Orchestrating a brighter world

NEC

KMD STEAM Aalborg 2019

Building apps with Smart City Technology: FIWARE

Edge Computing in FogFlow

17.09.2019, Jonathan Fürst

NEC Laboratories Europe

IoT: Instrumenting the Physical World

→ Goal: Use sensor data to understand (model) and influence (actuate) physical systems.

Let's just use the cloud for that!

Cloud Computing

What might be the problem with that?

Cloud Computing

Good:

- Centralized (easy and efficient to manage)
- Elastic and powerful resources
- Cheap (pay what you use)

Bad:

- Latency
- Reliability
- Bandwidth Limitations

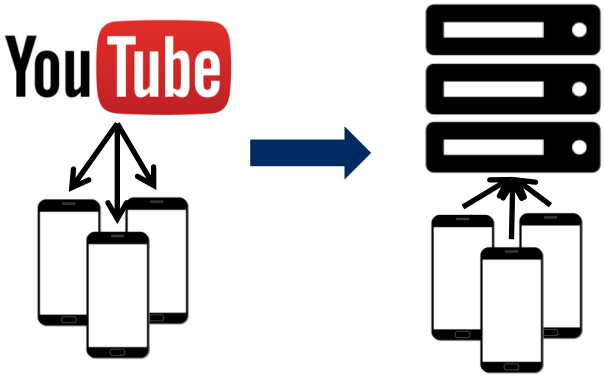
Ugly:

- Data privacy
- Security

Paradigm Shift in IoT Applications

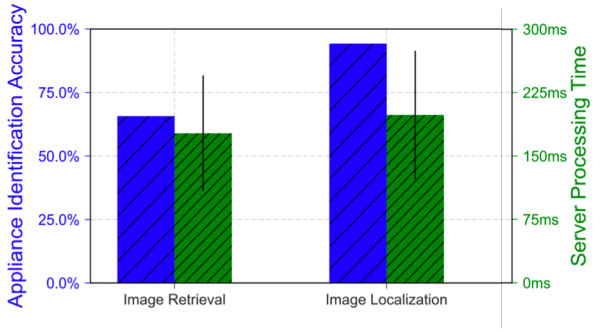
Data Flow is being reversed

- Traditional: Content distribution from core to edge.
- IoT: Generate (huge amounts of) data at the network edge and move towards core to process in cloud.



Diverse and **interdependent QoS requirements** of IoT application

- Complex tradeoffs between **responsiveness**, **accuracy**, **power** consumption, and **cost**.



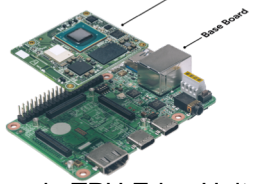
Rich clients with (some) processing power, that make decision based on local data.

- E.g., drone, autonomous vehicle, smartphone...


ARM Cortex-A53

x86–64 Intel i7


NVIDIA Jetson TX2


Google TPU Edge Unit

Example: Camera Input

Camera Input

- Huge amounts of data (would congest network links).
- High processing power needed for CV algorithms (cannot just do all on device).
- Requires low latency for many applications (augmented reality, surveillance, traffic…).

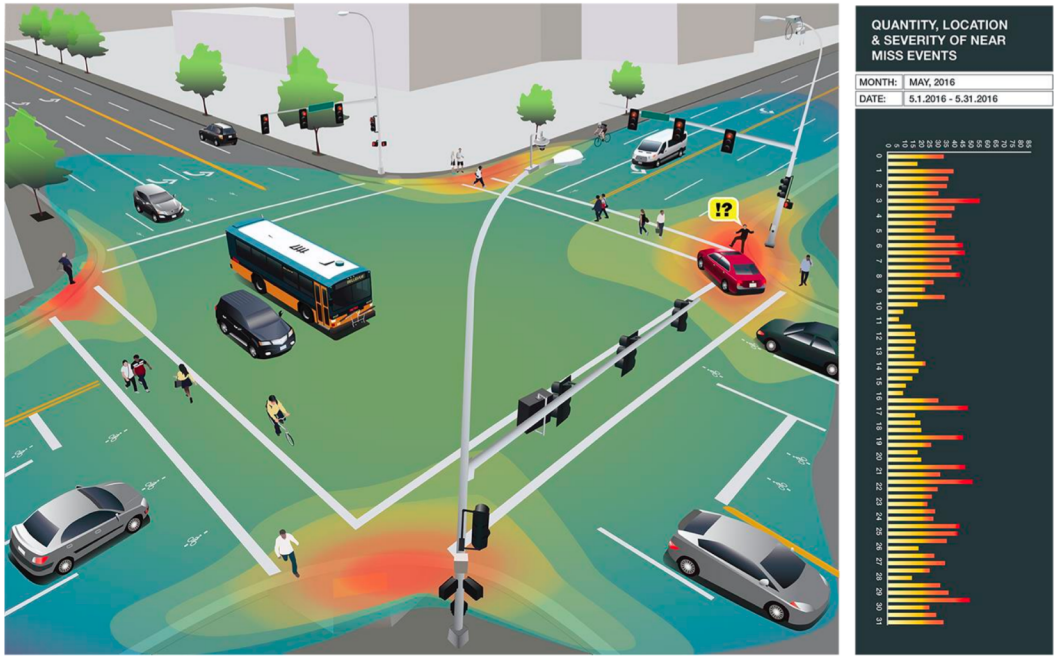
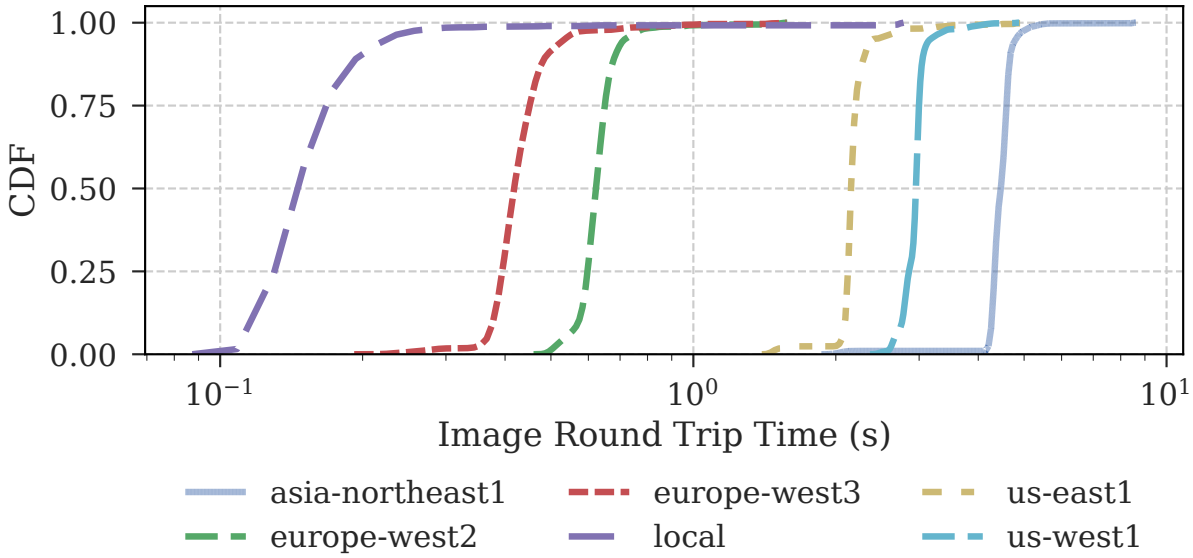


Image source: Vision Zero, Franz Loewenherz, City of Bellevue

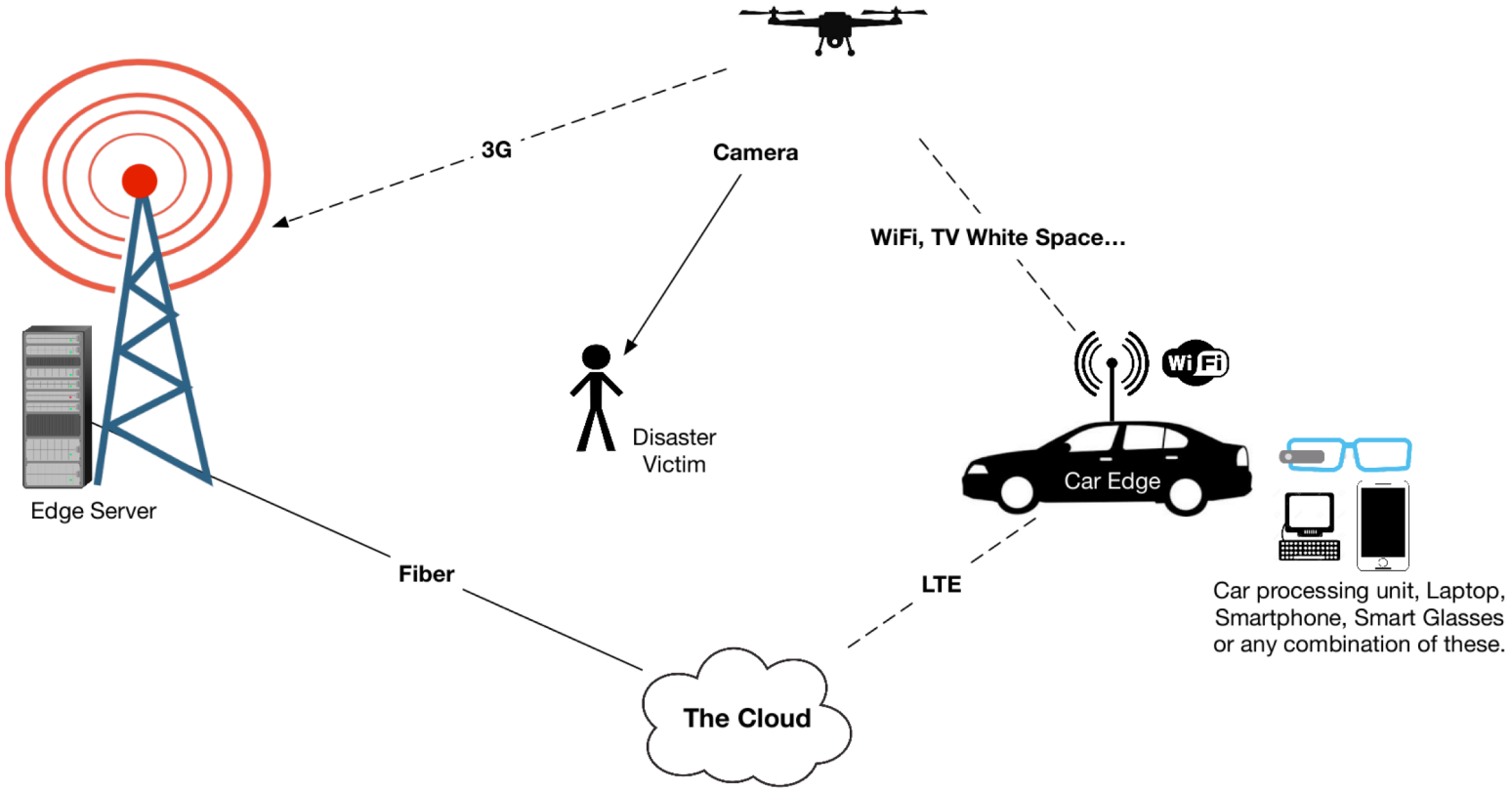
Edge Computing: Goals

- Smaller Latency → Enable responsive sensitive applications
- Improve Reliability → Don't depend on cloud response
- Avoid Bandwidth Limitations → Just single hop to edge device
- Improve Data Privacy → Data processing in own network

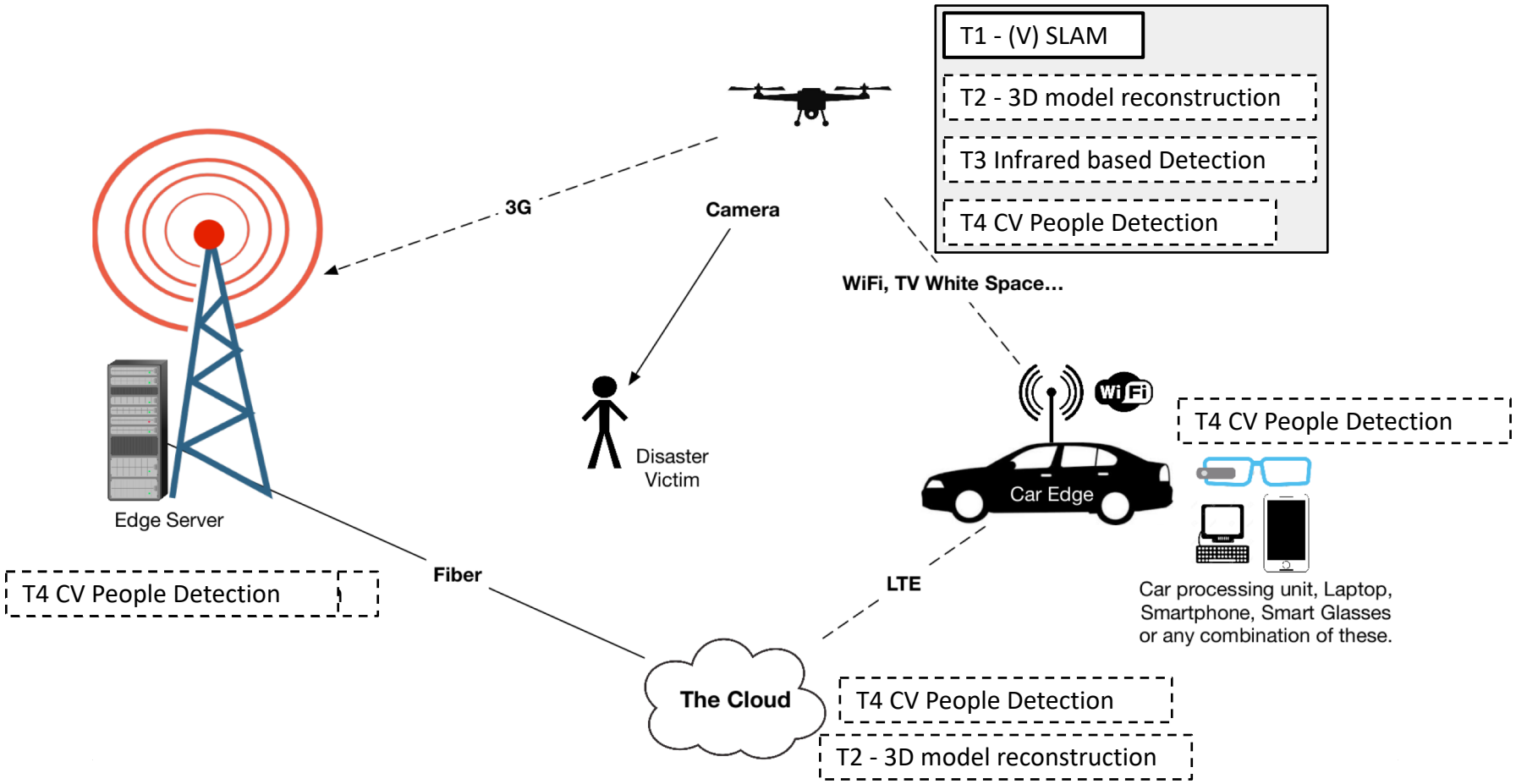


→ Bring computation closer to data producers and consumers (e.g., IoT devices).

Future Application Scenario



Future Application Scenario



Edge Computing: Challenges

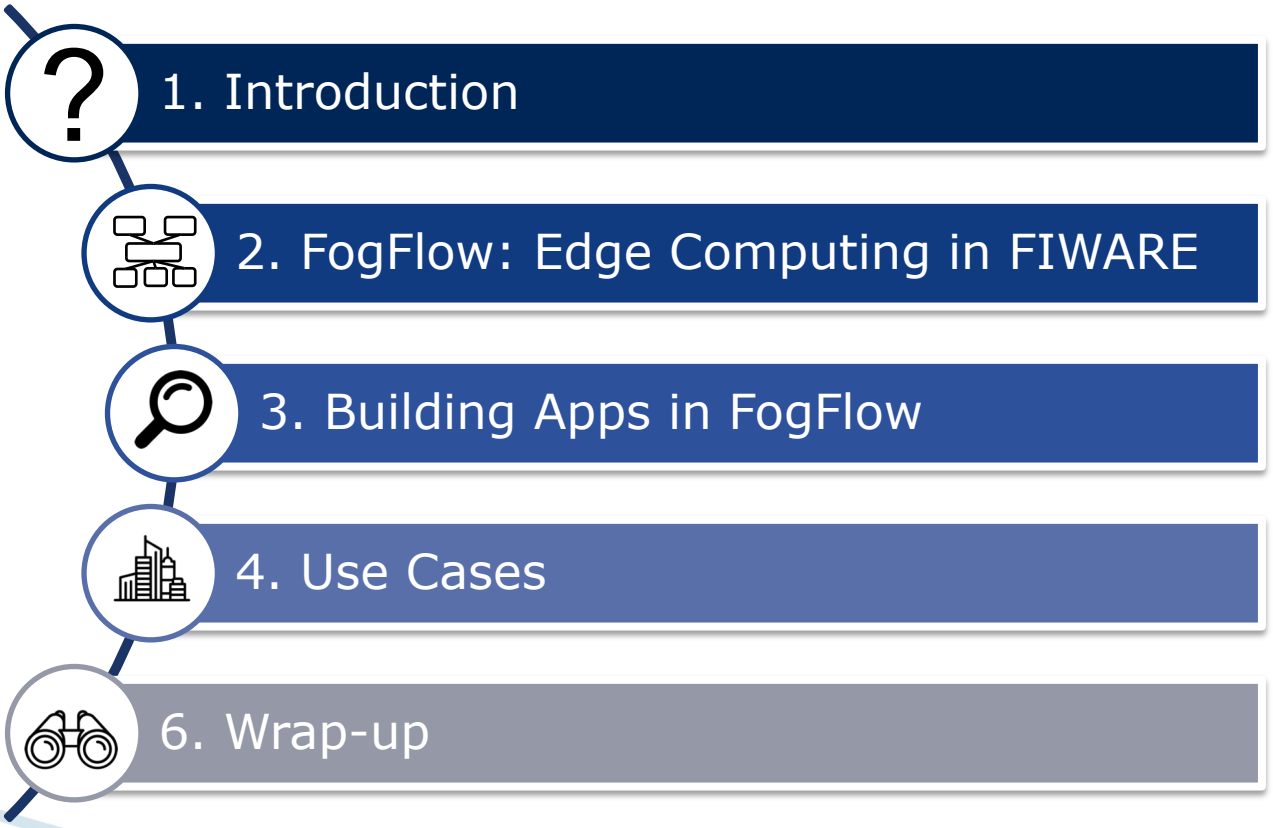
1. How to distribute computational tasks across networked devices efficiently? Bonus: Run them reliable.

} Distributed Execution Framework

2. How to provide easy interface to the developers?

} Programming Model

Overview


1. Introduction
2. FogFlow: Edge Computing in FIWARE
3. Building Apps in FogFlow
4. Use Cases
6. Wrap-up

FogFlow: Edge Computing in FIWARE

FIWARE: Open Source Ecosystem for Smart Solutions

Context describes what happens, who, where, when, why, and how

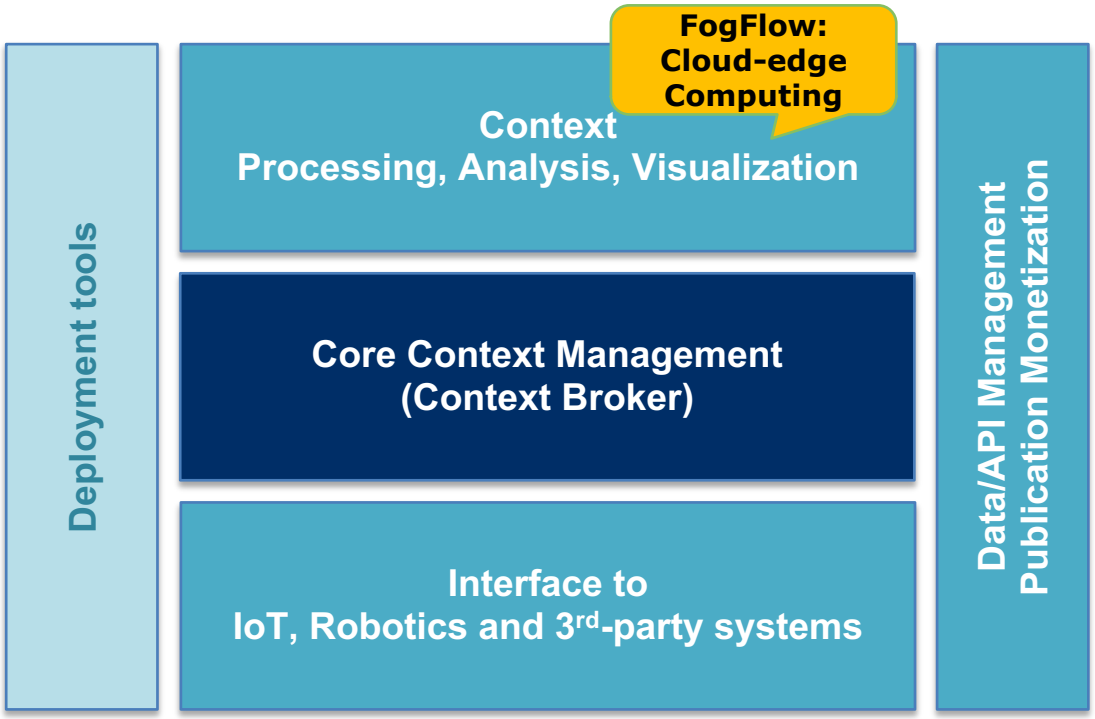
- FIWARE is an open source ecosystem/platform to provide smart solutions for many domains: smart city, smart industry, …
- FIWARE is driving the de-facto standard (**NGSI-LD**, next generation service interface for linked data) for information sharing and exchange cross domains
- **FogFlow** framework is an open-sourced FIWARE key enabler

FIWARE NGSI LD FogFlow

https://www.fiware.org

FIWARE: Open Source Ecosystem for Smart Solutions

- Providing more than 20 open source generic enablers for building up out-of-shelf smart solutions for smart cities, smart industry, …
- Driving key standards for breaking information silos
- Making IoT simpler and transforming Big Data into knowledge
- Enabling data economy and ensuring data sovereignty



Motivation

*Edge computing has great potential to reduce **bandwidth consumption, end-to-end latency** and **privacy**, but it raises much more complexity than cloud computing since the cloud-edge environment is more open, heterogeneous, and dynamic*

service/application providers

New requirements come frequently

new services

service realization during the development phase

Complicate to realize services due to lack of programming model and poor interoperability: spend months for each service

?

Programming Model

Can we program applications over cloud-edges easily, like programming them in the cloud?

resource management during the deployment phase

No approach of dealing with dynamics like device mobility, instant service usage, temporary failure: applications have to face those issues

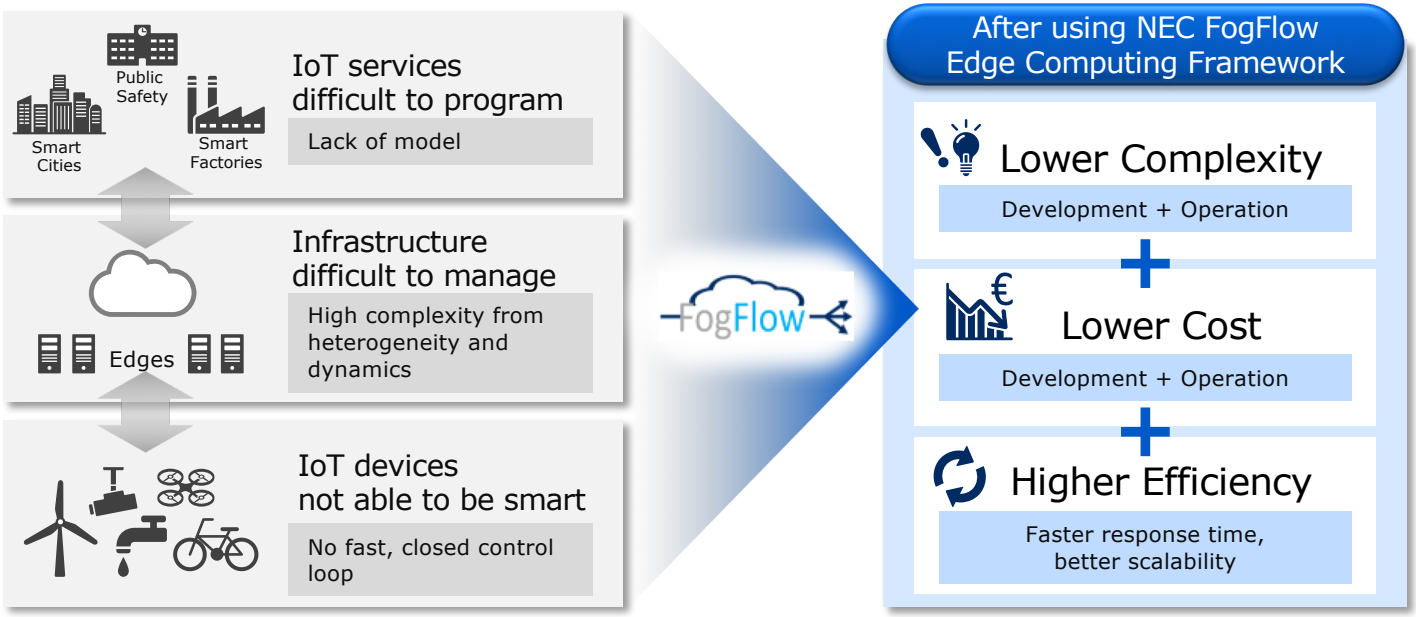
?

Execution Framework

Can we let the cloud-edge platform to automatically manage and optimize its own resources under such dynamics?

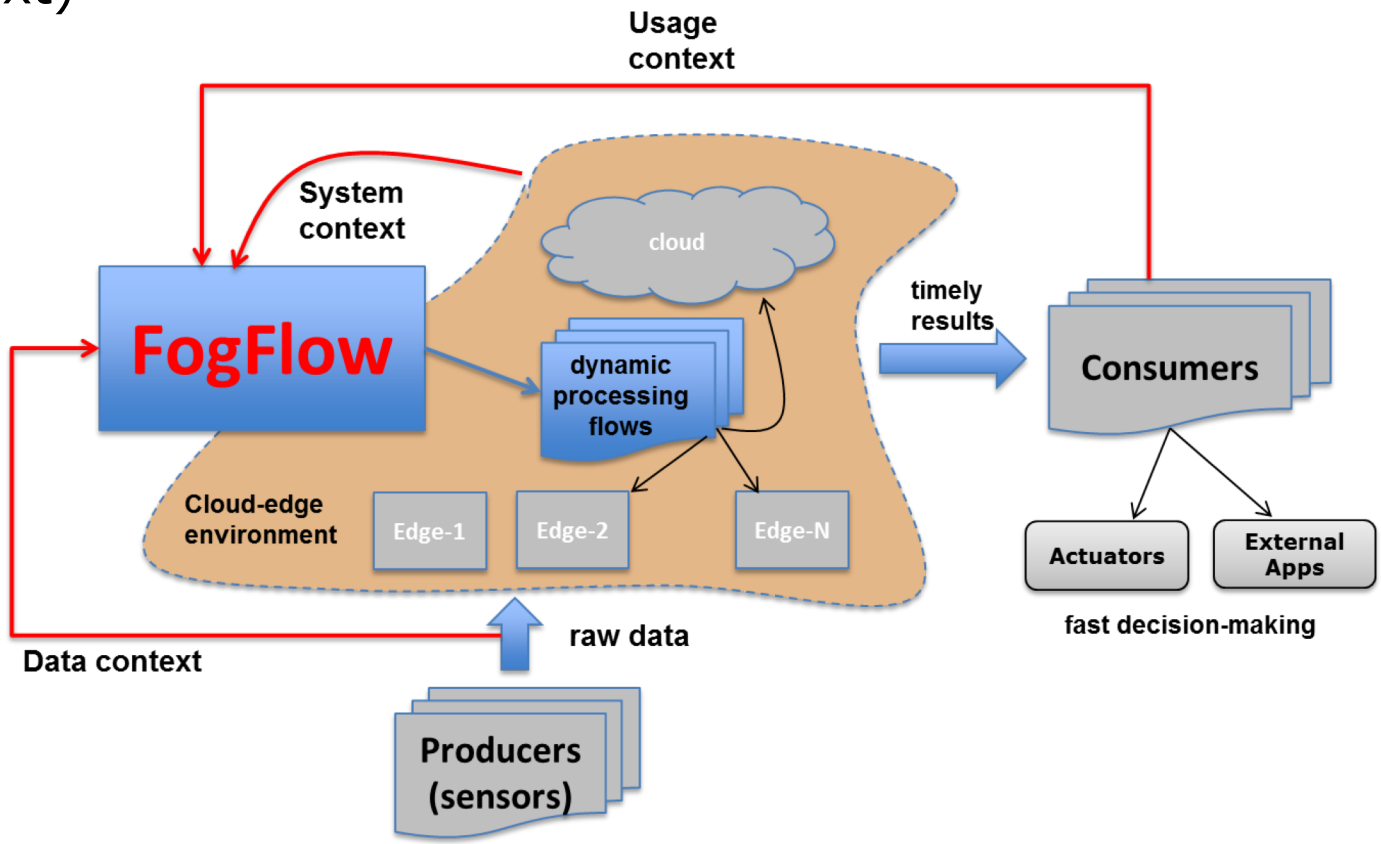
Cheng, Bin, et al. "FogFlow: Easy Programming of IoT Services Over Cloud and Edges for Smart Cities." IEEE Internet of Things Journal (2017).

Benefits

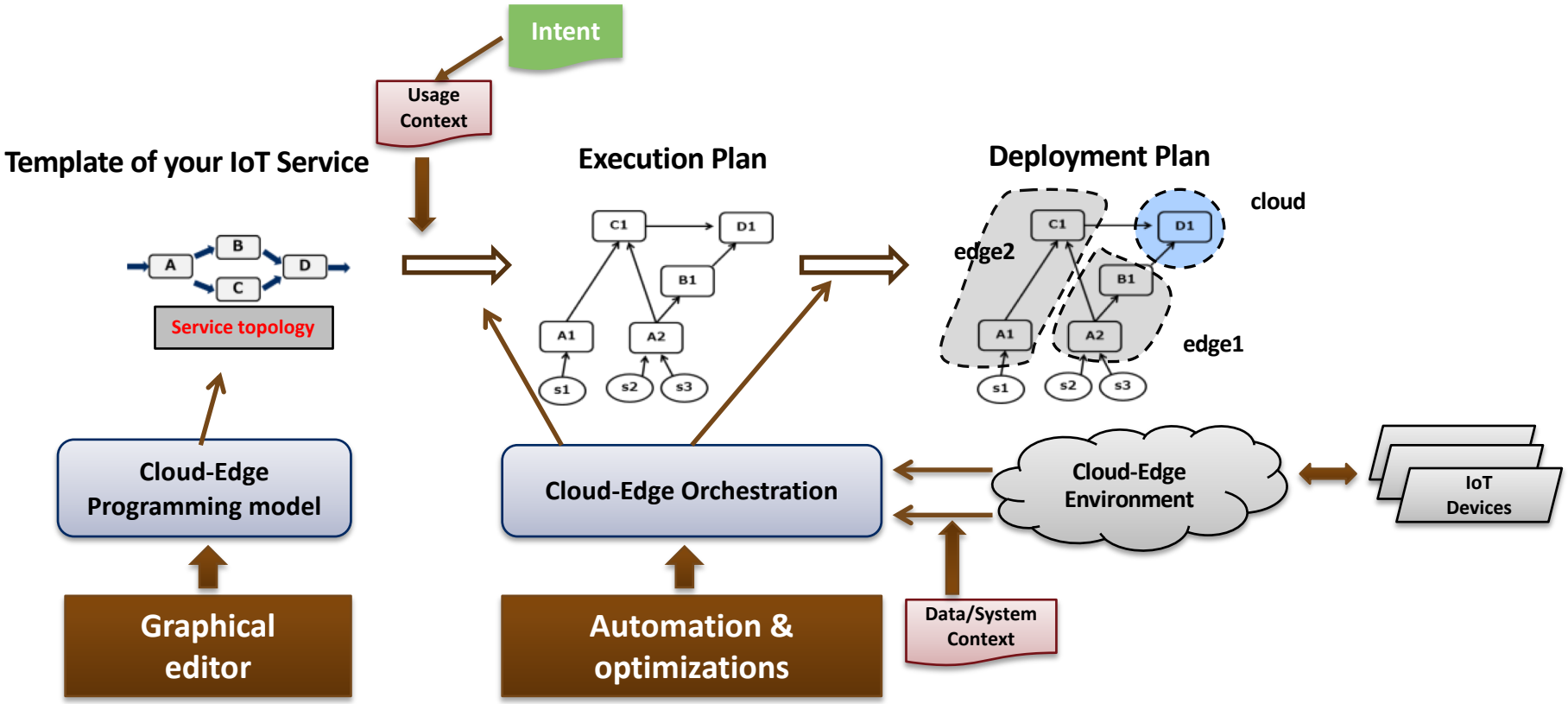


High Level View

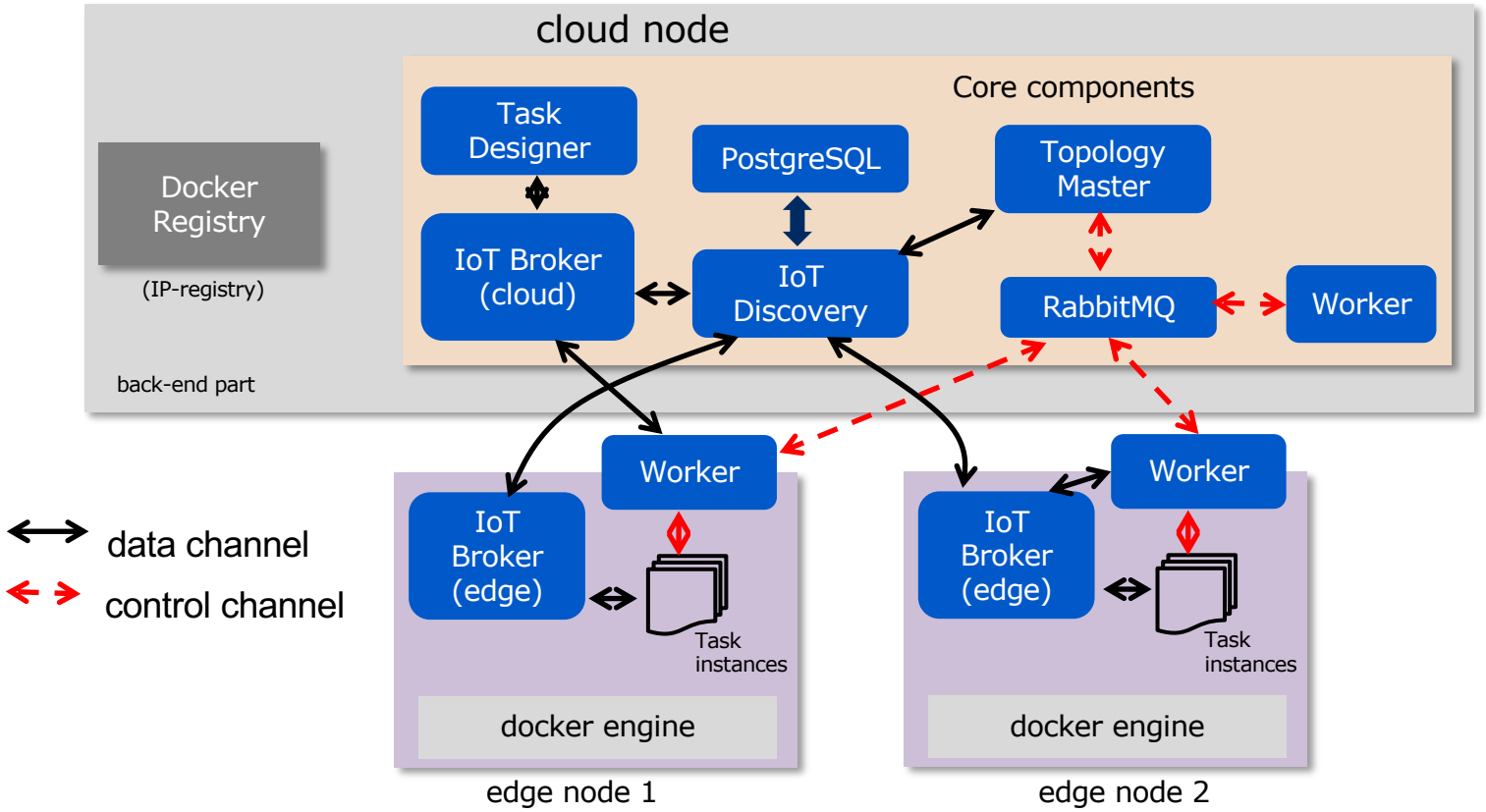
FogFlow is a ***cloud-edge orchestrator*** to orchestrate dynamic NGSI-based data processing flows on-demand between producers and consumers for providing timely results to make fast actions, based on context (system context, data context, and usage context)



How it works

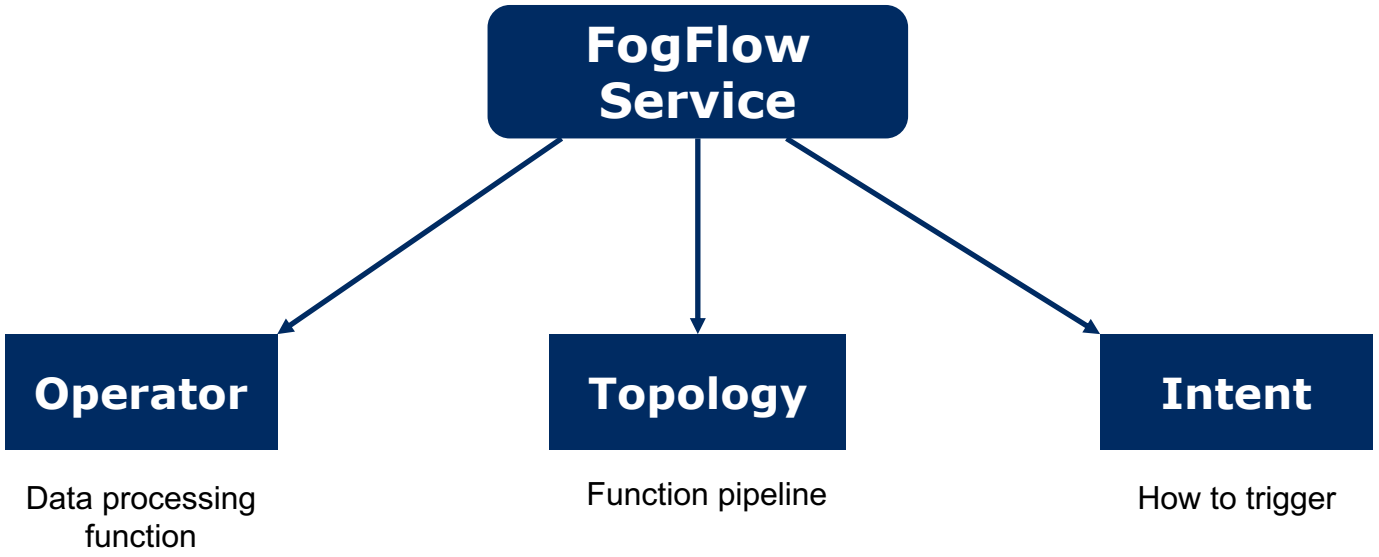


System Deployment View of FogFlow

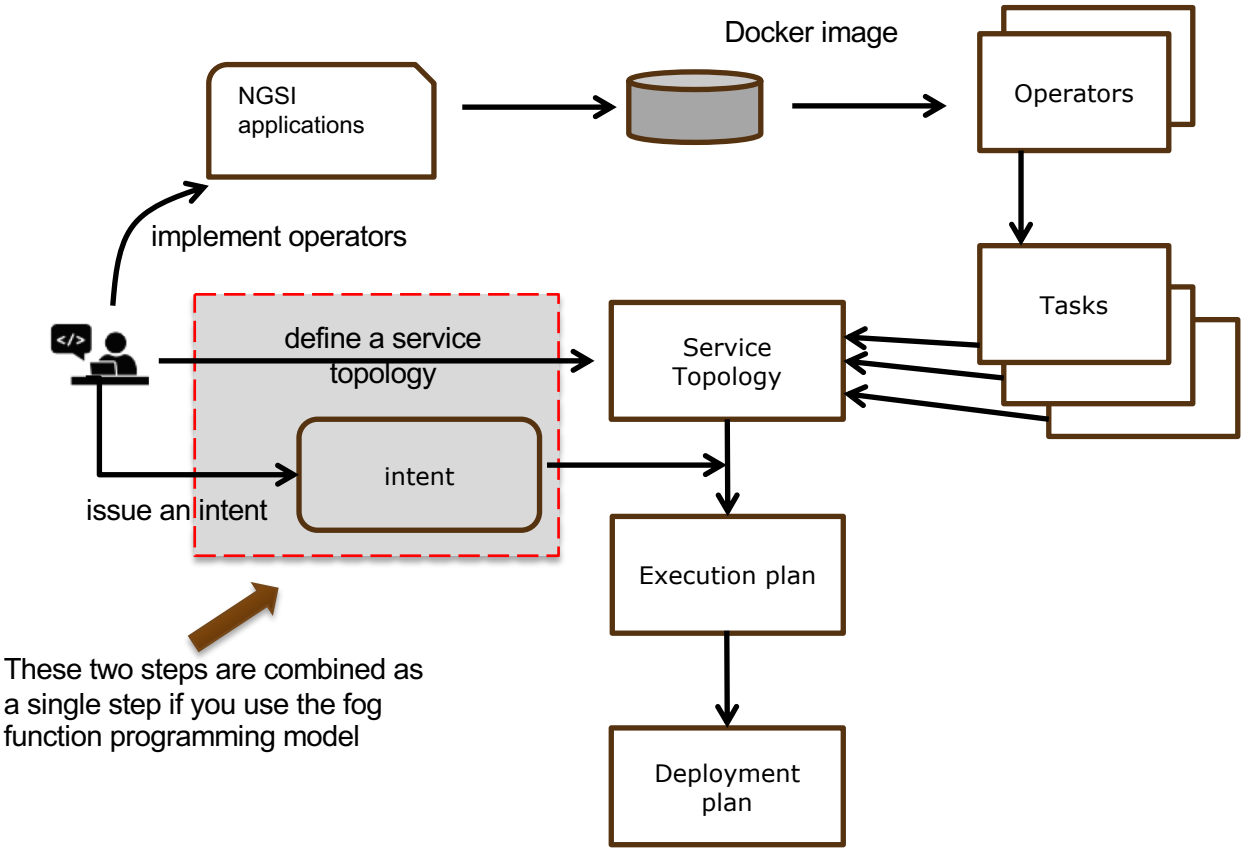


Building Apps in FogFlow

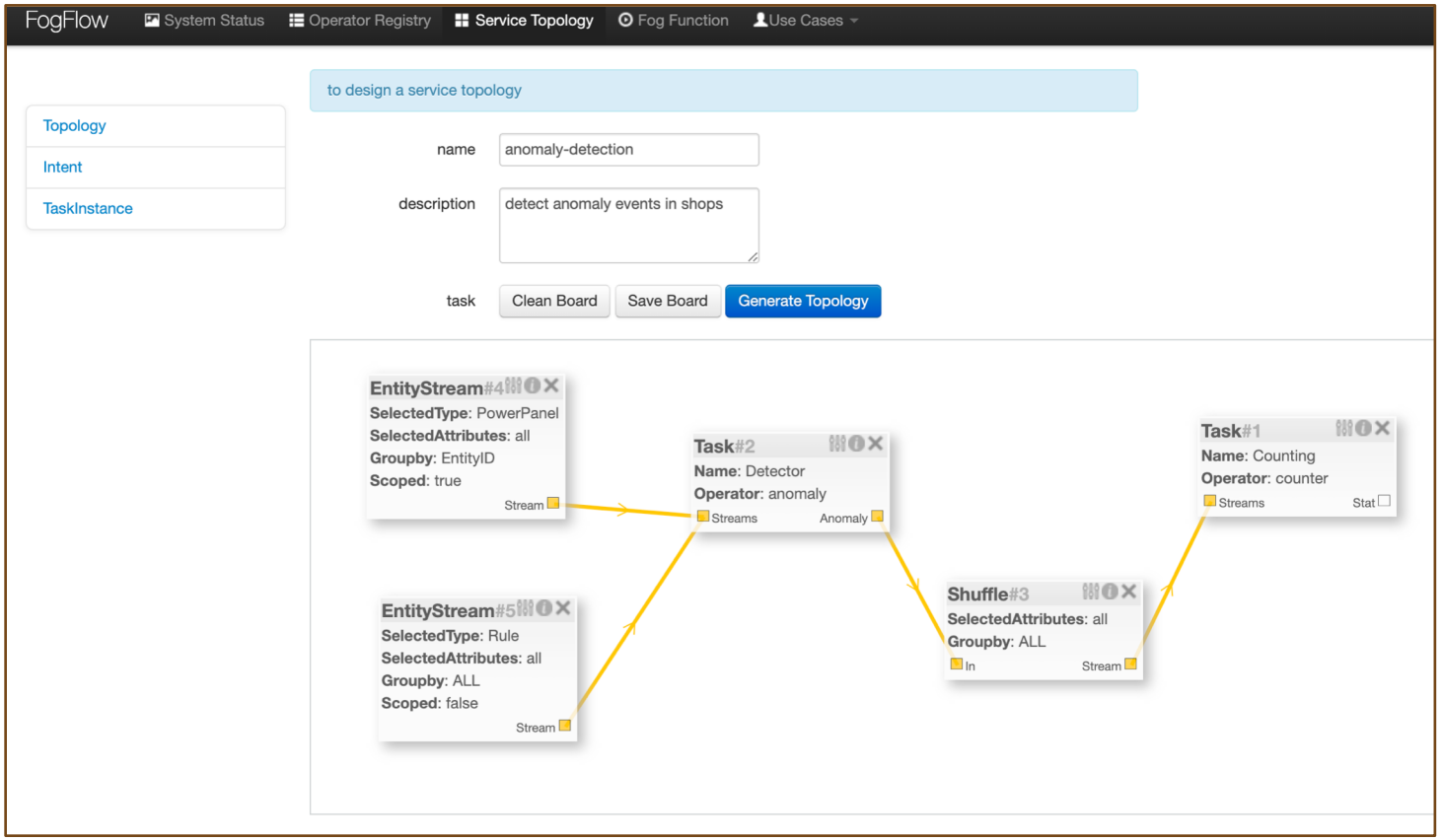
Three Elements to Program FogFlow Services



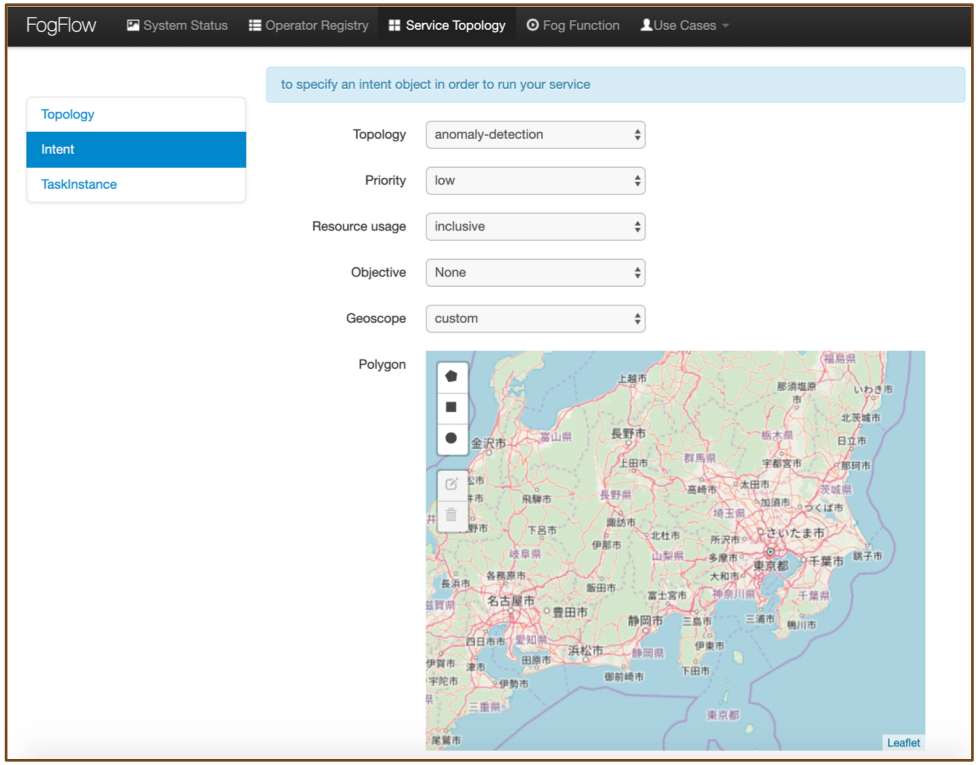
Development Process for a FogFlow Service

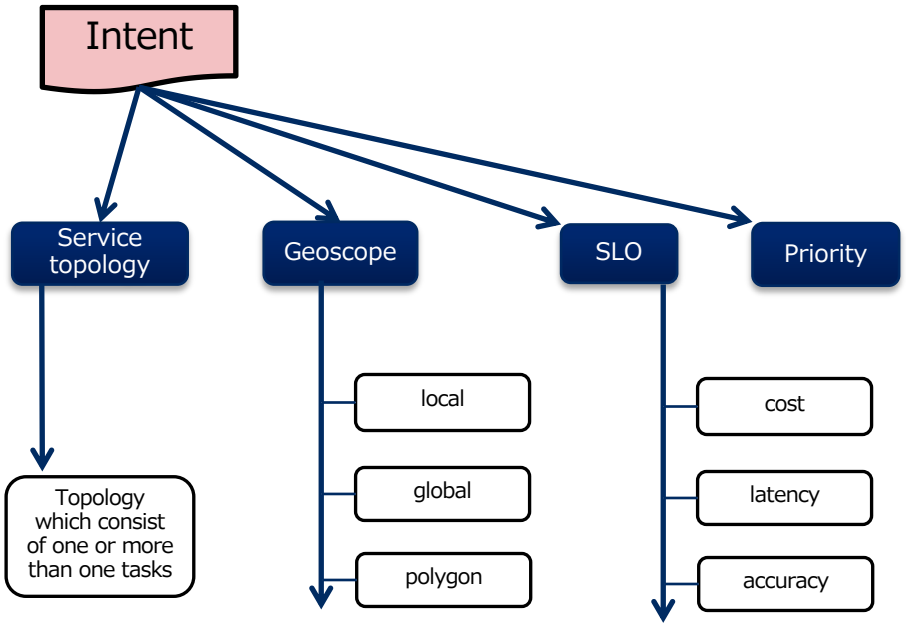


Define a Service Topology

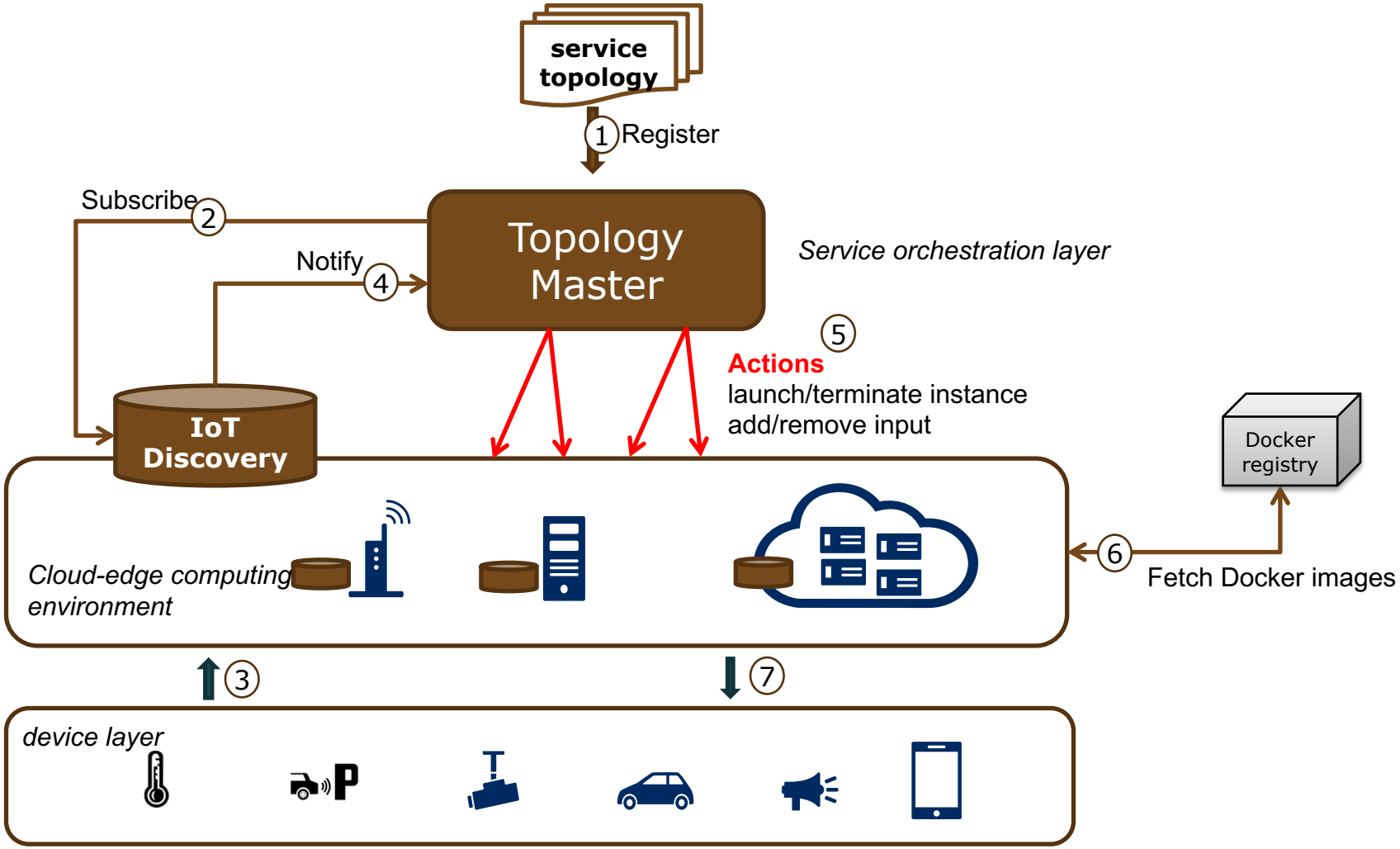


Define an intent object to trigger the service topology



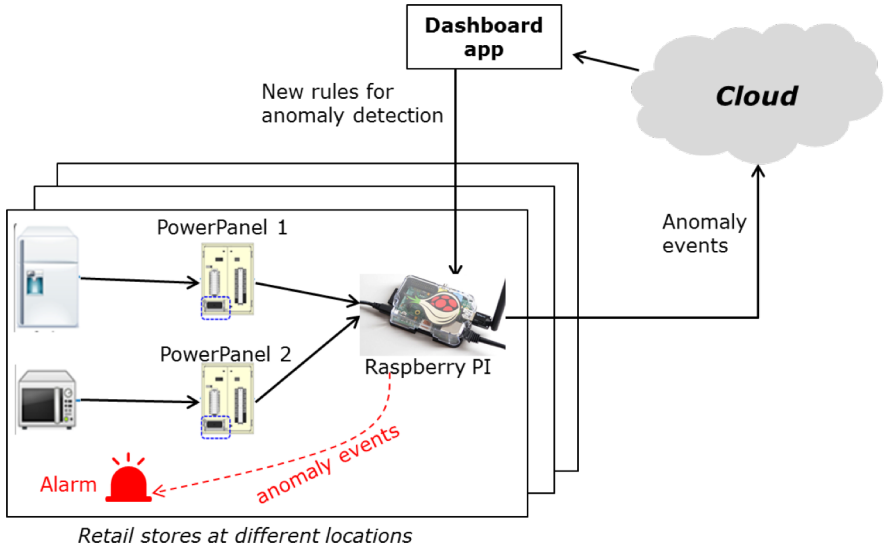


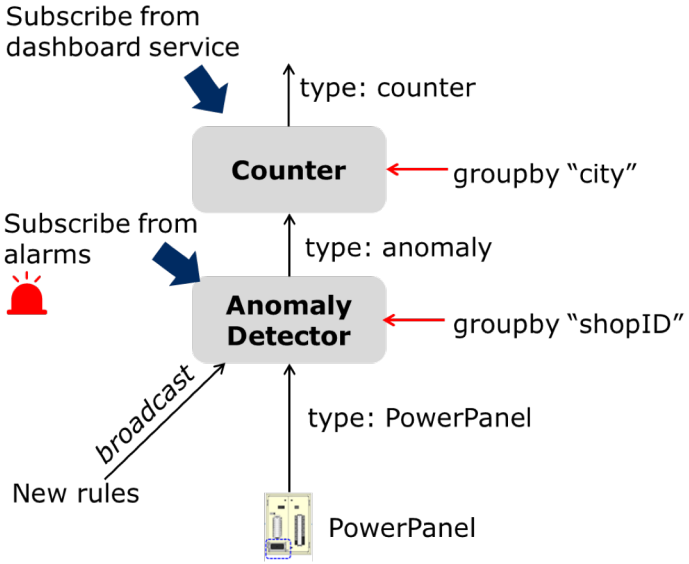
Topology Master as Service Orchestrator



Use Cases

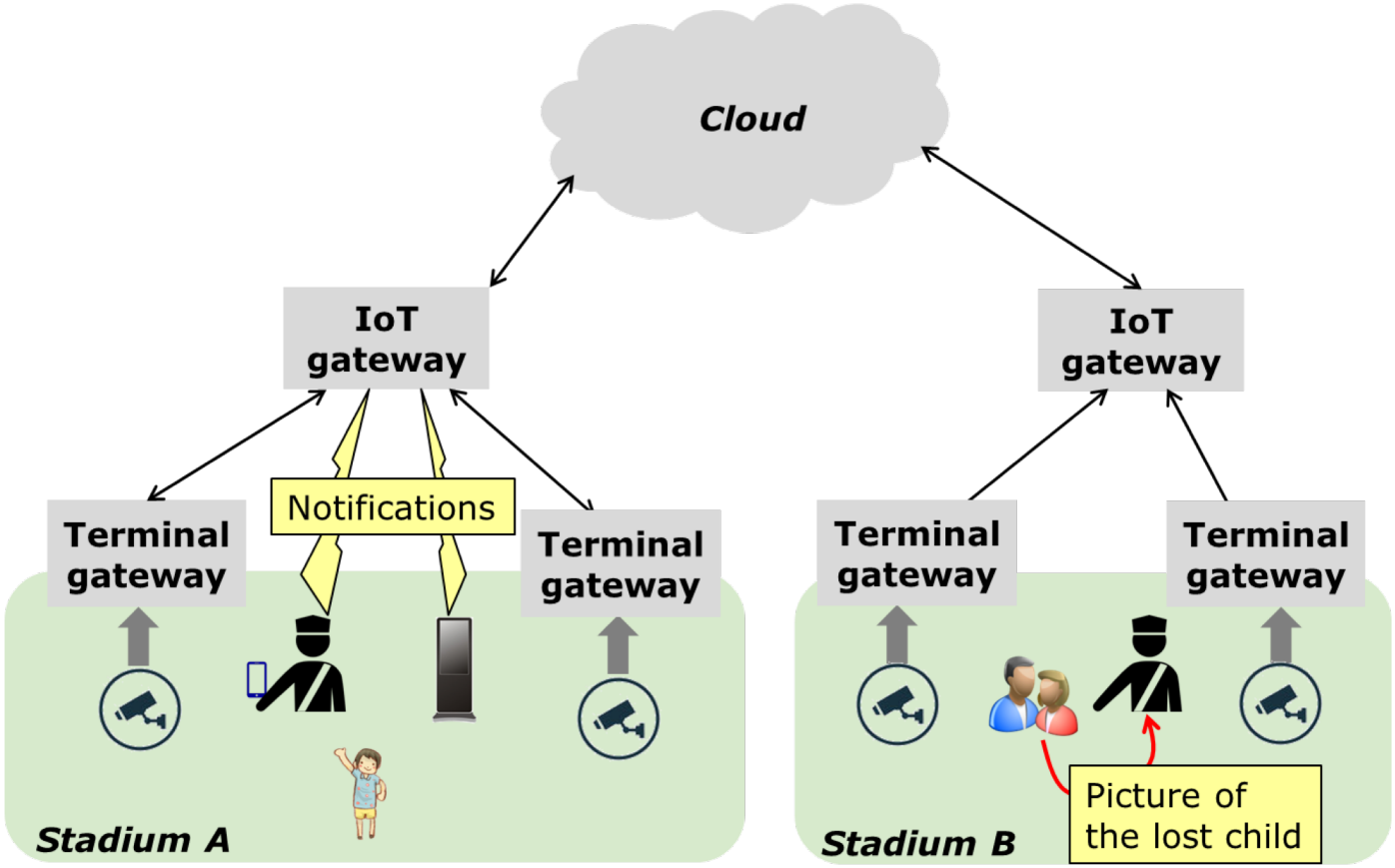
(1) Anomaly Detection in Shops





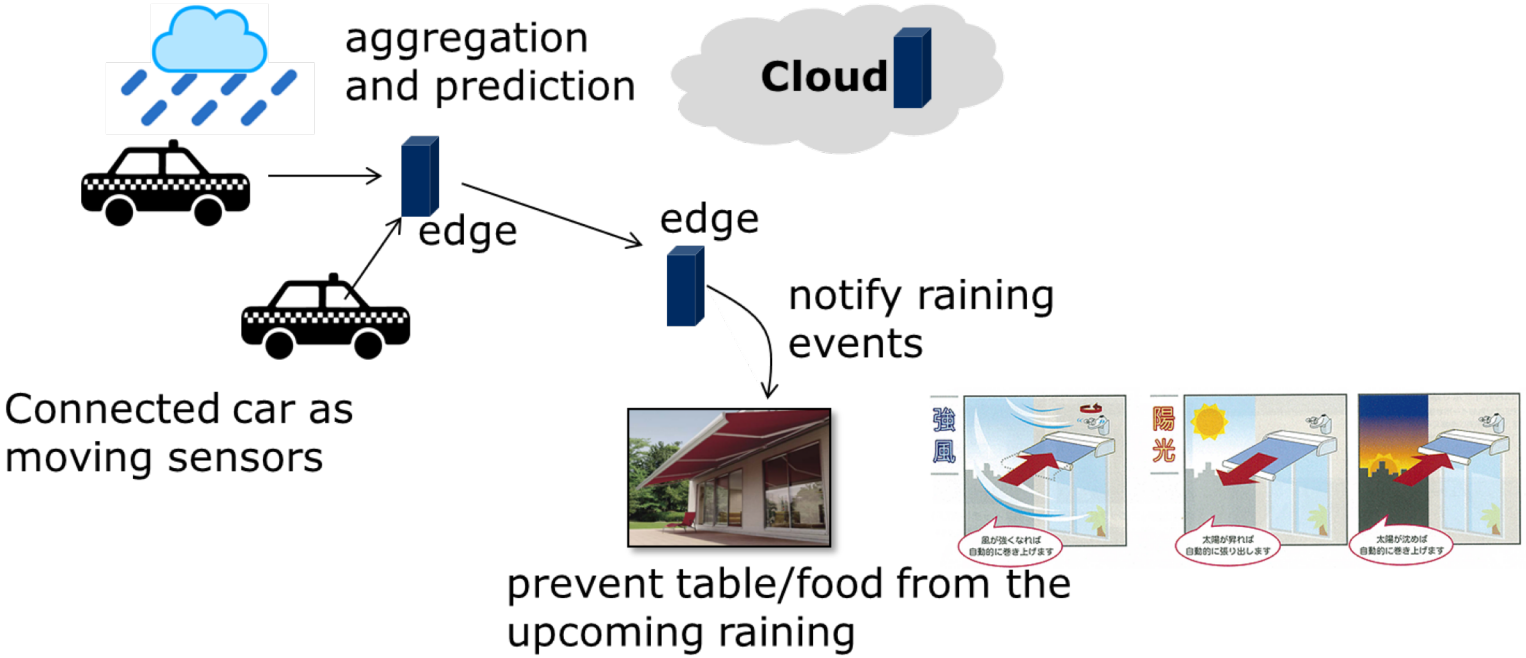
(2) Lost Child Finder

1. Find the child faster and reduce 99% bandwidth as compared to cloud-based
2. Reduce development and operation cost

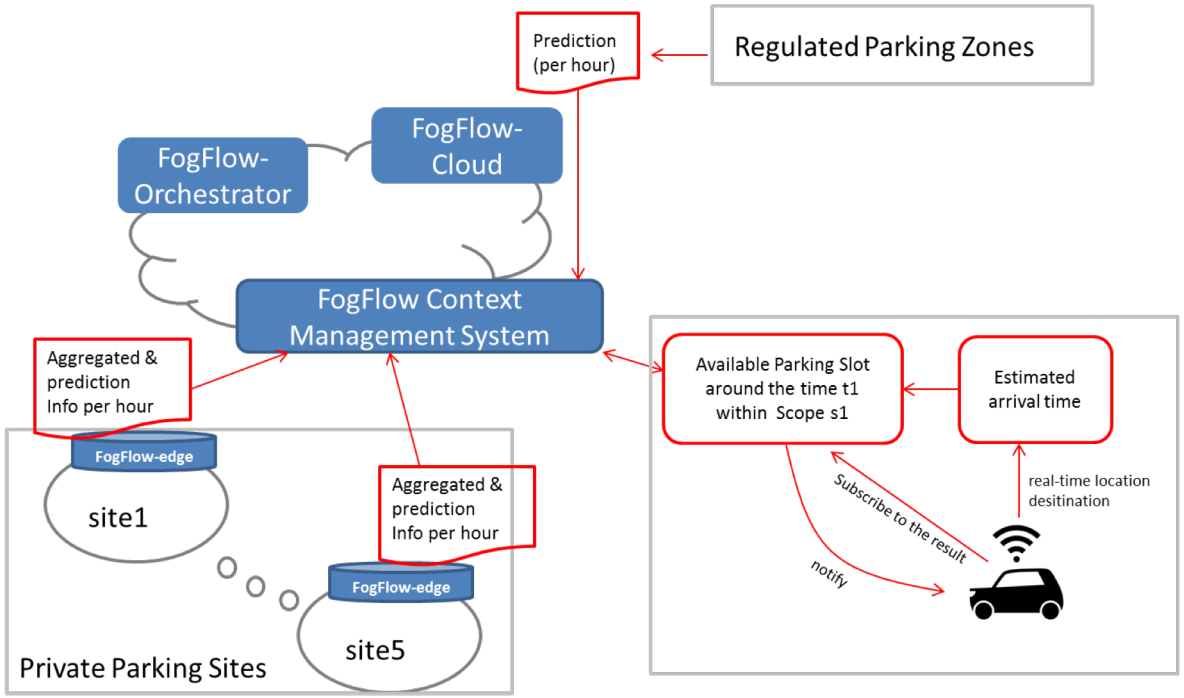


(3) Smart Awning

1. Enable faster response time with lower resource usage
2. Nearly zero management cost

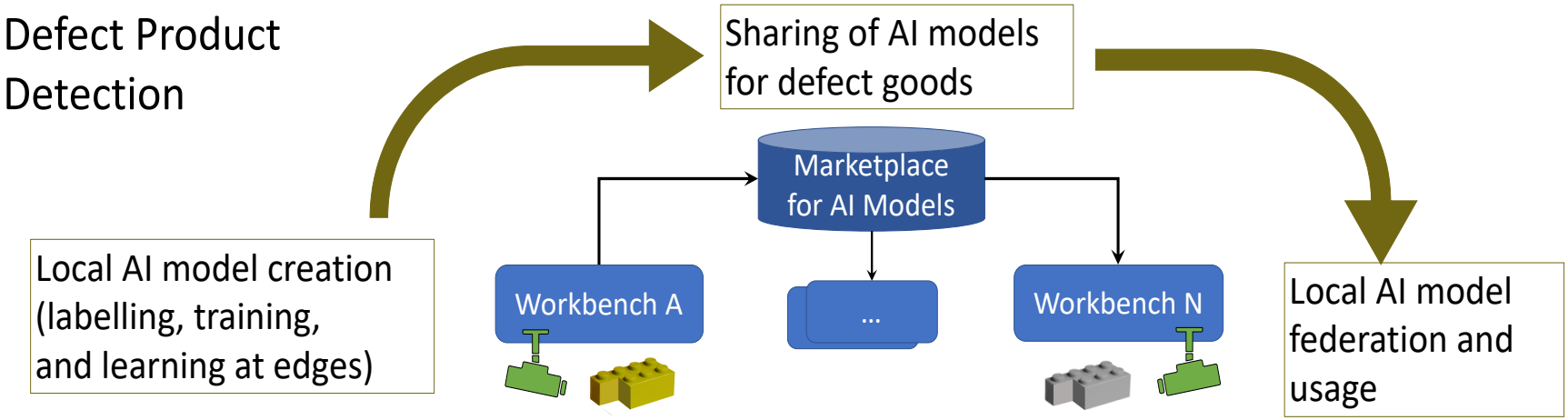


(4) Smart Parking



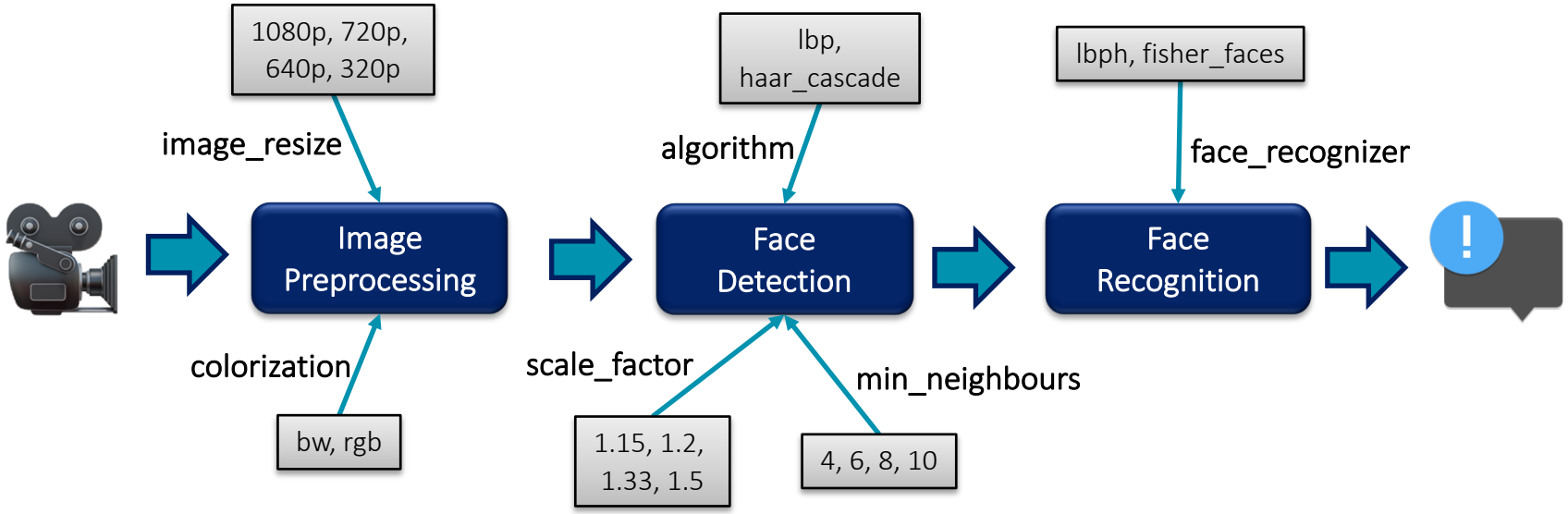
1. Leverage both public open data and the data sources from private parking sites which do not share data to the cloud, rather in their own controllable local sites
2. Reduce cost for personalized real-time park recommendation

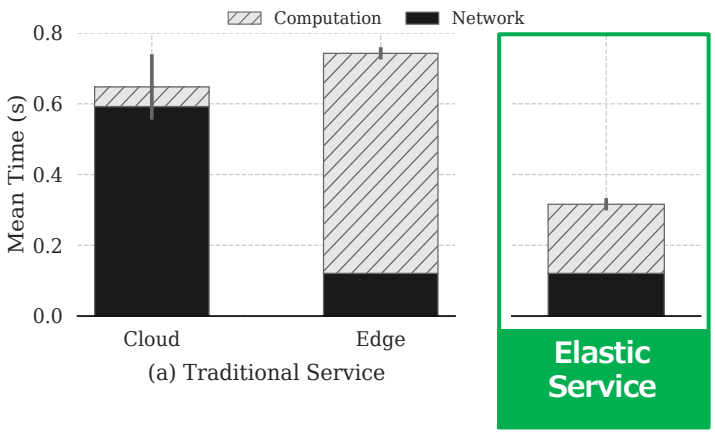
(5) Defect Product Detection



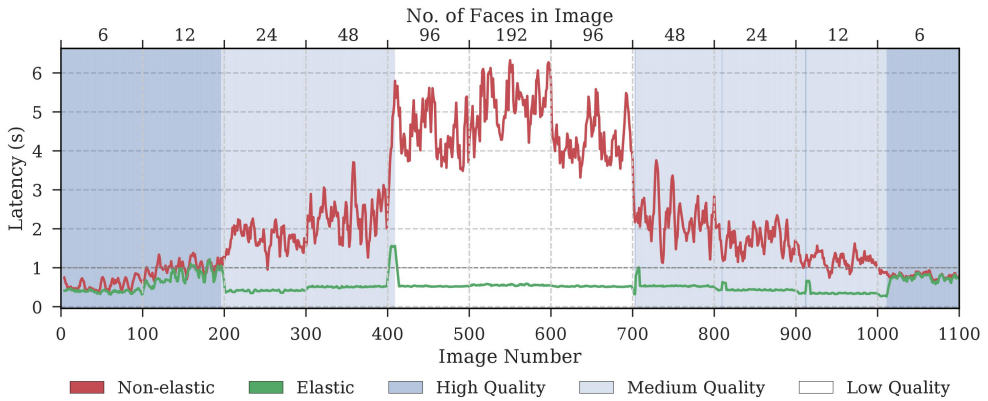
Wrap Up

Ongoing Work: Elastic Services and Reinforcement Learning





Elastic services achieve a lower latency than traditional services, by adapting the task to be run on the Edge.



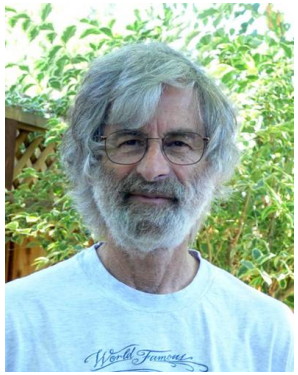
Elastic Service Adapts to Changing Inputs.

Take-aways

- *Edge computing* is a concept that brings computational resources (cloud resources) closer to the device (sensors, mobile phones, drones, cars...) to overcome **latency**, **bandwidth**, **reliability** and **privacy** limitations of cloud computing, combined with added hardware heterogeneity.
- *FogFlow* is an edge computing framework developed in NEC Labs and open-sourced as FIWARE component. Multiple successful deployments for smart city and smart industry use cases.
- Increased complexity of processing pipelines, coupled with strict latency requirements require even more advanced dynamic adaptation such as *Elastic Services*.

"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable." --Leslie Lamport

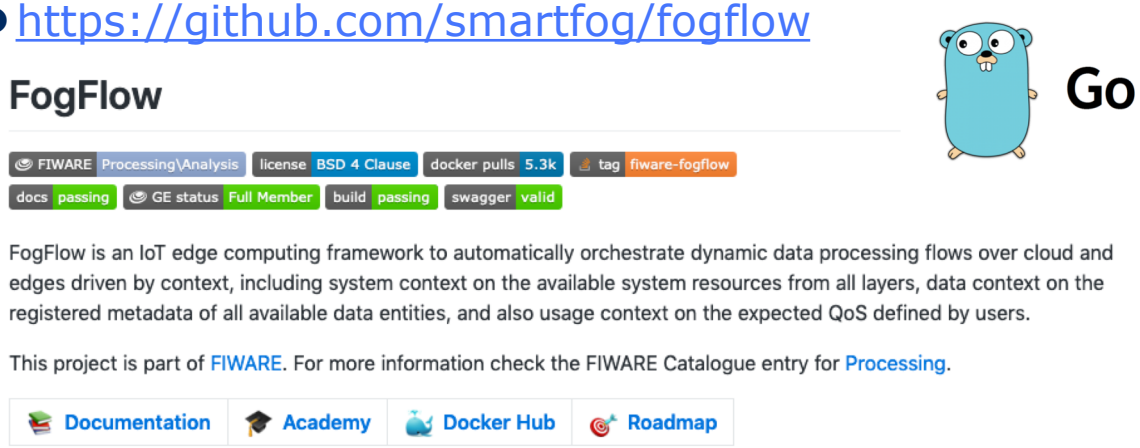
ACM Turing Award (2013)



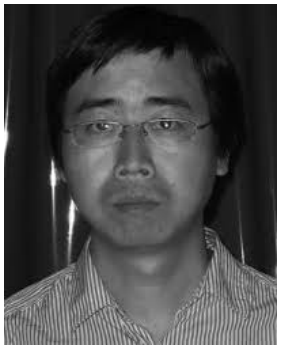
Further Reads

Code and documentation (BSD-4 licensed):

- https://github.com/smartfog/fogflow



Go



Bin Cheng
bin.cheng@neclab.eu

Technical papers

- Cheng, Bin, et al. "Fog Function: Serverless Fog Computing for Data Intensive IoT Services." *2019 IEEE International Conference on Services Computing (SCC)*. IEEE, 2019.
- Cheng, Bin, et al. "FogFlow: Easy programming of IoT services over cloud and edges for smart cities." *IEEE Internet of Things Journal* 5.2 (2017): 696-707.
- Fürst, Jonathan, et al. "Elastic Services for Edge Computing." *2018 14th International Conference on Network and Service Management (CNSM)*. IEEE, 2018.
- Fürst, Jonathan, et al. "Towards adaptive actors for scalable iot applications at the edge." *Open Journal of Internet Of Things (OJIOT)* 4.1 (2018): 70-86.
- Argerich, M. Fadel, Bin Cheng, and Jonathan Fürst. "Reinforcement Learning based Orchestration for Elastic Services." *arXiv preprint arXiv:1904.12676* (2019).

Contact Information

NEC Laboratories Europe

jonathan.fuerst@neclab.eu

https://www.jofu.org/

@JF87

Jonathan Fürst