Ensembles of IoT, Network functions and Clouds

Hong-Linh Truong

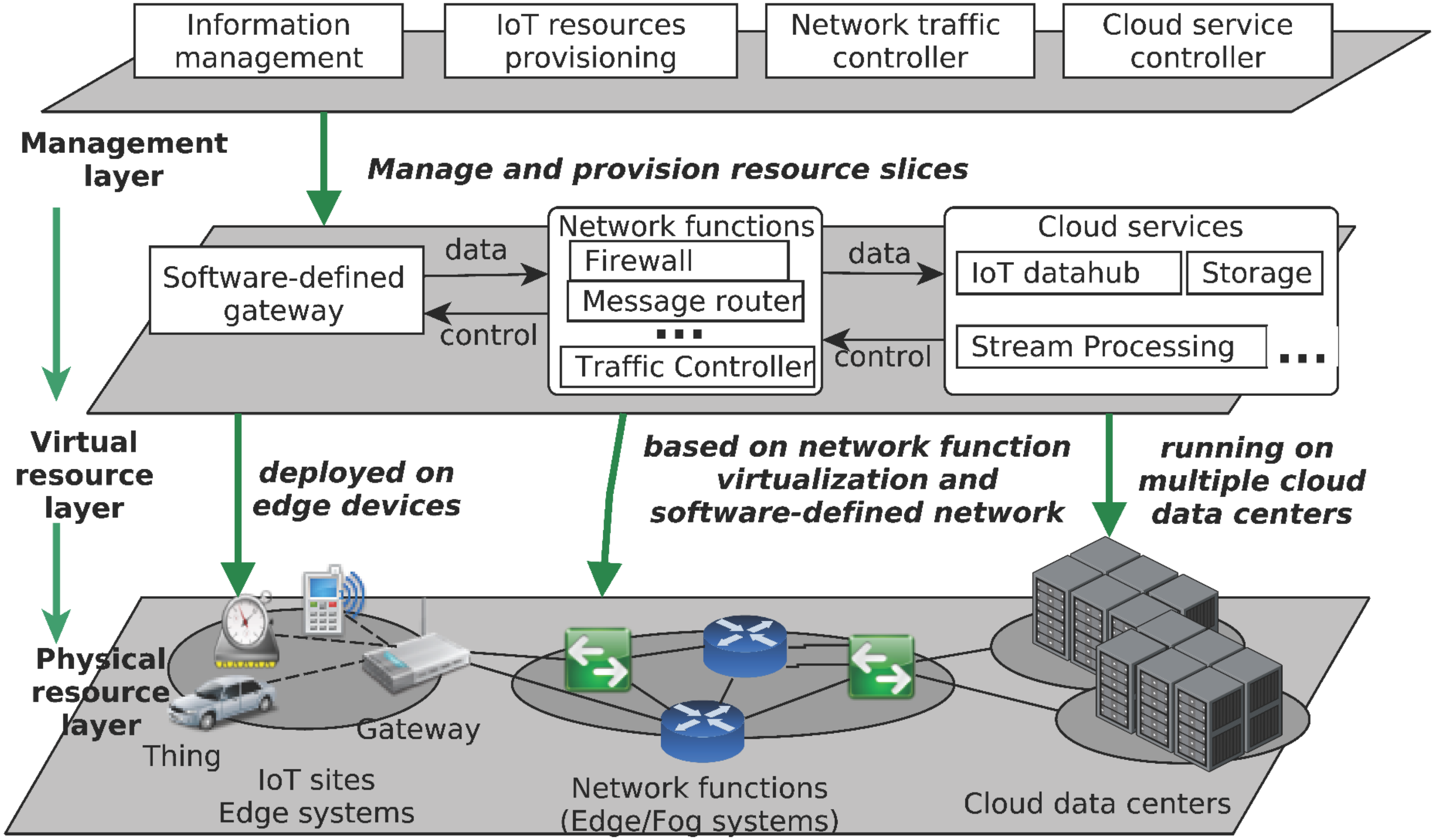
Faculty of Informatics, TU Wien
hong-linh.truong@tuwien.ac.at
http://rdsea.github.io

Outline

- Scenarios
- Ensemble models
- Resource harmonization
- Interoperable resource slices
- Conclusions and future directions

Application-oriented Ensembles of IoT, Network functions, & Clouds: Necessity?

Systems of IoT, Network Functions, and Clouds



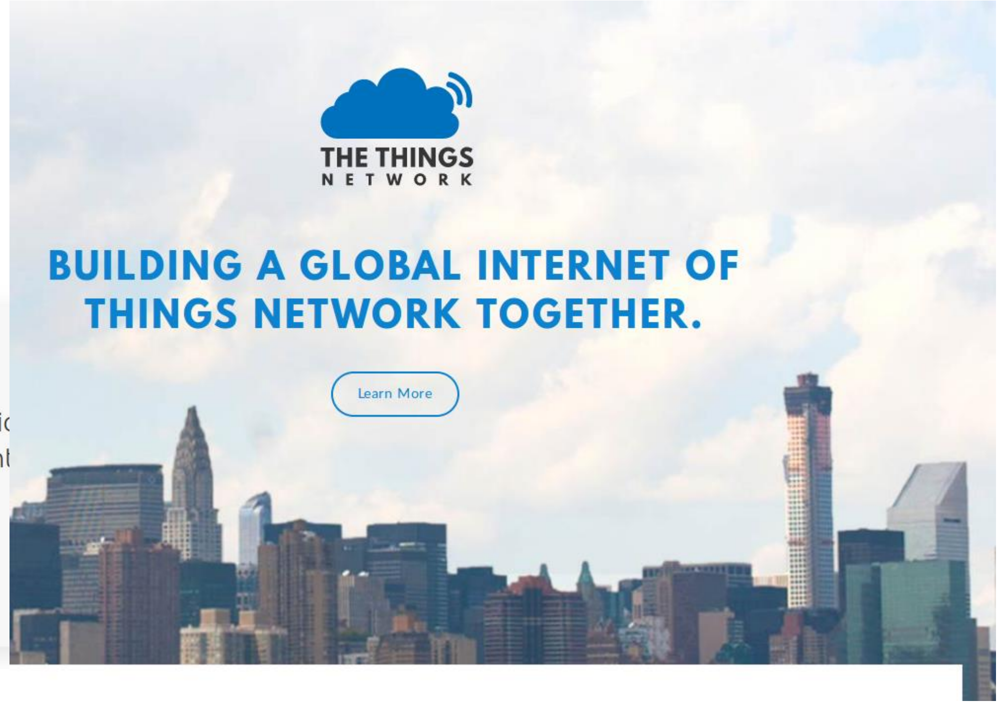
Moving to blending IoT, network functions and cloud resources

- Type 1
  - **Mainly focus** on IoT networks: sensors, IoT gateways, IoT-to-cloud connectivity (e.g., connect to predix.io, IBM Bluemix, Azure IoT, Google Cloud, Amazon IoT, etc.)
- Type 2
  - **Mainly focus** on (public/private) services in data centers: e.g., IoT data hubs, NoSQL databases, and big data ingest systems
- Type 3
  - **Equally focus** on both IoT and cloud sides and have the need to control at both sides
  - Highly interactions between the two sides, including the network in the middle

All types of service models

- Cloud resources as services are known
- NFV/5G coming and more service providers at the edge are available
  - Network function virtualization
- IoT infrastructure-as-a-service
  - Pay-per-use IoT communication
- IoT data-as-a-service Cloud service models
  - Public and private providers

Examples of IoT networks



Enabled by Semtech's LoRa technology, which allows a sensor network to operate on low power while providing strong connectivity over a long distance, the SK Telecom LPWAN covers 99 percent of South Korea's population. The company predicts that it will have over four million things connected to its IoT networks by the end of 2017.

The LoRaWAN™ technology platform based on Semtech's LoRa two-way wireless RF technology is a comprehensive solution for low power, long-range connectivity, which is essential for LPWANs supporting IoT applications. It offers deep penetrability, secure connectivity, long battery lifetime and streamlined implementation for simple network rollout, as well as integration into existing infrastructure.

http://www.sktelecom.com/en/press/detail.do?idx=1172

Price Plan	Data Allowance* (Frequency of communication)	Monthly Flat Rate (VAT Excluded)	Examples of Services	Note
Band IoT 35	100KB	KRW 350	Metering and monitoring services (e.g. Advanced Metering Infrastructure (AMI), environmental monitoring, water leakage monitoring, etc.)	- Discount benefits for long-term contracts: Ranging from a 5% discount for two-year contracts to a 20% discount for 5 year-contracts - Multi-line discount: Ranging from a 2% discount for those using 500 lines to a 10% discount to those who use 10,000 lines
Band IoT 50	500KB	KRW 500		
Band IoT 70	3MB	KRW 700	Tracking services (e.g. locating tracking For people/things, asset management, etc.)	
Band IoT 100	10MB	KRW 1,000		
Band IoT 150	50MB	KRW 1,500	Control service (e.g. safety management, lighting control, shared parking, etc.)	
Band IoT 200	100MB	KRW 2,000		

*Data usage exceeding the data allotment provided will be charged at KRW 0.005 per 0.5KB.

Some application scenarios

- Emergency responses, on-demand crowd sensing, Geo Sports monitoring, cyber-physical systems testing, etc.

Geo Sports: Picture courtesy Future Position X, Sweden

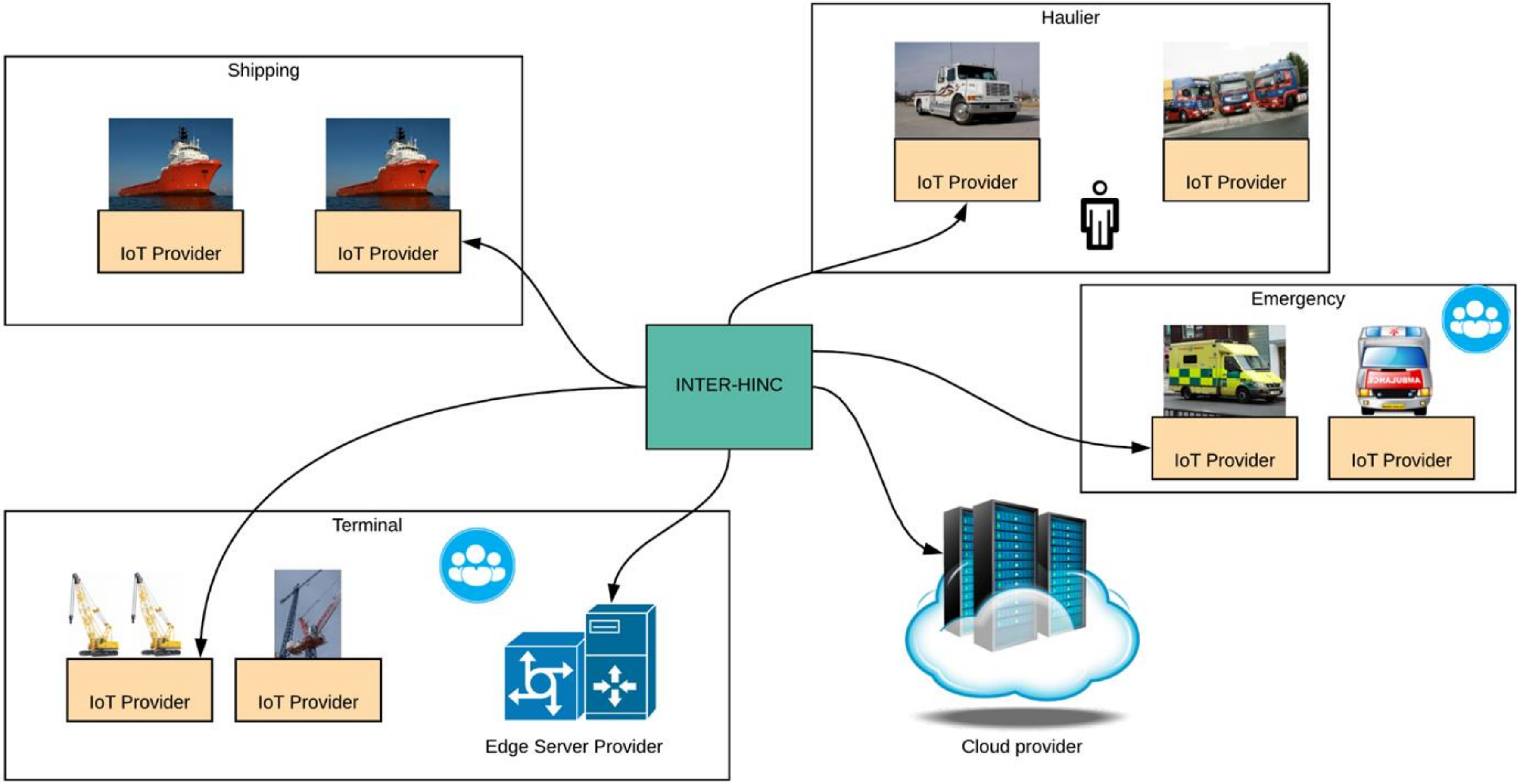
Seaport Scenarios

http://inter-iot-project.eu/

- Built a top existing INTER-IoT scenarios
  - Coordinating activities when an accident happens
- Example of actions in a resource slice in a seaport:

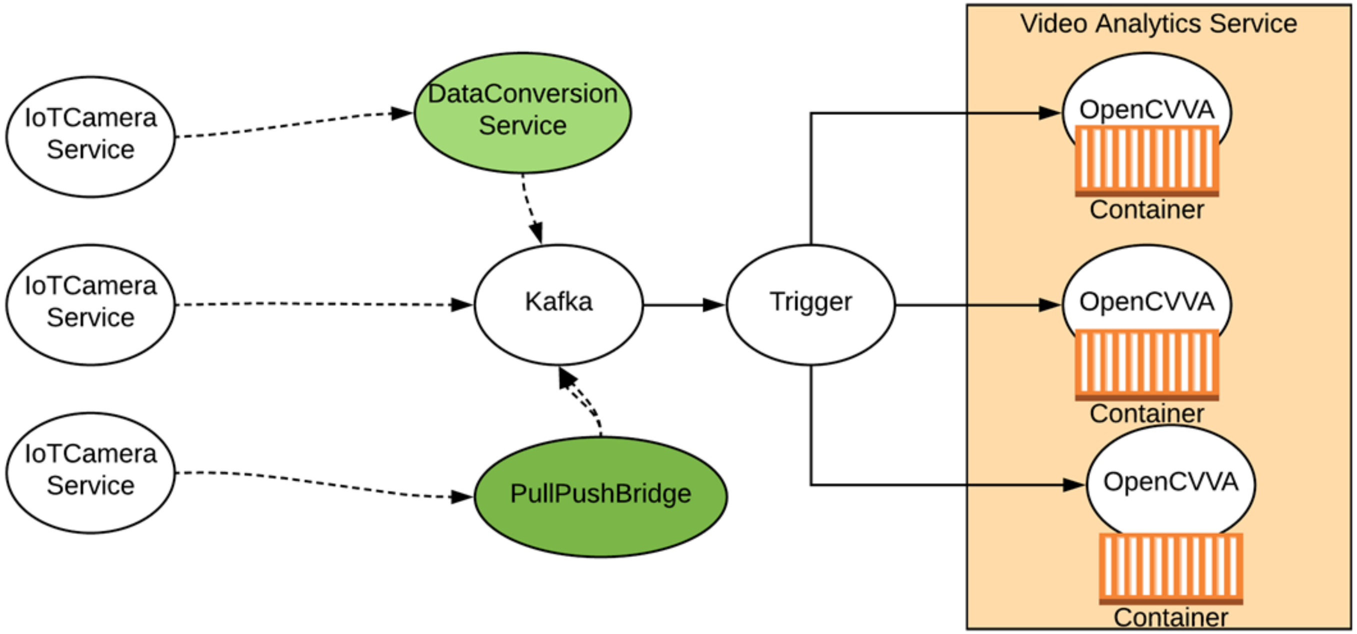
1) activating monitoring containers with sensitive goods in the port
2) analyzing and controlling robotic cranes and trucks to make sure that they do not prevent the emergency responses as well as ready to support the responses
3) sending alarms and controlling vessel arrivals and revising transport planning,
4) providing information for operational assistance for the emergency responses
5) activating systems to support the monitoring of people impacted by the accident using devices and platforms for chronic disease and cognitive decline prevention.

Seaport Scenarios



- Similar issues: data contract, conversion, quality of data, and transfer protocols
- But interoperability cannot work without *network engineering and data protection*

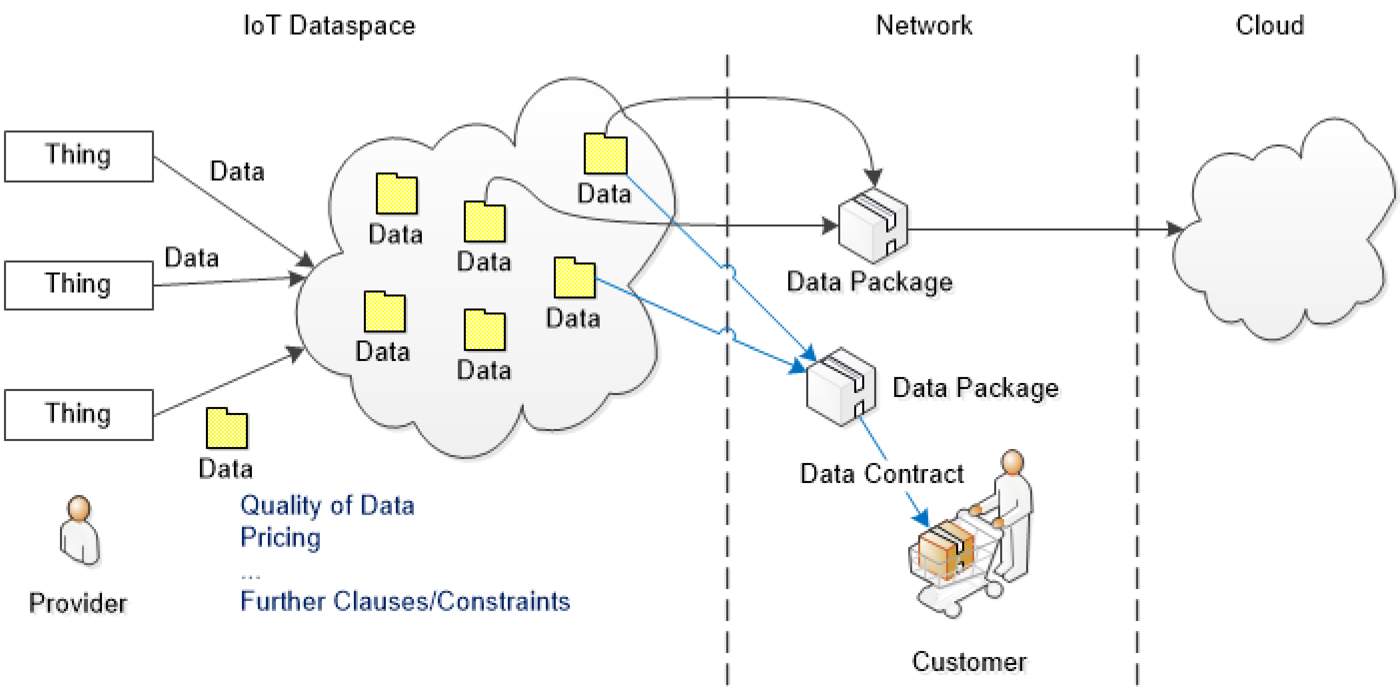
Video Analytics in Smart Cities



- *Many things can establish very fast*: querying camera, downloading video, transferring data and launching analytic components
- Large-scale on-demand video analytics need *responsive and reactive interoperability* w.r.t data conversion, protocol translations and data contracts

Examples of IoT data services

IoT data as a service can be offered by different types of providers



Florin-Bogdan Balint, Hong-Linh Truong, On Supporting Contract-aware IoT Dataspace Services, the 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud 2017), 6-8 April 2017 in San Francisco, USA
.

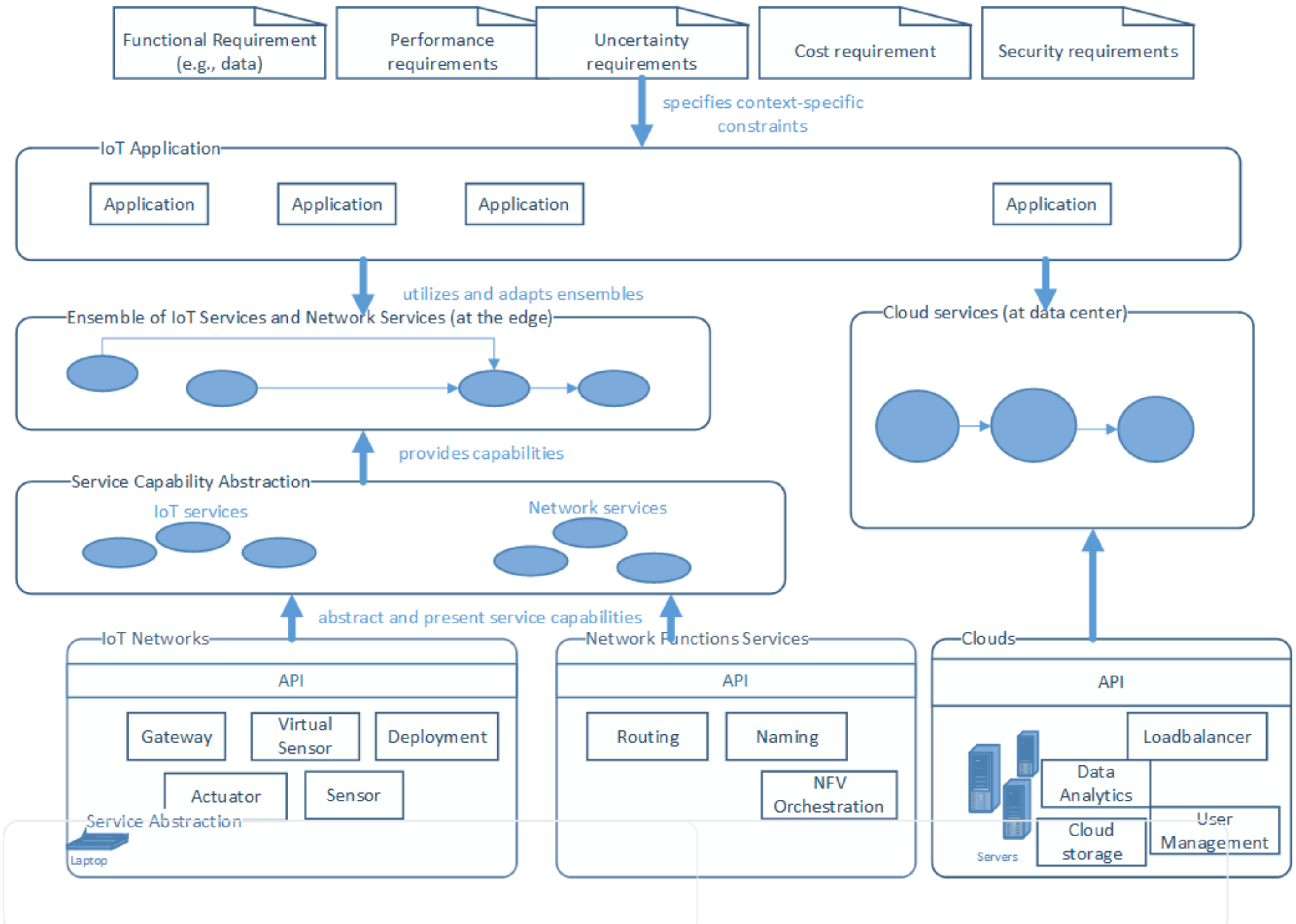
Tools treating IoT, network functions, and cloud services in an isolated manner are not enough

EMSEMBLE MODELS

Papers and links:

- Notes on ensembles of IoT, network functions and clouds for service-oriented computing and applications, https://link.springer.com/article/10.1007%2Fs11761-018-0228-2
- Hong Linh Truong, Nanjangud C. Narendra: SINC - An Information-Centric Approach for End-to-End IoT Cloud Resource Provisioning. ICCCRI 2016: 17-24
- http://sincconcept.github.io

Ensembles of IoT, Network functions and clouds



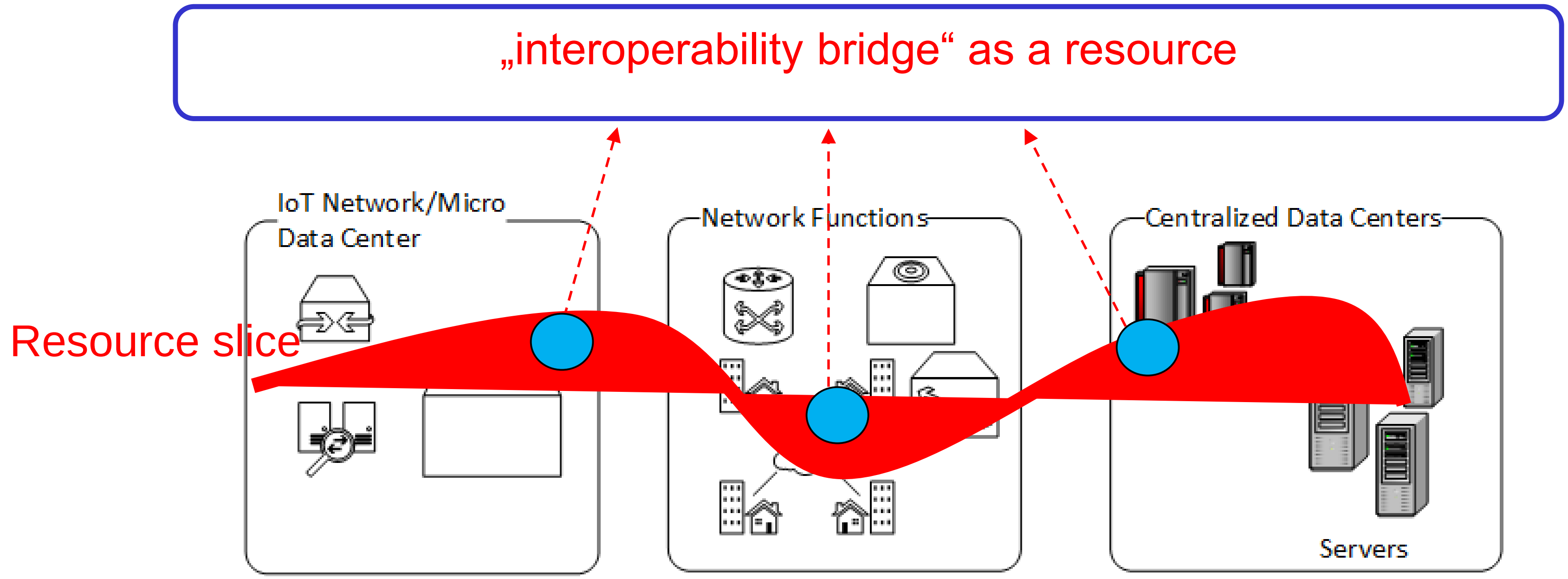
Challenges

- Modeling distributed IoT, network functions and cloud capabilities in an ensemble
- Slicing end-to-end network of resources
- Composing resources in ensembles of IoT, network functions and clouds
- (Re-)configuring composed resources
- Testing and monitoring

The SINConcept: http://sincconcept.github.io/
Hong-Linh Truong, Nanjangud Narendra, SINC - An Information-Centric Approach for End-to-End IoT Cloud Resource Provisioning, 2016 International Conference on Cloud Computing Research & Innovation (ICCCRI2016), CloudAsia 2016, May 3-5, 2016, Singapore

Ensemble and its resource slice

- Resources: data streams, analytics, broker, storage, etc.
- Any existing software and service supporting interoperability is also a resource
- A composition of "non-Interoperable components" can create a virtual resource for interoperability



Resource slice concept and related papers: http://sincconcept.github.io/

Resource Management, Configuration and Adaptation (1)

- Creating slices, each slice includes a set of partitions of resources
  - Modeling and capturing user requirements for slices
- Creation and Management
  - Develop new algorithms for creating slices by leveraging existing works for IoT, networks, and services
  - Integrate with NFV orchestrators, virtual sensors, gateways, cloud APIs and SDN controllers.
  - Deal with different resource provisioning models imposed by underlying infrastructures
  - Configuration by leveraging different deployment tools for IoT, network functions and clouds

Resource Management, Configuration and Adaptation (2)

- Monitoring and Management
  - Develop end to end metrics for slices
  - Integrate monitoring capabilities from different providers and correlating monitoring data
- Runtime slice adaptation
  - Performance as well as uncertainties at infrastructures, applications and their integration levels
  - Adaptation capabilities across IoT, network functions and clouds
  - Multiple level of adaptations based on end-to-end problems and partition problems

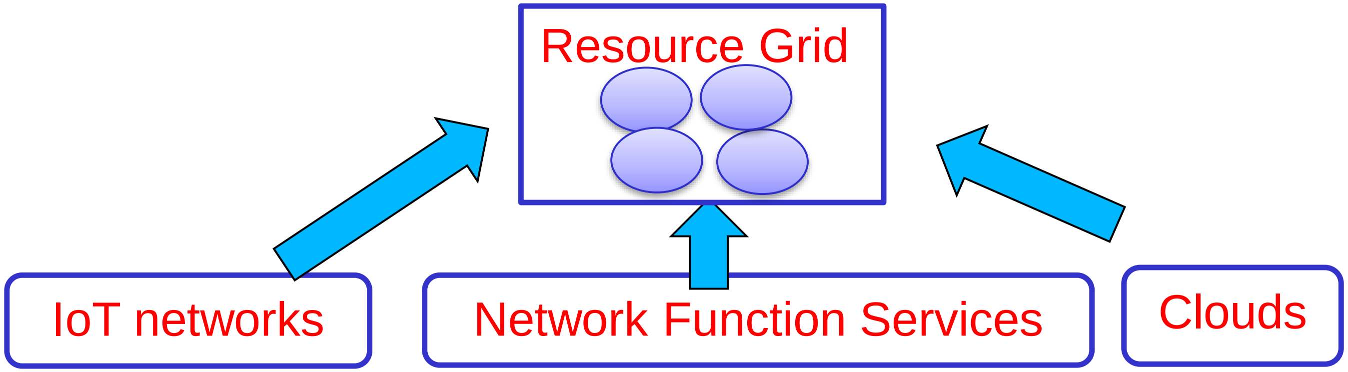
Towards the Implementation

- Using REST API to integrate resource management capabilities from different providers
- Distributed communication middleware, e.g., based on AMQP/MQTT, for querying resource information and propaging controls
- TOSCA or other topology description tools for modeling topologies for supporting configuration and deployment
- Leveraging existing deployment techniques for IoT and clouds (Ansible, Terraform, Chef, SALSA)
- Testbed established with open sources: Dockers, OpenStack, Weave, OpenDailyLight, etc. by utilizing cloud, network and IoT devices

HARMONIZING RESOURCES FOR ENSEMBLES?

[FiCloud2016] Duc-Hung Le, Nanjangud C. Narendra, Hong Linh Truong: HINC - Harmonizing Diverse Resource Information across IoT, Network Functions, and Clouds. FiCloud 2016: 317-324

Integrating diverse types of resources

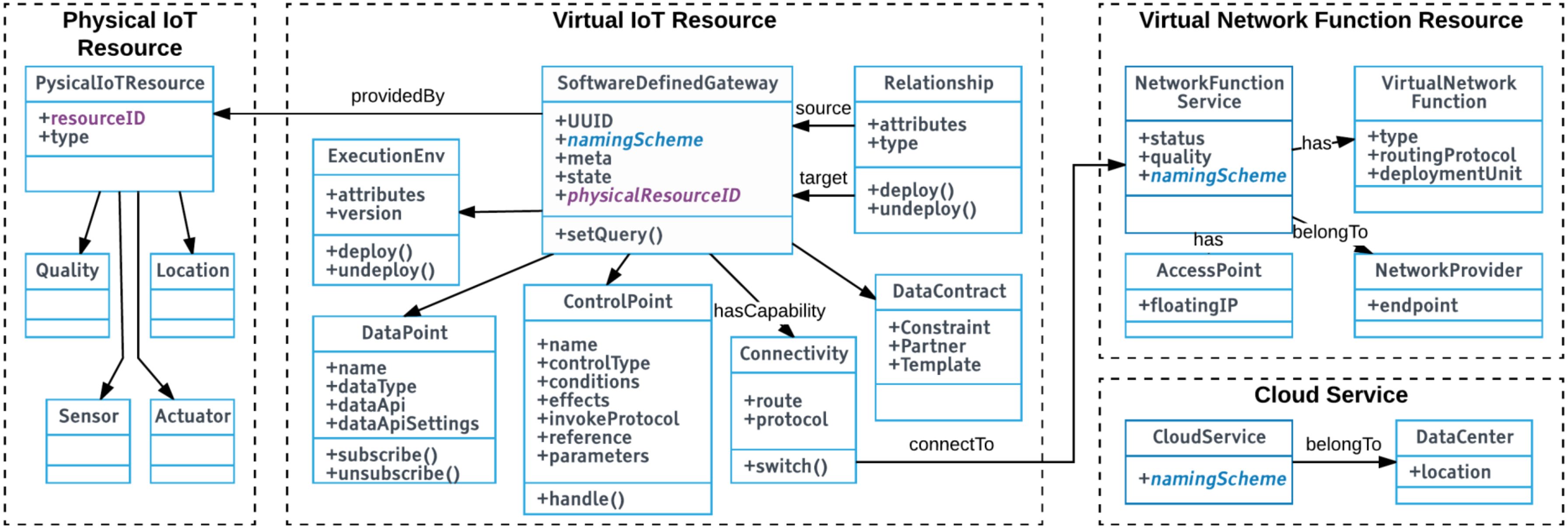


- Make a Resource Grid ready for creating ensembes
  - Harmonize IoT, network functions and cloud resources
- API Integration and Communication
  - Use REST API for obtaining metadata and control of resources
    - Sensoring data can be transferred through different middleware
  - Work with existing metamodel (IoTivity, OpenHAB, IoTDM, ETSI MANO, OCCI, CIMI, etc.)
  - Rely on scalable cloud communication middleware (e.g., AMQP & MQTT)

Examples of existing providers/models

Provider	Category	APIs	Information models
FIWare Orion	IoT	RESTful (NGSI10), one-time query or subscription	High level attributes on data and context
FIWare IDAS	IoT	RESTful for read/write custom models and assets	Low level resource model catalogs
IoTivity	IoT	REST-like OIC protocol, support C++, Java and JavaScript	Multiple OIC model
OpenHAB	IoT	RESTful for query and control IoT resources	Low level resource model catalogs
OpenDayLight	Network	Dynamic REST generated from Yang model (model-driven)	Low level resource model catalogs
OpenBaton	Network	RESTful for network service description	ETSI MANO v1.1.1 data model
OpenStack	Cloud	RESTful, multiple language via SDK, OCCI, CIMI	OpenStack model, OCCI, CIMI

Information model



Physical: Sensor/actualtor/devices in providers' models

Virtual IoT: SD-Gateway and capabilities.

Network functions: edge-to-edge, edge-to-cloud network.

Clouds: VM, data services, data analytics.

Reducing complexity in accessing and control resources

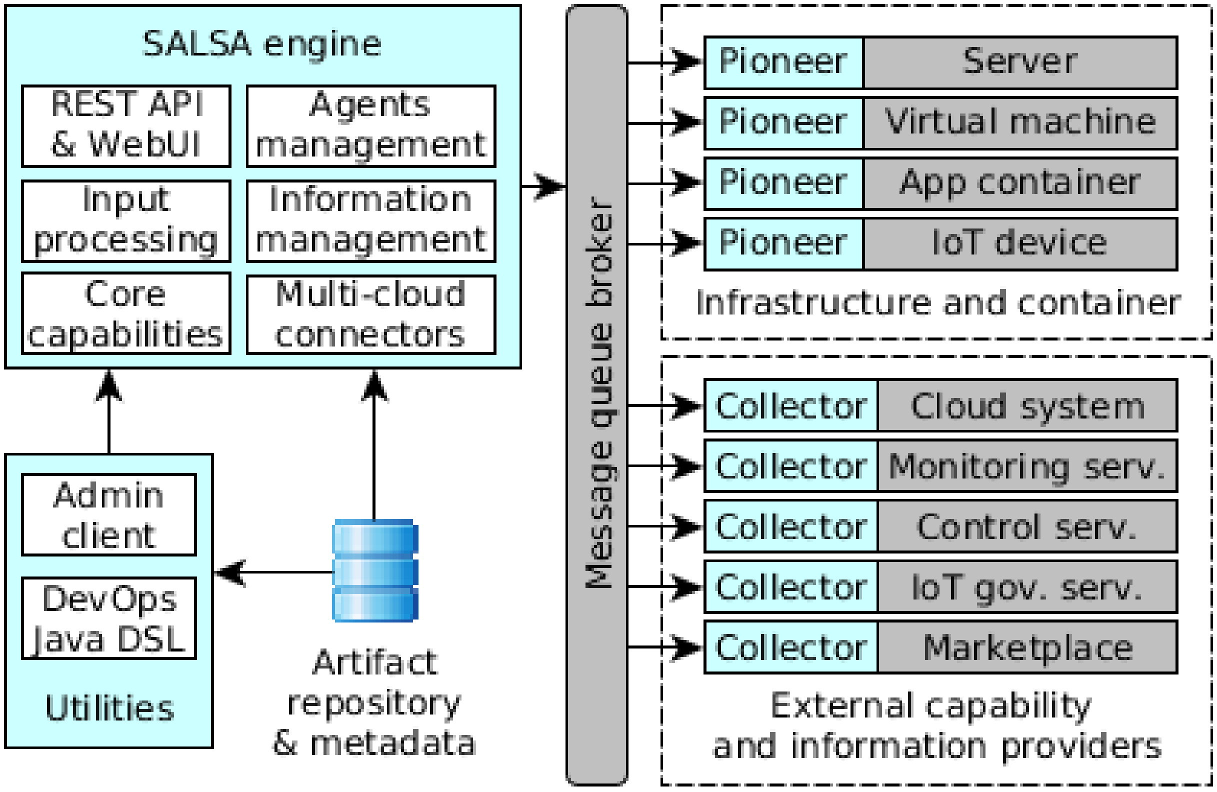
```
// specify the sensors whose sample rate needs to be changed
DataPoint template = new DataPoint('BodyTemperature');
SensorProps sensorProps = new SensorProps();
sensorProps.setRate(5);
template.getExtra().add(new PhysicalResource(sensorProps));
QueryManager queryMng = new QueryManager('ampq://10.99..');
List datapoints = queryMng.QueryDataPoints(template);

// observe the resource and send back the control
for (DataPoint dp : datapoints) {
 DataPointObservator obs = new DataPointObservator(dp) {
  public void onChange(DataPoint newVal) {
   SensorProps props = newVal.getExtra('SensorProps');
   if (props.getRate() > 5) { props.setRate(5); }
   ControlPoint control = newVal.getControl('changeRate',5);
   queryMng.SendControl(control);
 } }; }

// create the template of the query
NetworkFunctionService nfsTemp = new NetworkFunctionService(
nfsTemp.getQuality().setBandwidth("16 GB/s");
CloudService cloudTemplate = new CloudService('storage');
cloudTemplate.hasAttribute("capacity", "1 TB");

// Query the list of resources
QueryManager queryMng = new QueryManager('ampq://10.99..');
List networks = queryMng.QueryNetworkFunctionService(nfsTemp
List storages = queryMng.QueryCloudService(cloudTemplate);
// further queries
```

1. Query data points

2. Control the resource

3. Query network functions and clouds

As we able to get resources → deploy and configuration



Deployment architecture of SALSA.

http://tuwiendsg.github.io/SALSA/

INTEROPERABLE RESOURCE SLICE FOR ENSEMBLES

Hong-Linh Truong, Towards a Resource Slice Interoperability Hub for IoT, 3rd edition of Globe-IoT 2018: Towards Global Interoperability among IoT Systems, IEEE, 2018. Orlando, Florida, USA, April 17-20, 2018.

http://www.infosys.tuwien.ac.at/staff/truong/publications/2018/rsihub_draft_jan18.pdf

Key requirements

- Cross systems and cross layers:
  - Interoperability is not just needed IoT resources
  - We need IoT, network functions and cloud resources
- A combination of issues w.r.t. data, protocols, contracts, service quality, etc.
- Developer goals:
  - Reduce time to establishing interoperability solutions
  - Configure suitable components to deal with interoperability
  - Deploy and provision interoperable bridges
  - Address application-specific concerns
  - Aim to develop full automatic solutions

Key concepts

- Client `c` specifies a resource slice: `RS(c)`
- We make the resource slice interoperable, creating `RSi(c)` from `RS(c)`
- We focus on resources can be represented by data points and control points with suitable metadata: resource is `r(DP,CP,MT)`
  - DP (data points), CP (control points), MT (metadata)
  - A service provider must provide enough metadata of its resources
  - We must be able to deploy a software artifact supporting interoperability on-demand

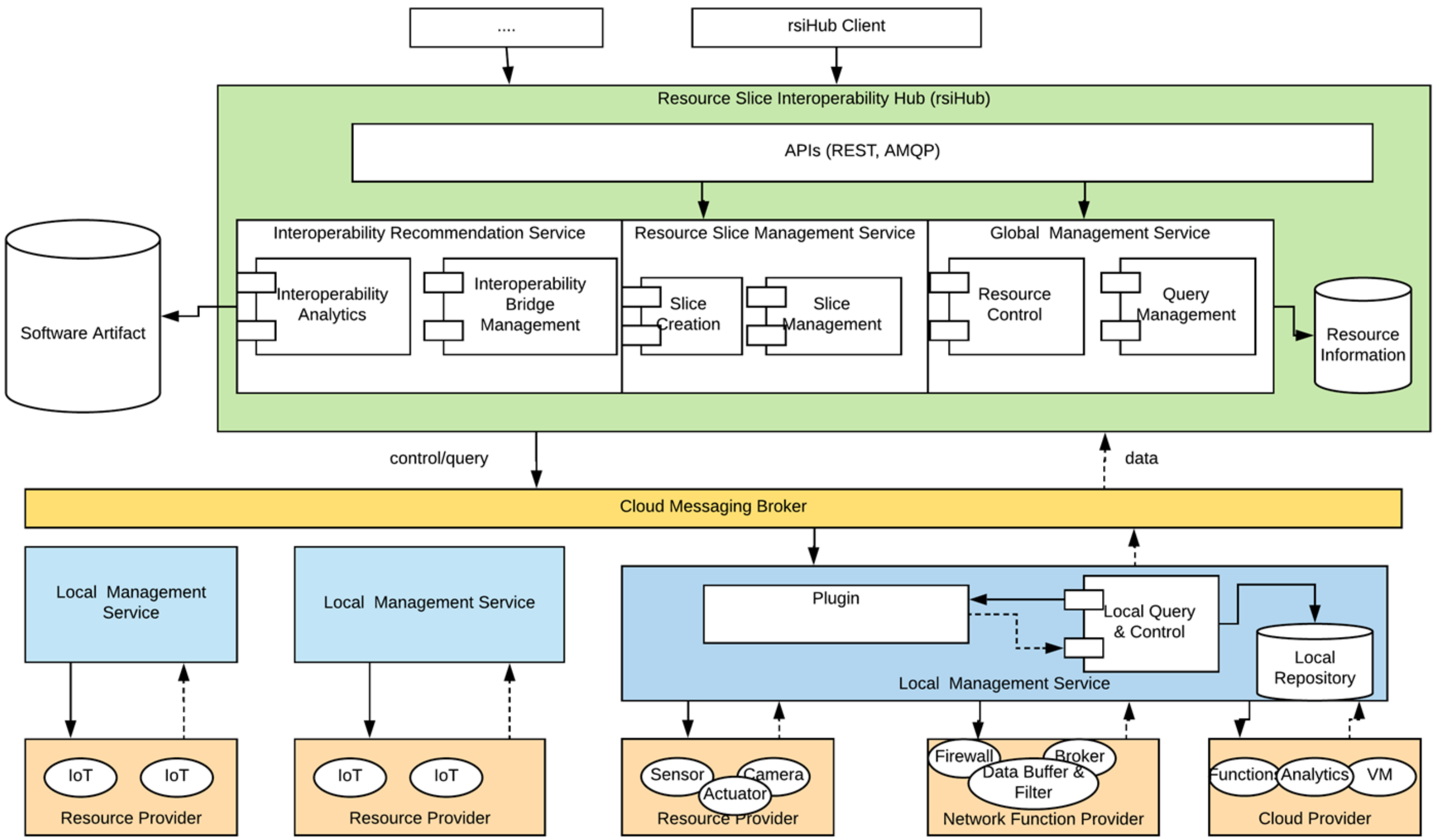
Examples

- `DP(r)= {dpa,dpc}` return all videos and the current video, and `CP(r)={cpp }` put video to a storage
- A service `S` can provide a set of `R={ri}` ($R=\{r_i\}$) allow to use `DP(r)` and `CP(r)`
- Examples of slices
  - `RS(c) ={ri, GoogleStorage, FaaS}`
  - `RS(c) ={ri, Kafka, Trigger, Container}`
- Our interoperable slice structures: many-to-one, one-to-one, one-to-many
- Augment `RS(c)` with `IBE(c)={ibei}` → `RSi(c)` through analytics, recommendation and composition

Integration requirements for dynamic interoperability

- Resource providers:
  - Instances of resource providers provisioning resources
  - Resources and providers can be controlled at runtime
- Repository for artifacts for interoperability
  - Artifacts can be instantiated into the right environments
  - E.g., a middleware service for performing protocol translation, a data pipeline for covering data, or a function for filtering IoT data

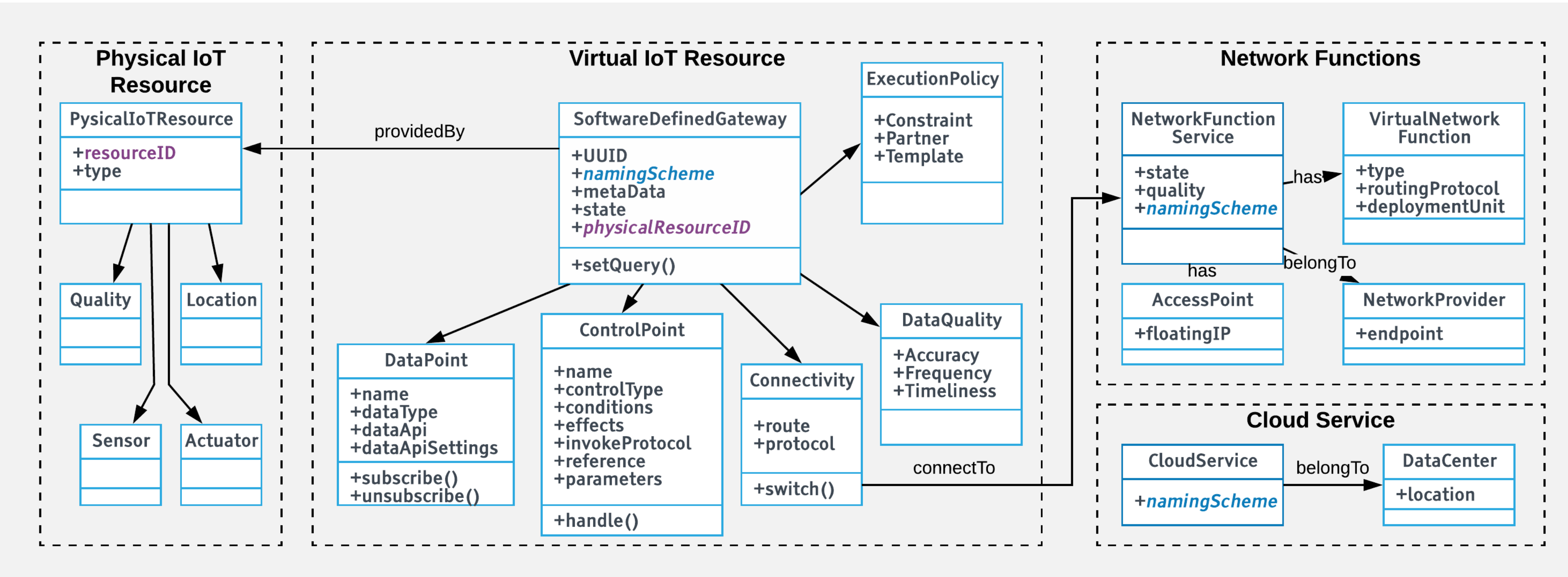
Resource Slice Interoperability Hub architecture



Enriching Client’s Resource slice

- Dynamic interoperability solutions cover
  - Interoperability analytics
  - Recommendation and composition
  - Resource provisioning and configuration
- Metadata focused on runtime aspects
  - Data quality, data delivery, data compliance regulations, data transfer protocols
- Interactions for interoperability
  - Data transformation
  - Protocol translation for data delivery
  - Data contract assurance

Resource abstractions and metadata for interoperability



- Extension of HINC model [FICloud 2016]
- Added new metadata for interoperability context: DataPoint, ControlPoint, ExecutionPolicy, DataQuality, and Connectivity

Metadata

- Typical resource metadata and “interoperabity metadata” but in an extensible model
- We have more interesting novel aspects of *metadata reflecting dynamics for interoperability solutions*, e.g., quality, contract, delivery frequency (related to V* of data) rather than traditional static “protocols” and “formats”

Examples of metadata

- Resource type
  - SENSOR, ACTUATOR; MESSAGEQUEUE, FIREWALL, FILE_STORAGE, VPN, CONTAINER, VIRTUALMACHINE
  - FAAS, INTEROPERABILITY_BRIDGE, DATA_TRANSFORMATION, PROTOCOL_TRANSLATION
- Protocols:
  - REST, MQTT, AMQP, CoAPP
  - IP, LoRaWAN, NB-IOT
- Data access patterns: PubSub (fan-out), ReqResp, Queue
- Data contract
  - Data quality: ACCURACY, TIMELINENESS, etc.
  - Data delivery: FREQUENCY (e.g. sampling rates)
- Execution policy and data regulation: e.g., within EU
- Data format: CSV, JSON, AVRO, RDF

Resource metadata at runtime

Sensor encapsulated in datapoints

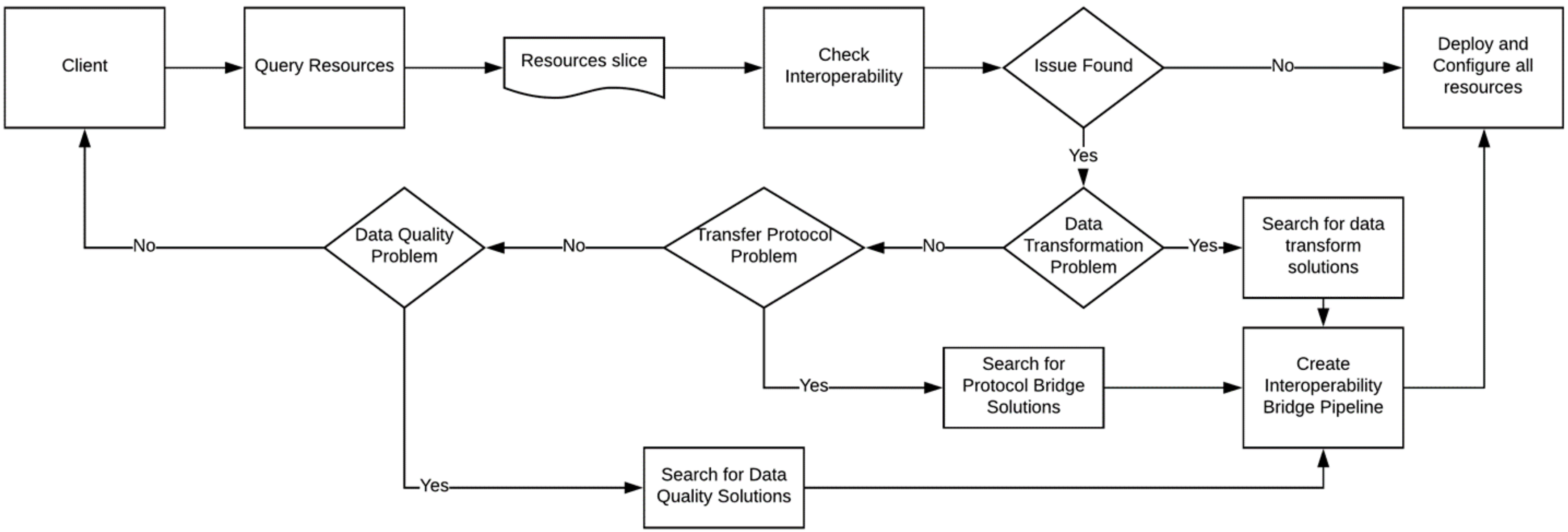
```json
{
    "name": "sensor capacity",
    "pluginName": "bts-sensor-provider",
    "resourceType": "IOT_RESOURCE",
    "location": null,
    "metadata": {
        "category":"sensor",
        "subcategory":"humidity",
        "output":{
            "pattern":"pubsub",
            "protocol":"mqtt",
            "dataformat":"csv",
            "uri": "tcp://35.205.X.X:1883",
            "topic": "topic_softwareartefact1523451460864",
            "createdAt": 1523446948
        }
    },
    "controlPoints": [],
    "dataPoints": [
        {
            "name": "capacitance",
            "unit": "farad"
        }
    ],
    "uuid": "sensor1523699860359"
},
```

*Some sensitive information removed

Software artifact for interoperability

```json
{
    "name": "js-artefact-runner",
    "pluginName": "js-artefact-plugin",
    "resourceType": "SOFTWARE_UNIT",
    "location": null,
    "metadata": {
        "software_artefact_ref": "softwareartefact1523451460864",
        "category": "datatransformer",
        "input": {
            "pattern":"pubsub",
            "protocol":"mqtt",
            "dataformat":"csv",
            "brokers": [
                {
                    "host": "35.205.X.X",
                    "port": 1883,
                    "clientId": "testclient1",
                    "username": "xxx",
                    "password": "xxx",
                    "topics": [
                        "topic_softwareartefact1523451460864"
                    ]
                }
            ]
        },
        "output": {
            "pattern":"pubsub",
            "protocol":"mqtt",
            "dataformat":"json",
            "brokers": [
                {
                    "host": "35.205.X.X",
                    "port": 1883,
                    "clientId": "testclient1",
                    "username": "xxx",
                    "password": "xxx",
                    "topics": [
                        "test"
                    ]
                }
            ]
```

Key steps in interaction in solving interoperability

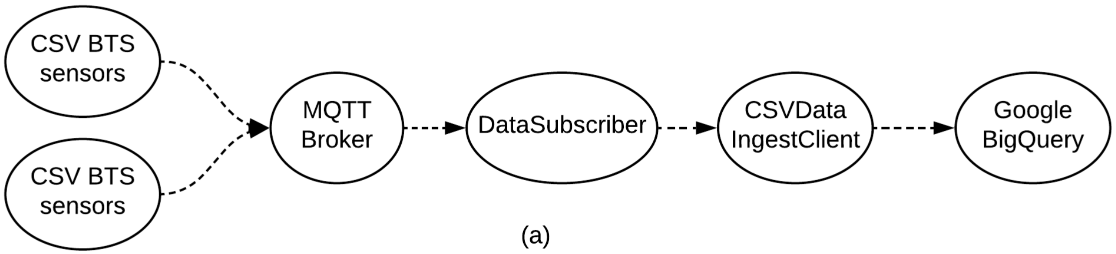


Key steps are reflected in *interoperability analytics, recommendation and provisioning*

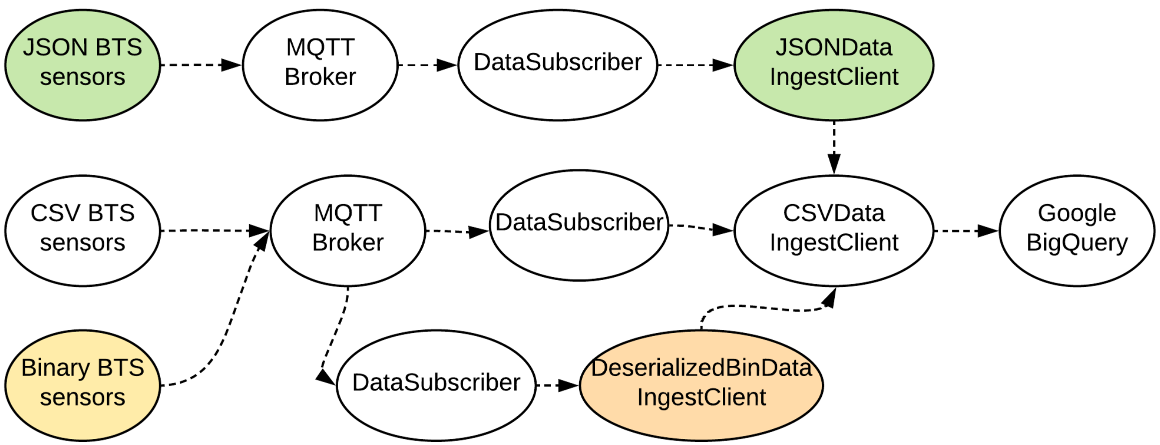
Provisioning and Configuration

- Microservices with containers and virtual machines suitable for IoT, network functions and clouds
- Extensibility
  - not all services are developed by us
  - Leverage static interoperability solutions at runtime
- Different layers and models, e.g.,
  - Long running services based on specific protocols, like messaging, storage, queues, and REST web services
  - Function-as-a-service, based on event triggers
  - Batch workflow/pipeline styles

Examples with Base Transceiver Station Slice



(a)



(b)

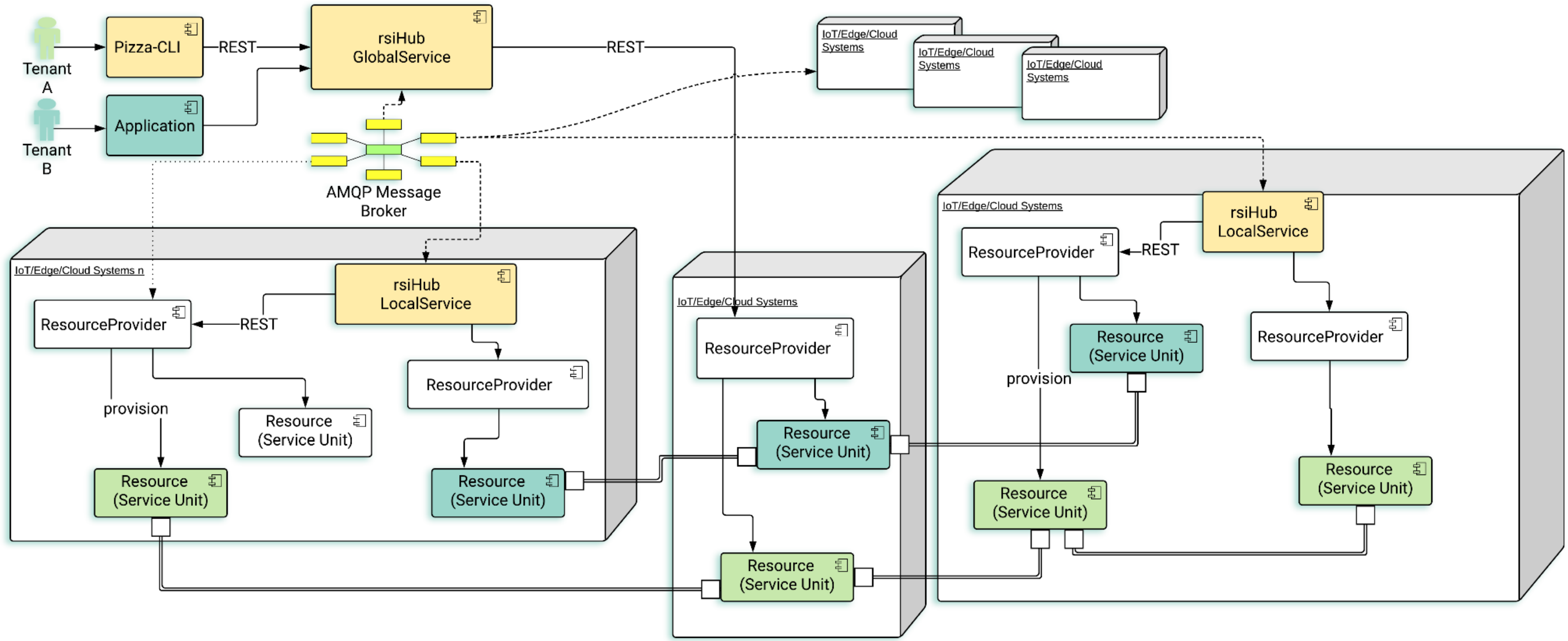
Other examples of runtime operations:

- Network functions like firewall can be added into the pipeline
- Traffics at NFV can be reengineered.

Current prototype

- rsiHub prototype
  - https://github.com/SINCConcept/HINC
- Examples of resource providers, including for interoperability solutions
  - https://github.com/rdsea/IoTCloudSamples
- Video demo
  - A slice with BTS with data transformation
    - https://youtu.be/_SCrK8Q3xBs
    - By Lingfan Gao and Michael Hammerer

Tooling



Check:

- https://github.com/SINCConcept/HINC
- Hong-Linh Truong, Lingfan Gao, Michael Hammerer, Service Architectures and Dynamic Solutions for Interoperability of IoT, Network Functions and Cloud Resources, 12th European Conference on Software Architecture, September 24-28, 2018, Madrid, Spain

Summary

- Ensembles of IoT, network functions and clouds
  - Important for various types of applications
  - SINC: a conceptual framework
- Key issues
  - We need to consider more data and network aspects.
  - Adaptation and optimization under uncertainties
  - Modeling and testing are very challenging
  - DevOps for ensembles.

→ We need new set of tools and techniques for managing and testing ensembles.

Interesting topics for research

- APIs for programming resource queries and controls (http://sincconcept.github.io/HINC/)
- Configuration tools (http://tuwiendsg.github.io/SALSA/)
- Uncertainty testing and analytics
- Monitoring and analytics
- Ensemble requirement modeling, composition algorithms and optimization
- Interoperability issues (http://www.inter-iot-project.eu/)
- Policy execution

Check http://rdsea.github.io and https://github.com/tuwiendsg/COMOT4U for new update

Thanks for your attention!

Hong-Linh Truong

Faculty of Informatics
TU Wien

rdsea.github.io