Heterogeneous Wireless Device Management in Edge Computing Systems for IoT Services

1st Mafeni Vitumbiko
School of Electronic Engineering
Seoul, Korea
vitumafeni@dcn.ss.ac.kr

2nd Younghan Kim
School of Electronic Engineering
Seoul, Korea
younghak@ssu.ac.kr

***Abstract*—In a heterogeneous Internet of Things (IoT) setup, it is impractical and requires human intervention to deploy workloads for every device after enrolment, especially considering the diverse range of wireless technologies involved. Furthermore, it is extremely complex to design and develop a large-scale system that can accommodate and integrate heterogeneous wireless technologies to achieve interoperability between the devices. As the number of devices increases, it is necessary to dynamically adjust workloads to accommodate the growing demand and maintain quality of service. In this paper we employ a state-of-art solution called Project-Flotta, which is utilized for managing edge devices and edge workloads. However, Project-Flotta does not provide a way of automating and managing heterogeneous wireless technologies for an IoT service. Therefore, we propose a system based on Project-Flotta leveraging its kubernetes custom resources by extending its functionality to adapt diverse wireless technologies.**

***Index Terms*—kubernetes, Project-Flotta, horizontal pod auto scaling, automation, edge compuning, heterogenous, wireless sensor**

I. Introduction

IoT covers diverse wireless protocols and devices, but integrating them in mixed setups is challenging [1], [2]. Handling various devices is complex [3], and cloud-centric approaches introduce latency [4]. At the same time, edge computing, dynamic resource management is crucial [5], [6]. Addressing IoT device heterogeneity, self- configuration is key for QoS in time-sensitive applications [7], [8]. Kubernetes is popular in cloud systems for its dynamic resource allocation. Kubernetes excels in dynamic resource allocation [9], [10]. Kubernetes' Horizontal Pod Autoscaler (HPA) improves utilization [11], [12]. Project-Flotta leverages Kubernetes [13], yet lacks support for diverse wireless devices, needs manual intervention, lacks load balancing, and struggles with IoT wireless tech [14]. Our study enhances Flotta by proposing:

- An EdgeDeviceAutoConfig Custom Resource (CR) and Controller for automated edge device management and configuration based on predefined specifications.
- Extension of Edge Operator, API, and CR to enable provisioning diverse wireless technologies.
- Edge device-based workload auto scaler (EHPA) for dynamic workload adjustments.
  The rest of the article is organized as follows. Section II discusses related work and background, Section III describes the proposed architecture, Section IV describes preliminary

The article is structured as follows: Section II delves into related work, Section III explains the proposed architecture, Section IV covers the preliminary system implementation, and Section V concludes and outlines future work.

II. Related Work and Background

A. Related research work

In this section, we provide a brief overview of relevant literature. While few projects directly relate to our platform, they share a similar edge computing approach.

In [3], IoT and Cloud integration for addressing heterogeneity is explored. Study [4] presents a cluster-based edge computing system using Docker, Kubernetes, and Prometheus.

For edge computing challenges, [14] introduces a KubeEdge-based node autoscaler. Additionally, [15] proposes a traffic-aware autoscaler for IoT in edge computing. All those have major contributions in Kubernetes.

Further contributions [16]–[19] enhance Kubernetes HPA and predictive models for QoS. Notably, Akri [20] supports edge device auto configuration in Kubernetes. These studies offer insights into diverse edge computing applications and challenges.

B. Horizontal Pod AutoScaler

Horizontal scaling involves deploying more Pods in response to increased load. This differs from vertical scaling where more resources like memory or CPU are allocated to existing Pods. If load reduces and the Pod count is above the set minimum, the HorizontalPodAutoscaler shrinks the workload resource (like Deployment or StatefulSet). When current metric equals desired metric, application pod count remains constant. Pseudo code for HPA can be seen in Algorithm 1

III. System Architecture Description

In this section, we introduce the main conceptual parts of our proposal and select an open source project, Project-Flotta, as the framework software. Figure 1 shows the proposed Architecture which is based on Flotta. At the edge, the Device Plugin component incorporates drivers for diverse wireless IoT technologies, enabling support for a wide range

Algorithm 1 Auto-scaling Algorithm [14]

```
Require: Input:
1:  $pods$: list of application pods in cluster.
2:  $curPods$: current number of application pods.
3:  $dPods$: desired number of application pods.
4:  $curMetVal$: current metric value.
5:  $dMetVal$: desired metric value.
6:  $HPA\_Sync\_Period$: HPA sync period.
7:  while true do
8:      $curPods$ = getCurPods()
9:      $curMetVal$ = getCurMetricValue(app)
10:     $dMetVal$ = getDesiredMetricValue(app)
11:     $ratio = \frac{curMetVal}{dMetVal}$
12:     $dPods = \lceil ratio \times curPods \rceil$
13:     if $dPods \neq curPods$ then
14:         setDesiredPods(app, dPods)
15:     end if
16:     time.sleep($HPA\_Sync\_Period$)
17: end while
```

of heterogeneous IoT devices. This allows sensor nodes adhering to specific protocols to be discovered and connected. In heterogeneous setups, devices use various communication technologies, data formats, and proprietary protocols [8]. Each Device Plugin converts data formats on a specific wireless communication model or protocol from the stream of protocols into a universal format understood by the Device Watcher. The

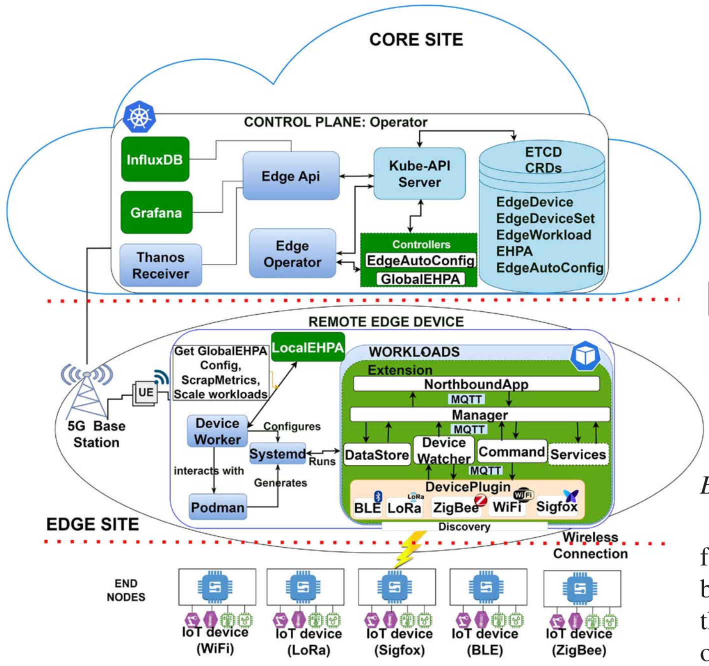


Fig. 1. Proposed Architecture

command component controls wireless devices, while services include user apps. Edge devices are managed centrally by the Cloud, deploying workloads and monitoring statuses.

For Flotta enhancement, two controllers are introduced: GlobalEHPA retrieves HPA configs and real-time statuses, while LocalEHPA manages device scaling and balancing. EdgeAutoConfig aligns end node devices.

At the device level, LocalEHPA handles scaling and balancing as device numbers rise.

IV. Preliminary Implementation

A. Testing Setup

We validate our architecture with a practical proof of concept, integrating open-source tech into a unified system. Managed by the cloud center, three Flotta edge devices cover separate locations. Our simulation (Figure 2) mirrors different sites using VLANs and a 5G UE for connectivity.

Our assumption use case is in agriculture where LoRa serves large-scale farming, Wi-Fi for greenhouses. The edge device acts as a LoRa gateway via the LoRa plugin, handling data from end nodes and actuation commands.

Upstream and downstream communication is done through Flotta API then Flotta operator, Grafana, InfluxDB, and kubectl. MQTT transports wireless plugin data, accessible through for sensor data and interactions on the edge.

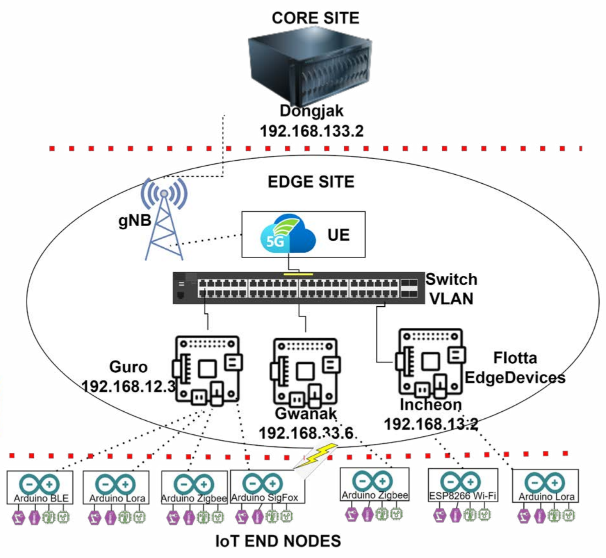
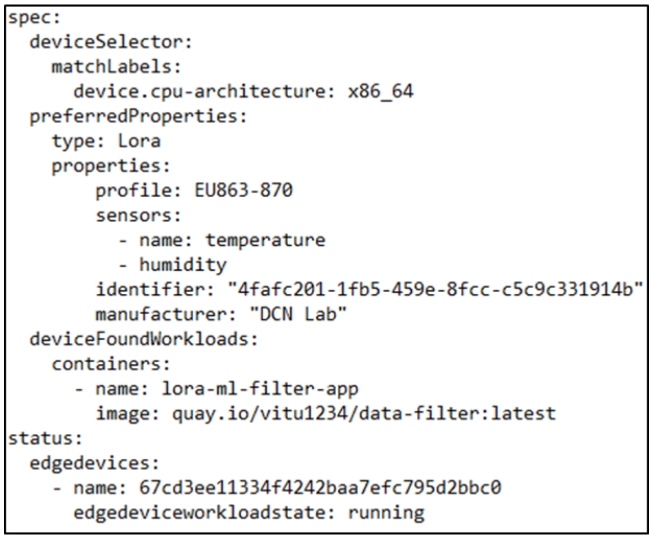


Fig. 2. Setup Topology

B. Preliminary Results

As the research uses Project-Flotta which focuses on small footprint devices starting from a tiny Raspberry Pi, we started by assessing resource usage of the introduced components on the raspberry Pi. After that we gradually increase the number of IoT devices connected to the edge devices through wireless technologies (in this case, LoRaWAN and Wi-Fi). Finally, we test memory usage of the edge devices and the cloud nodes on the APIs as traffic increases.

On the other hand, to test automated workload deployment, first we start by creating the EdgeAutoConfig CR shown in figure 3 and enroll edge devices on Flotta. If the edge device properties being registered matches the specifications in the preferredProperties part, the provided workload is then deployed to the selected edge device. In the example CR, if any IoT end nodes are connected to the edge device using Lora communication, with any of the given properties, the edge workloads image is deployed for consumption and the connected device information is added to the Status part of the CR.



```yaml
spec:
  deviceSelector:
    matchLabels:
      device.cpu-architecture: x86_64
  preferredProperties:
    type: Lora
    properties:
        profile: EU863-870
        sensors:
          - name: temperature
          - humidity
        identifier: "4fafc201-1fb5-459e-8fcc-c5c9c331914b"
        manufacturer: "DCN Lab"
  deviceFoundWorkloads:
    containers:
      - name: lora-ml-filter-app
        image: quay.io/vitu1234/data-filter:latest
status:
  edgedevices:
    - name: 67cd3ee11334f4242baa7efc795d2bbc0
      edgedeviceworkloadstate: running
```

Fig. 3. EdgeAutoConfig CR

V. CONCLUSION AND FUTURE WORKS

Our paper tackles challenges in managing wireless IoT networks, particularly in heterogeneous setups, with a focus on resource management for quality of service. Using the open-source Project-Flotta platform, we pinpoint limitations: lack of load balancing, edge HPA, and manual workload setup.

To address these, we propose enhancements. Firstly, we extend Flotta with EdgeDeviceAutoConfig for automated edge workload deployment. Secondly, we introduce an independent edge workload manager (EHPA) for dynamic adjustments.

We present initial implementation and testing, planning more tests for remaining wireless technologies in the future. This work contributes to efficient IoT network management.

ACKNOWLEDGMENT

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-01015, Development of Candidate Element Technology for Intelligent 6G Mobile Core Network)

REFERENCES

[1] A. Cilfone, L. Davoli and G. Ferrari, ”Virtualizing LoRaWAN Nodes: a CoAP-based Approach,” 2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Rome, Italy, 2019, pp. 1-6, doi: 10.1109/ISAECT47714.2019.9069691

[2] V. Safronov, J. Brazauskas, M. Danish, R. Verma, I. Lewis, and R. Mortier, “Do We Want the New Old Internet? Towards Seamless and Protocol-Independent IoT Application Interoperability,” in Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks, 2021, pp. 185–191. doi: 10.1145/3484266.3487374

[3] R. Mafamane, M. Ouadou, A. T. J. Hassani and K. Minaoui, ”Study of the heterogeneity problem in the Internet of Things and Cloud Computing integration,” 2020 10th International Symposium on Signal, Image, Video and Communications (ISIVC), Saint-Etienne, France, 2021, pp. 1-6, doi: 10.1109/ISIVC49222.2021.9487539.

[4] Y.-W. Chan, H. Fathoni, H. -Y. Yen and C. -T. Yang, ”Implementation of a Cluster-Based Heterogeneous Edge Computing System for Resource Monitoring and Performance Evaluation,” in IEEE Access, vol. 10, pp. 38458-38471, 2022, doi: 10.1109/ACCESS.2022.3166154.

[5] S. Taherizadeh and V. Stankovski, “Auto-Scaling Applications in Edge Computing: Taxonomy and Challenges,” in Proceedings of the International Conference on Big Data and Internet of Thing, 2017, pp. 158–163. doi: 10.1145/3175684.3175709.

[6] T. P. da Silva, A. F. R. Neto, T. V. Batista, F. A. S. Lopes, F. C. Delicato and P. F. Pires, ”Horizontal Auto-Scaling in Edge Computing Environment using Online Machine Learning,” 2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), AB, Canada, 2021, pp. 161-168

[7] Sharma, Y., Bhamare, D., Sastry, N.R., Javadi, B., & Buyya, R. (2022). SLA Management in Intent-Driven Service Management Systems: A Taxonomy and Future Directions. ArXiv, abs/2208.01218.

[8] M. Aboubakar, M. Kellil, and P. Roux, “A review of IoT network management: Current status and perspectives,” Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 7, pp. 4163–4176, 2022, doi: https://doi.org/10.1016/j.jksuci.2021.03.006.

[9] L. Toka, G. Dobreff, B. Fodor and B. Sonkoly, ”Machine Learning-Based Scaling Management for Kubernetes Edge Clusters,” in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 958-972, March 2021, doi: 10.1109/TNSM.2021.3052837.

[10] Internet Research Task Force (IRTF), Intent Classification,October 2022: https://www.rfc-editor.org/rfc/rfc9316

[11] L. Lovén et al., ”Scaling up an Edge Server Deployment,” 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Austin, TX, USA, 2020, pp. 1-7, doi: 10.1109/PerComWorkshops48775.2020.9156204

[12] K. Casey, “Kubernetes Autoscaling, explained,” The Enterprisers Project, https://enterprisersproject.com/article/2021/3/kubernetes-autoscaling-explanation (accessed May 26, 2023).

[13] “What is Flotta,” Project Flotta, https://project-flotta.io/documentation/v0_2_0/intro/overview.html (accessed May 26, 2023).

[14] L. H. Phuc, M. Kundroo, D. -H. Park, S. Kim and T. Kim, ”Node-Based Horizontal Pod Autoscaler in KubeEdge-Based Edge Computing Infrastructure,” in IEEE Access, vol. 10, pp. 134417-134426, 2022, doi: 10.1109/ACCESS.2022.3232131

[15] L. H. Phuc, L. -A. Phan and T. Kim, ”Traffic-Aware Horizontal Pod Autoscaler in Kubernetes-Based Edge Computing Infrastructure,” in IEEE Access, vol. 10, pp. 18966-18977, 2022

[16] A. A. Pramesti and A. I. Kistijantoro, ”Autoscaling Based on Response Time Prediction for Microservice Application in Kubernetes,” 2022 9th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), Tokoname, Japan, 2022, pp. 1-6

[17] L. M. Ruíz, P. P. Pueyo, J. Mateo-Fornés, J. V. Mayoral and F. S. Tehàs, ”Autoscaling Pods on an On-Premise Kubernetes Infrastructure QoS-Aware,” in IEEE Access, vol. 10, pp. 33083-33094, 2022

[18] Z. Zhou et al., AHPA: Adaptive Horizontal Pod Autoscaling Systems on Alibaba Cloud Container Service for Kubernetes. 2023.

[19] S.-H. Kim and T. Kim, “Local Scheduling in KubeEdge-Based Edge Computing Environment,” Sensors, vol. 23, no. 3, 2023, doi: 10.3390/s23031522.

[20] T.-T. Nguyen, Y.-J. Yeom, T. Kim, D.-H. Park, and S. Kim, “Horizontal Pod Autoscaling in Kubernetes for Elastic Container Orchestration,” Sensors, vol. 20, no. 16, 2020, doi: 10.3390/s20164621.

[21] “Akri Official Documentetion,” Akri, https://docs.akri.sh/ (accessed May 26, 2023).