Flotta - Edge Example Application with sensing the Internet

1. Project Overview

Flotta Operator is a Kubernetes operator used to manage the workloads of the IoT edge devices via Kubernetes API.

Flotta workloads running on low-resource devices like RaspberryPI, Nvidia Jetson Boards, etc. These devices can be located anywhere, under so many circumstances which introduces many problems that affect the performance of the network.

The current situation is there is no example project for using Flotta which is making a barrier to adopting the Flotta Project.

The project goals are twofold:

1. **Edge Example Application**
    - Implementation of workload application that would run on a bunch of Flotta devices collecting video streams/pictures from webcam and/or getting other data from sensors.
    - Implementation of WebApp that runs in the Flotta cluster that presents the results of aggregated data from devices in a nice UI to help users get some insights from data.
2. **Sense the Internet**

Investigating the Internet network topology and measuring the performance of the network using devices managed by Flotta.

This will be done by performing large-scale network topology discovery efficiently (both at the IP level and at the router level), sensing the quality of the network using active probing tools like traceroute, ping ..etc, and presenting this data in clear graphs with filters on the WebApp that running in the cluster to help project users making better decisions.

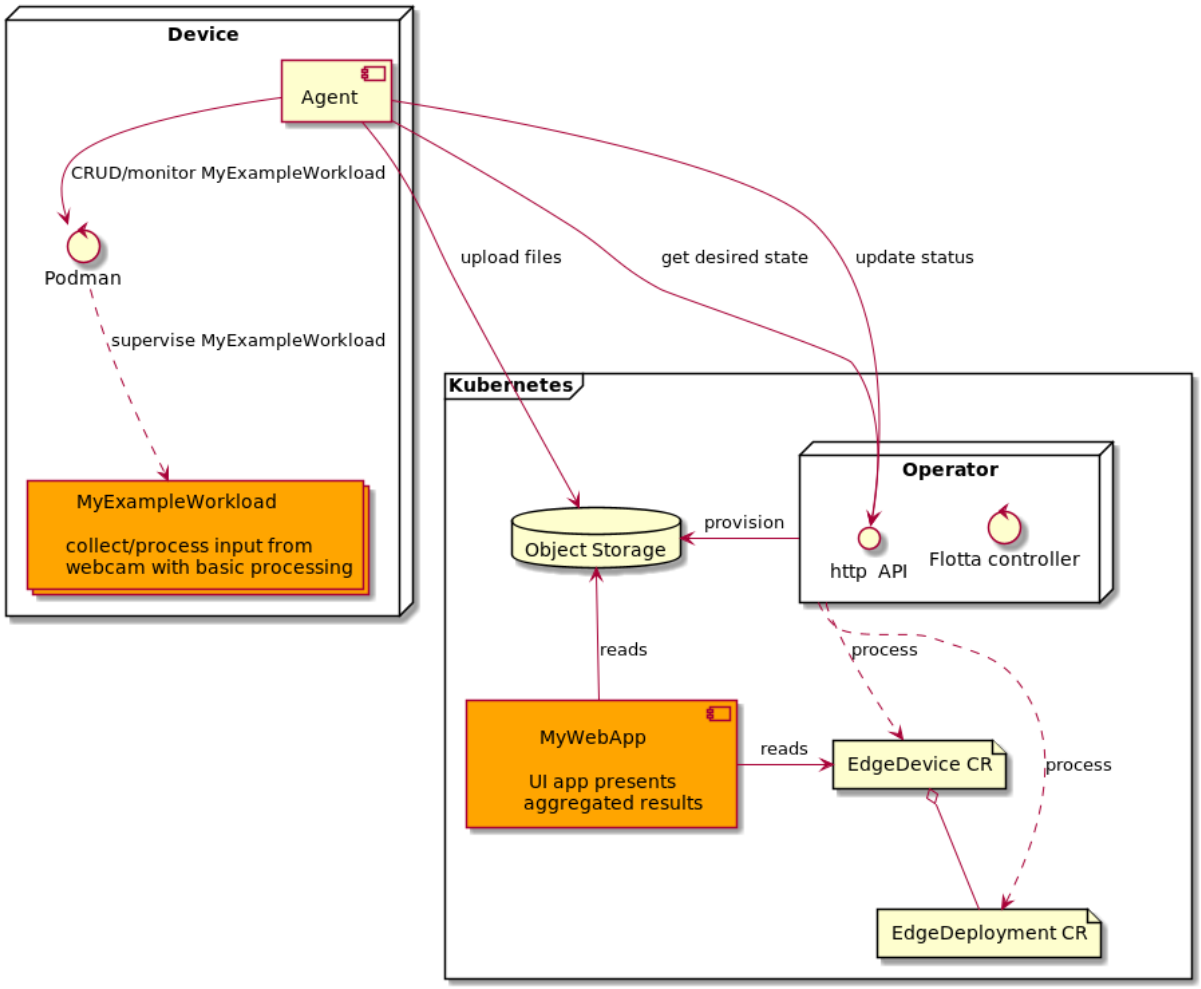
2. Deliverables

1. Implement the workload app that is running on Flotta devices, with capabilities of
    a. Reading & collecting from webcams and sensors.
    b. Performing a large-scale network topology discovery.
    c. Sharing collected data to the Flotta cluster.
2. Implement a Web App in the cluster, with capabilities of
    a. Presenting aggregated data from sensors and webcams in a nice format.
    b. Presenting devices networks topology in graphs.
    c. Providing filters and Frontend functionalities to help users read collected data.
3. Write Documentation for the Edge Example App.

3. Implementation Plan

General picture for the project

This is the General picture and design for an Edge Device running with the Flotta Operator.



Orange boxes are the project implementation.

We have 2 main components:

1. Workload App - Device Side

Workload app is a containerized app to collect inputs from sensors and/or webcams and from the network topology discovery tools.

This app is responsible for collecting and storing the data in its volume and sending it to the cluster.

2. Web App - Cluster Side

A Web app is responsible for reading the data from the bucket in the cluster and presenting it on the Frontend with topology graphs and filters for the data.

Implementation Details

Workload App

I am going to implement the workload app in the **Go programming language** which has robust support for building applications in the Kubernetes and Web community.

For the network topology discovery, I am going to use one of the well-known tools like iffinder, kapar, or other equal tools after discussing it with the team.

Also, I am going to use **Docker** as a containerization tool.

Lastly, I am going to write some **unit tests** with **Ginkgo** and **Gomega** that combination of them make a very good **BDD** (Behavior-Driven Development) test framework for Go.

Tasks Involved:

- Initializing a Go app and its Dockerfile.
- Collecting data from sensors and webcams and saving it until sending it to the cluster.
- Collecting network data through network tools and saving it until sending it to the cluster.
- Sending data to the cluster over the agent.
- Writing unit tests with Ginkgo and Gomega.
- Writing Documentations (e.g Readme files).
- Writing a blog post about the app details

Web App

I am going to split this application into a **Frontend** app and a **Backend** app to reduce the dependability between the two of them.

For the Backend, I am going to use **Go** as most of the apps in Flotta are written in.

This app is going to read the data from the cluster bucket.
This app will have **unit tests** also to ensure its reliability.

For the Frontend, I am going to use **NuxtJS** which is a framework on top of **VueJS** but it may not be the best choice as it has many capabilities that we don't need, so I am going to discuss with the community what may be the best choice for this app

These two projects will be containerized with **Docker** too.

Tasks Involved:

- **Backend**
  - Initializing a Go app and its Dockerfile.
  - Retrieving the files from the object storage.
  - Reading and processing files data.
    - Processing of network data
  - Creating endpoints to expose the data to the Frontend.
    - Expose sensors collected data endpoints.
    - Expose Network collected data endpoints.
  - Writing unit tests with Ginkgo and Gomega.
  - Writing Documentation (i.e Readme file).
- **Frontend**
  - Searching for a NuxtJS boilerplate to start with.
  - Consuming the backend endpoints.
  - Creating filters and Graphs for data visualization.
  - Writing Documentation (i.e Readme file).

Documentation for both Applications

I am planning to write documentation for both projects to make it easier for contributions in the future, the documentation will cover:

- Description of the projects and how they work.
- How we can run the projects to contribute.
- Updating the main project Documentation with references for the new projects.
- In general, the Documentation will be in markdown format with some UMLs if needed.

4. Milestones

Notes:

- I plan to work for about 30 hours per week.
- My college final exams are not accurately determined but they will take place in June so, maybe I am going to start a little bit late than the official start date.
- I might work on the project before the official start date (before June) but I am not counting on this in the milestones.
- I plan to extend the program period to 14 weeks instead of 12 and submit my final results from 19/09 to 23/09 (week 14 from the program start date).

Milestone	Tasks	Start Date	End Date
1. Before the official start date		**18/03/2022**	**12/06/2022**
1.1	- Understand the Flotta-operator job. - Familiarize with the code. - Familiarize with the community. - Install and run the project locally. - Solve some open issues.	18/03/2022	01/04/2022
1.2	- Read and educate myself about networking topics.	01/04/2022	09/04/2022
Off time - College Exams (Midterms)		10/04/2022	01/05/2022
1.3	- Start to build some basic functionalities.	02/05/2022	30/05/2022
Off time - College Exams (Finals)		01/06/2022	30/06/2022
2. Workload App		**03/07/2022**	**30/07/2022**
2.1	- Implement collecting data from sensors or webcams & sending it to the cluster.	03/07/2022	09/07/2022
2.2	- Implement collecting network data using the topology discovery tools.	10/07/2022	20/07/2022
2.3	- Write Unit tests for workload app. - Write a Readme file for the workload app.	21/07/2022	28/07/2022
2.4	- Write a Blog Post for the workload app.	29/07/2022	31/07/2022
First Evaluation		**01/08/2022**	**05/08/2022**

2. Workload App		**31/07/2022**	**06/08/2022**
2.5	- Implement workload Custom Resource Definition (CRD).	31/07/2022	6/08/2022
3. Web App		**07/08/2022**	**27/08/2022**
3.1	- Initialize the Backend app. - Expose sensors collected data endpoints.	7/08/2022	13/08/2022
3.2	- Implement the processing of network data - Expose network collected data endpoints. - Initialize the Frontend app.	14/08/2022	20/08/2022
3.3	- Implementing the UI for sensors collected data. - Implement the UI for network collected data with filters and graphs.	21/08/2022	01/09/2022
3.4	- Write Unit tests for the Web app. - Write Readme file for workload app.	02/09/2022	05/09/2022
3.5	- Write a Blog Post for the Web app. - Update the General Documentation app.	06/09/2022	11/09/2022
Time Buffer & enhancements		**12/09/2022**	**19/09/2022**
Second Evaluation - End of Program		**19/09/2022**	**23/09/2022**

5. Previous Open-Source Pull Requests

I'm pretty new to the open-source world but I got some professional work experience too

meshery/meshery-nsm - Open Source

- [merged] Update slack with newcomers issues.

project-flotta/flotta-operator - Open Source

- [opened issue] Error in installing the project with Go 1.18
- [merged PR] Mentioning Go version in docs

Our Education For Investments - work experience - 1,5 years

- Building backend microservices in Go.
- Creating and Deploying our Apps using Docker, k8s, AWS.

Kite Agency - work experience - 1 year

- Building Web Apps using PHP Laravel and VueJS for Frontend.
- Intern for the first 3 months and Full-time developer for the rest of the year.

6. Personal Details

Name	Ahmad
Surname	Muhammad Ateya

Contacts and Professional Profiles

Email	ahmad.m.ateya@gmail.com
GitHub	https://github.com/ahmadateya
LinkedIn	https://www.linkedin.com/in/ahmadateya