uMEC for Akraino Hackathon

November 12, 2019

Tapio Tallgren
Ferenc Székely
What is a “Smart City”?
The recently developed 5G smart pole concept in the LuxTurrim5G ecosystem is moving towards productization and practical piloting. Another goal is to create a platform utilizing a wide variety of data in a reliable and secured way and develop new digital services to meet real needs of cities.

Through a two-year and EUR 26 million funding, the group of 26 partners target the global smart city markets worth tens of billions euros in close collaboration. Business Finland provides innovation funding for the project.

The first phase of the LuxTurrim5G project, which ended in May, successfully developed the 5G smart pole concept, which integrates the 5G base station, weather and air quality sensors, video cameras, monitors electric vehicle charging unit and other active devices. The good results and the first pilots at the Nokia Campus in Espoo, Finland have attracted a lot of interest around the world and given the LuxTurrim5G ecosystem a boost for further expansion.
Open data

Open data is publicly available data that can be universally and readily accessed, used, and redistributed free of charge. It is structured for usability and computability. (Source: GovLab)

Explore open data resources

International Cooperation
Open Data
What is µMEC?
LuxTurrim5G light pole
µMEC concept

- µMEC complements the emerging 5G radio networks by enabling new applications
- µMEC is a small form factor HW+SW platform for especially the Smart City services on Ultra Far Edge
- It can use 5G, WLAN or fiber connection
- It can be installed on light poles, vehicles, etc
- The µMEC proof-of-concept is based on LuxTurrim5G and open source components
What is Multi-Access Edge Computing?
Multi-Access Edge Computing

› Standardized application development model for the Edge
› Interfaces are defined using OpenAPI that allows generating server and client stubs for tens of programming languages
› MEC-11 (Application Enablement) allows modifying traffic rules, DNS rules, and discovering new services
› Supports multiple transports, security with OAuth2.0 etc
ETSI MEC (Multi-Access Edge Computing)

### services

| GET | /services |

This method retrieves information about a list of service resources. This method is typically used in the "service availability query" procedure.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ser_instance_id</td>
<td>A mobile edge application instance may use multiple <code>ser_instance_ids</code> as an input parameter to query the availability of a list of mobile edge service instances. Either &quot;ser_instance_id&quot; or &quot;ser_name&quot; or &quot;ser_category_id&quot; or none of them shall be present.</td>
</tr>
<tr>
<td>ser_name</td>
<td>A mobile edge application instance may use multiple <code>ser_names</code> as an input parameter to query the availability of a list of mobile edge service instances. Either &quot;ser_instance_id&quot; or &quot;ser_name&quot; or &quot;ser_category_id&quot; or none of them shall be present.</td>
</tr>
<tr>
<td>ser_category_id</td>
<td>A mobile edge application instance may use <code>ser_category_id</code> as an input parameter to query the availability of a list of mobile edge service instances in a <code>serCategory</code>. Either &quot;ser_instance_id&quot; or &quot;ser_name&quot; or &quot;ser_category_id&quot; or none of them shall be present.</td>
</tr>
</tbody>
</table>

**AKRAINO Edge Stack**
What is Kubernetes?
API OVERVIEW

Welcome to the Kubernetes API. You can use the Kubernetes API to read and write Kubernetes resource objects via a Kubernetes API endpoint.

Resource Categories

This is a high-level overview of the basic types of resources provided by the Kubernetes API and their primary functions.

Workloads are objects you can use to manage and run your containers on the cluster.

Deployment objects allow you to declaratively specify an application’s configuration and behavior.

Discovery & Load resources allow you to define and use a set of Kubernetes objects together in an externally accessible, load-balanced service.

Storage resources allow you to store configuration and data for your application components.

Infrastructure resources help manage your underlying infrastructure.

Resource Objects

Resource objects typically have 3 components:

- Resource Object Data: This is metadata about the resource, such as its name, type, annotations, and labels. This contains fields that may be updated both by the end user and the system (e.g., annotations).
- ResourceSpec: This is defined by the user and describes the desired state of the object. It is this in which creating or updating an object.
- ResourceStatus: This is filled in by the server and reports the current state of the object. In most cases, users don't need to change this.

Resource Operations

Most resources provide the following operations:

Create

Create operations will create the resource in the storage backend. After a resource is create the system will apply the desired state.

Update

Updates come in 2 forms, Replace and Patch.

- Replace: Replacing a resource object will update the resource by replacing the existing one with the provided one. For read-only operations this is safe because an optimistic lock failure will occur if the resource is updated. To update the status, one must replace the specific status update operation.
- Patch: Replacing a resource object may not result immediately in changes being propagated to downstream objects. For instance replacing a ConfigMap or Secret resource will not result in all Pods seeing the changes.
API OVERVIEW

Welcome to the Kubernetes API. You can use the Kubernetes API to read and write Kubernetes resource objects via a Kubernetes API endpoint.

Resource Categories

This is a high-level overview of the basic types of resources provided by the Kubernetes API and their primary functions.

Resource Objects

Resource objects describe the behavior of other resources in the cluster. Each resource type has its own API operations.

- **Resource Object Data**: This is metadata about the resource, such as its name, type, annotations, and labels. This contains fields that are populated both by the end user and the system (e.g., annotations).
- **ResourceSpec**: This is defined by the user and describes the desired state of the resource. It is this in which creating or updating an object.
- **ResourceStatus**: This is filled in by the server and reports the current state of the system. In most cases, users don’t need to change this.

Resource Operations

Most resources provide the following operations:

- **Create**: Create operations will create the resource in the storage backend. After a resource is create the system will apply the desired state.

- **Update**: Updates come in 2 forms: **Replace** and **Patch**.
  - **Replace**: Replacing a resource object will update the resource by replacing the existing spec with the provided one. For read-only write operations this is safe because an optimistic lock failure will occur if the resources is updated. To update the status, one must utilize the specific status update operation.
  - **Patch**: Replacing a resource object may not result immediately in changes being propagated to downstream objects. For instance replacing a **ConfigMap** or **Secret** resource will not result in all Pods seeing the changes.
Putting it all together: your task is to create a **Smart City** service that uses a **µMEC** that is deployed in the city.

For simplicity, we have a ready-made application as a starting point.

Your solution can include elements outside this app.
Your city...
1. End user will use a browser to access the service

2. Web server will return a JavaScript page

3. JavaScript will use the camera to take a picture

4. The web server will use MEC-11 to find a Tensorflow service to analyze the image

5. The web server will use the results of the analysis and do something
How it works

› You need to bring in your own laptop and smart phone
› You will be able to download the sample code from Akraiino gerrit and see how it runs on micromec.org/hack/selfie
› To modify and run the modified code,
  › clone it to your own Github repo
  › enable webhooks on the repo
› Your app will be served on a web server with a public URL
› You can also modify the Tensorflow model that the sample app uses. Please talk to us, if you want to do that.
Development workflow

Developer

Github repo (with webhooks)

Micromec.org
- Program 1
- Program 2
- Program 3

µMEC cluster
- MEC-11
- Sensors
- Camera
- Lights
Create a new github repo

Click: [https://github.com/new](https://github.com/new)

Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner
Repository name *

Great repository names are short and memorable. Need inspiration? How about **curly-octo-robot**?

Description (optional)

Our team’s solution for the challenge.

Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Skip this step if you’re importing an existing repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None  Add a license: None

Create repository
Add a webhook to the repository

Payload URL: https://micromec.org/hack

Content type: application/json

Secret: leave it blank

SSL verification: Enable

Events: just the push event
Webhook is now active

Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we’ll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

- https://micromec.org/haack (push)
Back to webhook

Check **Recent Deliveries** on the webhook’s settings page.

The delivery from the webhook is marked with a **green** tick mark.

If the mark is **red** then please **contact** the hackathon **organizers** online.
Recent deliveries

Response should be HTTP 200.

Click Redeliver to test the hook.

Recent Deliveries

Access-Control-Allow-Origin: *
Content-Length: 13
Content-Type: text/html; charset=utf-8
Date: Tue, 12 Nov 2019 09:52:02 GMT
X-Powered-By: Express

{"msg":"ack"}
Local setup

Follow instructions from Github.

“Business as usual.”

Initiate your local copy of the repository and make the first commit and push.
Check the result

We pull and deploy your code to micromec.org.

See the list of projects: 
https://micromec.org/hack

Our server will host standard HTML, JS and CSS. We also allow small images (max. 1M).

Please do not push inappropriate content!
Develop your app

“Commit early, commit often!”

Work on your project and push changes from the command line (for instance).

Github will trigger a new deployment on micromec.org.
Check results again

Your new code is deployed and available at micromec.org.

Have a lot of fun...

Happy hacking!