence and statistics. PMLR, 2017.

[21] Warnat-Herresthal, Stefanie, et al. “Swarm Learning for decentralized and

confidential clinical machine learning.“ Nature 594.7862 (2021): 265-270.

[22] L. Farhan, R. Kharel, O. Kaiwartya, M. Quiroz, A. E. Alissa, and M. Abdulsalam, A Concise Review on Internet of Things (IoT) - Problems, Challenges and Opportunities. 2018. doi: 10.1109/CSNDSP.2018.8471762.

[23] A review of Internet of Things: qualifying technologies and boundless horizon ¦ SpringerLink.” https://link.springer.com/article/10.1007/s40860-020-00127-w (accessed Nov. 20, 2021).

[24] “(4) (PDF) A survey on application layer protocols for the Internet of Things.” https://www.researchgate.net/publication/303192188_A_survey_on_application_layer_protocols_for_the_Internet_of_Things (accessed Nov. 16, 2021).

[25] “Perci: Pervasive Service Interaction with the Internet of Things ¦ IEEE Journals & Magazine ¦ IEEE Xplore.” https://ieeexplore.ieee.org/document/5262929 (accessed Nov. 20, 2021).

[26] C. M. Fernández, M. D. Rodríguez, and B. R. Muñoz, “An Edge Computing Architecture in the Internet of Things,” 2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC), 2018, doi: 10.1109/isorc.2018.00021.

[27] K. Dolui and S. K. Datta, Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing. 2017, p. 6. doi: 10.1109/GIOTS.2017.8016213.

[28] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, “Zenith: Utility-Aware Resource Allocation for Edge Computing,” in 2017 IEEE International Conference on Edge Computing (EDGE), Honolulu, HI, USA, Jun. 2017, pp. 47-54. doi: 10.1109/IEEE.EDGE.2017.15.

[29] Y. Ai, M. Peng, and K. Zhang, “Edge computing technologies for Internet of Things: a primer,” Digital Communications and Networks, vol. 4, no. 2, pp. 77-86, Apr. 2018, doi: 10.1016/j.dcan.2017.07.001.

[30] “(2) (PDF) Fog Computing and the Internet of Things: A Review.”

https://www.researchgate.net/publication/324280213_Fog_Computing_and_the_Internet_of_Things_A_Review (accessed Nov. 17, 2021).

[31] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," p. 3.

[32] M. Saad, "Fog Computing and Its Role in the Internet of Things: Concept, Security and Privacy Issues," 2018, doi: 10.5120/IJCA2018916829.

[33] R. Morabito, R. Petrolo, V. Loscrì, and N. Mitton, "LEGIoT: A Lightweight Edge Gateway for the Internet of Things," Future Generation Computer Systems, vol. 81, pp. 1–15, Apr. 2018, doi: 10.1016/j.future.2017.10.011.

[34] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan, "Osmotic Computing: A New Paradigm for Edge/Cloud Integration," IEEE Cloud Computing, vol. 3, no. 6, pp. 76–83, Nov. 2016, doi: 10.1109/MCC.2016.124.

[35] M. Satyanarayanan, "The Emergence of Edge Computing," Computer, vol. 50, no. 1, pp. 30–39, Jan. 2017, doi: 10.1109/MC.2017.9.

[36] "Introduction - EdgeX Foundry Documentation." https://docs.edgexfoundry.org/2.1/ (accessed Nov. 20, 2021).

[37] "An Extensible Edge Computing Architecture: Definition, Requirements and Enablers." https://www-live.dfki.de/web/forschung/projekte-publikationen/publikationen-filter/publikation/9381 (accessed Nov. 20, 2021).

[38] E. Varga, B. Blagojević, and D. Mijić, "Composing Internet of Things Platforms in Smart Grid," MATEC Web of Conferences, vol. 208, 2018, doi: 10.1051/matecconf/201820802007.

[39] G. Klas, "Fog Computing and Mobile Edge Cloud Gain Momentum Open Fog Consortium, ETSI MEC and Cloudlets," 2015. https://www.semanticscholar.org/paper/Fog-Computing-and-Mobile-Edge-Cloud-Gain-Momentum-Klas/147b98a16391c658cfb2e401f340b991269711df1 (accessed Nov. 20, 2021).

[40] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for

VM-based Cloudlets in Mobile Computing," p. 9.

[41] A. Rasheed, P. H. J. Chong, I. W.-H. Ho, X. J. Li, and W. Liu, "An Overview of Mobile Edge Computing: Architecture, Technology and Direction," KSII Transactions on Internet and Information Systems (TIIS), vol. 13, no. 10, pp. 4849-4864, 2019, doi: 10.3837/tiis.2019.10.002.

[42] "Azure IoT - Internet of Things Platform ¦ Microsoft Azure." https://azure.microsoft.com/en-us/overview/iot/ (accessed Nov. 20, 2021).

[43] "Edgent Incubation Status - Apache Incubator." https://incubator.apache.org/projects/edgent.html (accessed Nov. 20, 2021).

[44] "CORD Platform ¦ Central Office Rearchitected as a Datacenter ¦ ONF," Open Networking Foundation. https://opennetworking.org/cord/ (accessed Nov. 20, 2021).

[45] S. K. Polu, "OAuth based Secured authentication mechanism for IoT Applications," p. 409, Jan. 2018.

[46] C.-H. Chen, M.-Y. Lin, and C. Liu, "Edge Computing Gateway of the Industrial Internet of Things Using Multiple Collaborative Microcontrollers," IEEE Network, 2018, doi: 10.1109/MNET.2018.1700146.

[47] "Azure IoT Central ¦ Microsoft Azure." https://azure.microsoft.com/en-us/services/iot-central/ (accessed Nov. 20, 2021).

[48] "CORD Archives," Open Networking Foundation. https://opennetworking.org/tag/cord/ (accessed Nov. 20, 2021).

[49] "Platform." https://www.edgexfoundry.org/software/platform/ (accessed Nov. 20, 2021).

[50] N. Dragoni et al., "Microservices: Yesterday, Today, and Tomorrow," in Present and Ulterior Software Engineering, M. Mazzara and B. Meyer, Eds. Cham: Springer International Publishing, 2017, pp. 195-216. doi: 10.1007/978-3-319-67425-4_12.

[51] P. D. Francesco, P. Lago, and I. Malavolta, "Migrating Towards Microservice Architectures: An Industrial Survey," 2018 IEEE International Conference on

Software Architecture (ICSA), 2018, doi: 10.1109/ICSA.2018.00012.

[52] S. Newman, Building microservices: designing fine-grained systems. 2015. Accessed: Nov. 20, 2021. [Online]. Available: http://public.eblib.com/choice/publicfullrecord.aspx?p=1938300

[53] C. Santana, B. de M. Alencar, and C. V. S. Prazeres, "Microservices: A Mapping Study for Internet of Things Solutions," 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), 2018, doi: 10.1109/NCA.2018.8548331.

[54] F. Liu, G. Tang, Y. Li, Z. Cai, X. Zhang, and T. Zhou, "A Survey on Edge Computing Systems and Tools," Proc. IEEE, vol. 107, no. 8, pp. 1537–1562, Aug. 2019, doi: 10.1109/JPROC.2019.2920341.

[55] A. Musaddiq, Y. B. Zikria, O. Hahm, H. Yu, A. Bashir, and S. Kim, "A Survey on Resource Management in IoT Operating Systems," IEEE Access, 2018, doi: 10.1109/ACCESS.2018.2808324.

[56] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing Offloading Latency for Digital Twin Edge Networks in 6G," IEEE Transactions on Vehicular Technology, vol. 69, no. 10, pp. 12240–12251, Oct. 2020, doi: 10.1109/TVT.2020.3018817.

[57] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Low-latency Federated Learning and Blockchain for Edge Association in Digital Twin empowered 6G Networks," IEEE Trans. Ind. Inf., vol. 17, no. 7, pp. 5098–5107, Jul. 2021, doi: 10.1109/TII.2020.3017668.

[58] K. Zhang, J. Cao, and Y. Zhang, "Adaptive Digital Twin and Multiagent Deep Reinforcement Learning for Vehicular Edge Computing and Networks," IEEE Transactions on Industrial Informatics, vol. 18, no. 2, pp. 1405–1413, Feb. 2022, doi: 10.1109/TII.2021.3088407.

[59] M. Savi and F. Olivadese, "Short-Term Energy Consumption Forecasting at the Edge: A Federated Learning Approach," IEEE Access, vol. 9, pp. 95949–95969, 2021, doi: 10.1109/ACCESS.2021.3094089.

[60] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence," IEEE Internet of Things Journal, vol. 7, no. 8, pp. 7457-7469, Aug. 2020, doi: 10.1109/JIOT.2020.2984887.

[61] S. Warnat-Herresthal et al., "Swarm Learning for decentralized and confidential clinical machine learning," Nature, vol. 594, no. 7862, pp. 265-270, Jun. 2021, doi: 10.1038/s41586-021-03583-3.

[62] Zhou, Zhi, et al. "Edge intelligence: Paving the last mile of artificial intelligence with edge computing." Proceedings of the IEEE 107.8 (2019): 1738-1762.

[63] Breiman L. Random forests. Machine learning. 2001 Oct;45(1):5-32.

[64] 0. Jiang, Zongmin, Yangming Guo, and Zhuqing Wang. "Digital twin to improve the virtual-real integration of industrial IoT." Journal of Industrial Information Integration 22 (2021): 100196.

[65] 1. Lim, Kendrik Yan Hong, Pai Zheng, and Chun-Hsien Chen. "A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives." Journal of Intelligent Manufacturing 31.6 (2020): 1313-1337.

[66] 2. Qi, Qinglin, and Fei Tao. "A smart manufacturing service system based on edge computing, fog computing, and cloud computing." IEEE Access 7 (2019): 86769-86777.

[67] 3. Huang, Huiyue, et al. "Digital Twin-driven online anomaly detection for an automation system based on edge intelligence." Journal of Manufacturing Systems 59 (2021): 138-150.

[68] 4. Mourtzis, Dimitris, et al. "A cloud-based approach for maintenance of machine tools and equipment based on shop-floor monitoring." Procedia Cirp 41 (2016): 655-660.

[69] 5. Kammerer, Klaus, et al. "Anomaly detections for manufacturing systems based on sensor data—insights into two challenging real-world production

settings.“ Sensors 19.24 (2019): 5370.

[70] 6. Lu, Yuqian, et al. “Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues.“ Robotics and Computer-Integrated Manufacturing 61 (2020): 101837.

[71] 7. El Saddik, Abdulmotaleb, et al. “Dtwins: a digital twins ecosystem for health and well-being.“ IEEE COMSOC MMTC Commun. Front 14 (2019): 39-43.

[72] 8. Costanzo, Alfio, et al. “Mobile cyber physical systems for health care: Functions, ambient ontology and e-diagnostics.“ 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2016.

[73] 9. Martinez-Velazquez, Roberto, Rogelio Gamez, and Abdulmotaleb El Saddik. “Cardio Twin: A Digital Twin of the human heart running on the edge.“ 2019 IEEE International Symposium on Medical Measurements and Applications (MeMeA). IEEE, 2019.

[74] Talbi, El-Ghazali. Metaheuristics: from design to implementation. Vol. 74. John Wiley & Sons, 2009.

[75] Makhadmeh, Sharif Naser, et al. “Optimization methods for power scheduling problems in smart home: Survey.“ Renewable and Sustainable Energy Reviews 115 (2019): 109362.

[76] Makhadmeh, Sharif Naser, et al. “Particle swarm optimization algorithm for power scheduling problem using smart battery.“ 2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT). IEEE, 2019.

[77] Ranjini, A., and B. S. E. Zoraida. “Intelligent residential energy management in smart grid.“ Indian Journal of Science and Technology 9.45 (2016).

[78] Lugo-Cordero, Hector M., et al. “Particle swarm optimization for load balancing in green smart homes.“ 2011 IEEE Congress of Evolutionary Computation (CEC). IEEE, 2011.

[79] Zhao, Zhuang, et al. “An optimal power scheduling method for demand response in home energy management system.“ IEEE transactions on smart

grid 4.3 (2013): 1391-1400.

[80] Nawaz, Fahad, et al. “An optimal Home energy management system based on time of use pricing scheme in smart grid.“ Int. J. Sci. Eng. Res 8 (2017): 882-894.

[81] Makhadmeh, Sharif Naser, et al. “Multi-objective power scheduling problem in smart homes using grey wolf optimiser.“ Journal of Ambient Intelligence and Humanized Computing 10.9 (2019): 3643-3667.

[82] Huang, Yantai, Lei Wang, and Qidi Wu. “A hybrid PSO-DE algorithm for smart home energy management.“ International Conference in Swarm Intelligence. Springer, Cham, 2014.

[83] Yi, Liu. “Study on an improved PSO algorithm and its application for solving function problem.“ Int. J. Smart Home 10.3 (2016): 51-62.

[84] Shakarami, Ali, et al. “A survey on the computation offloading approaches in mobile edge/cloud computing environment: a stochastic-based perspective.“ Journal of Grid Computing 18.4 (2020): 639-671.

[85] Kibria, Mirza Golam, et al. “Big data analytics, machine learning, and artificial intelligence in next-generation wireless networks.“ IEEE access 6 (2018): 32328-32338.

[86] Faul, Anita C. A concise introduction to machine learning. CRC Press, 2019.

[87] He, Ying, Nan Zhao, and Hongxi Yin. “Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach.“ IEEE Transactions on Vehicular Technology 67.1 (2017): 44-55.

[88] Berg, Ivan A., et al. “Machine learning in smart home control systems-Algorithms and new opportunities.“ AIP Conference Proceedings. Vol. 1906. No. 1. AIP Publishing LLC, 2017.

[89] Uddin, Md Zia, and Mi Ryang Kim. “A deep learning-based gait posture recognition from depth information for smart home applications.“ Advances in Computer Science and Ubiquitous Computing. Springer, Singapore, 2016. 407-413.

[90] Liang, Tiankai, et al. “An unsupervised user behavior prediction algorithm based on machine learning and neural network for smart home.“ IEEE Access 6 (2018): 49237-49247.

[91] Salhi, Lamine, et al. “Early detection system for gas leakage and fire in smart home using machine learning.“ 2019 IEEE International Conference on Consumer Electronics (ICCE). IEEE, 2019.

[92] Jiang, Rong, et al. “Energy-theft detection issues for advanced metering infrastructure in smart grid.“ Tsinghua Science and Technology 19.2 (2014): 105-120.

[93] Liu, Yang, Yuchen Zhou, and Shiyan Hu. “Combating coordinated pricing cyberattack and energy theft in smart home cyber-physical systems.“ IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 37.3 (2017): 573-586.

[94] Li, Weixian, et al. “A novel smart energy theft system (SETS) for IoT-based smart home.“ IEEE Internet of Things Journal 6.3 (2019): 5531-5539.