Dynamic IoT Data, Protocol, and Middleware Interoperability with Resource Slice Concepts and Tools

version 0.1.0 – Last update 15.10.2018

Hong-Linh Truong

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

Acknowledgements:

Many results are from the joint work with my students and colleagues. Especially thanks to Lingfan Gao, Michael Hammerer, and Duc-Hung Le

Partially supported by

- H2020 Inter-IoT (http://www.inter-iot-project.eu/)
- Google Cloud Platform Research Grant for resources for development and experiments

Note: this research tutorial includes materials under submission or in working papers.

Outline

- IoT Cloud development and IoT platform interoperability
- DevOps and Resource Slice for Dynamic IoT interoperability solutions
- Examples and use cases with rsiHub and IoTCloudSamples
- Summary

Tutorial material

- Slides and relevant documents will be uploaded to the tutorial site

https://github.com/rdsea/iot2018tutorial

- For some tests today:
  - Some services have been deployed in Google Cloud Platform
  - But we wont be able to run many examples
    - Follow the links and material examples in the slides and the tutorial site

Part 1: IoT Cloud development and IoT interoperability

IoT Cloud software systems development

Existing IoT, edge, network functions and cloud infrastructures

- Cloud resources
  - Datahub, message brokers, databases, analytics, etc.
  - Such resources can be requested on-demand
- Edge/fog and network functions resources
  - Firewall, lightweight brokers, storage, etc.
  - Software-based network functions, deployed on-demand using cloud technologies
- IoT resources
  - Sensors, Actuator, IoT Gateways, IoT/Edge service platforms, etc.

Resource notion

- Avoid thinking resources as CPU and memory (aka virtual machines)!

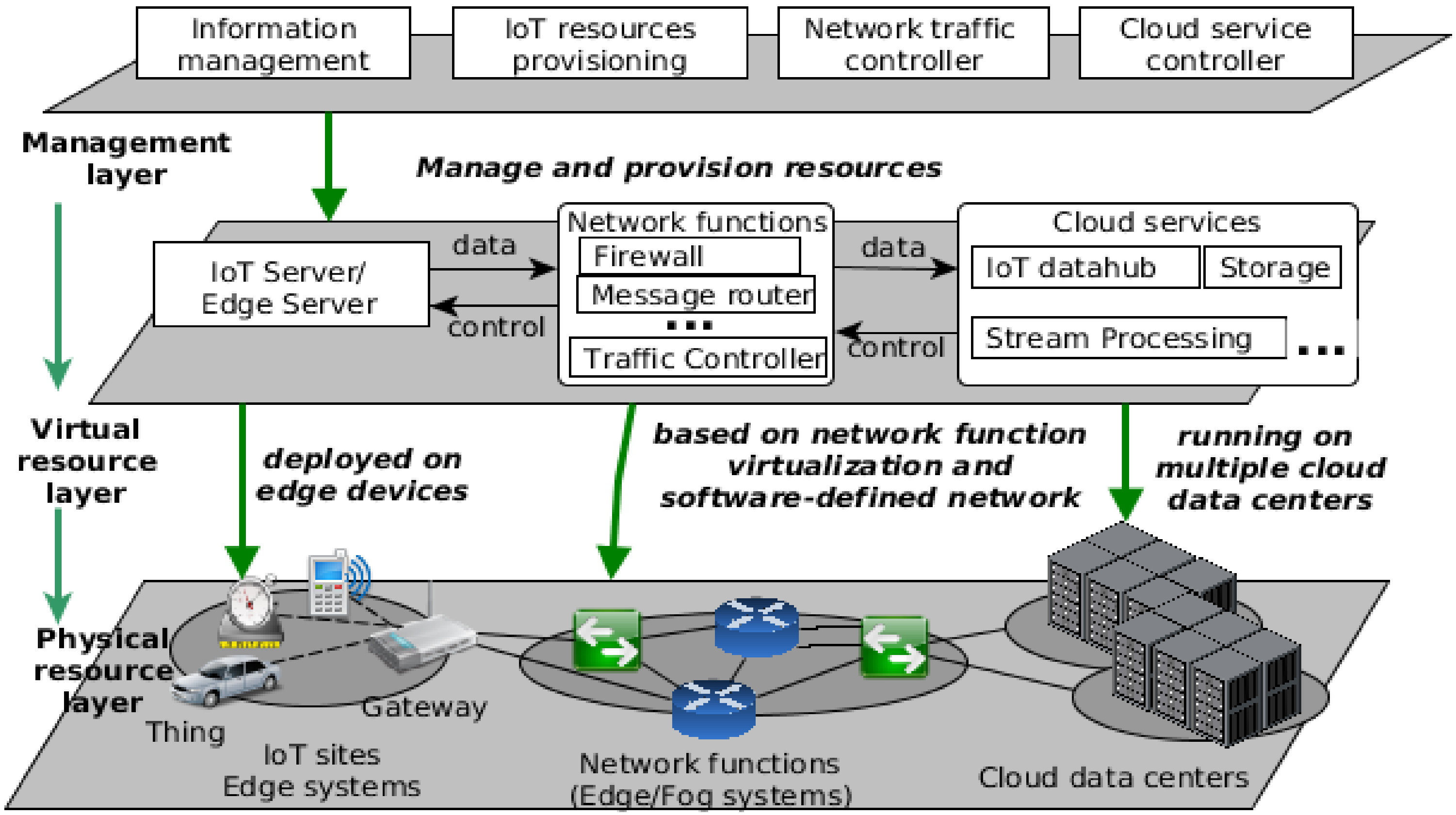
Resources in IoT Cloud software systems:

data streams, analytic component, message broker, storage, etc.

- Resources are used as services: pay-per-use and one-demand with well-defined interfaces

  → provisioned when needed!

A view on resources for IoT Cloud software development



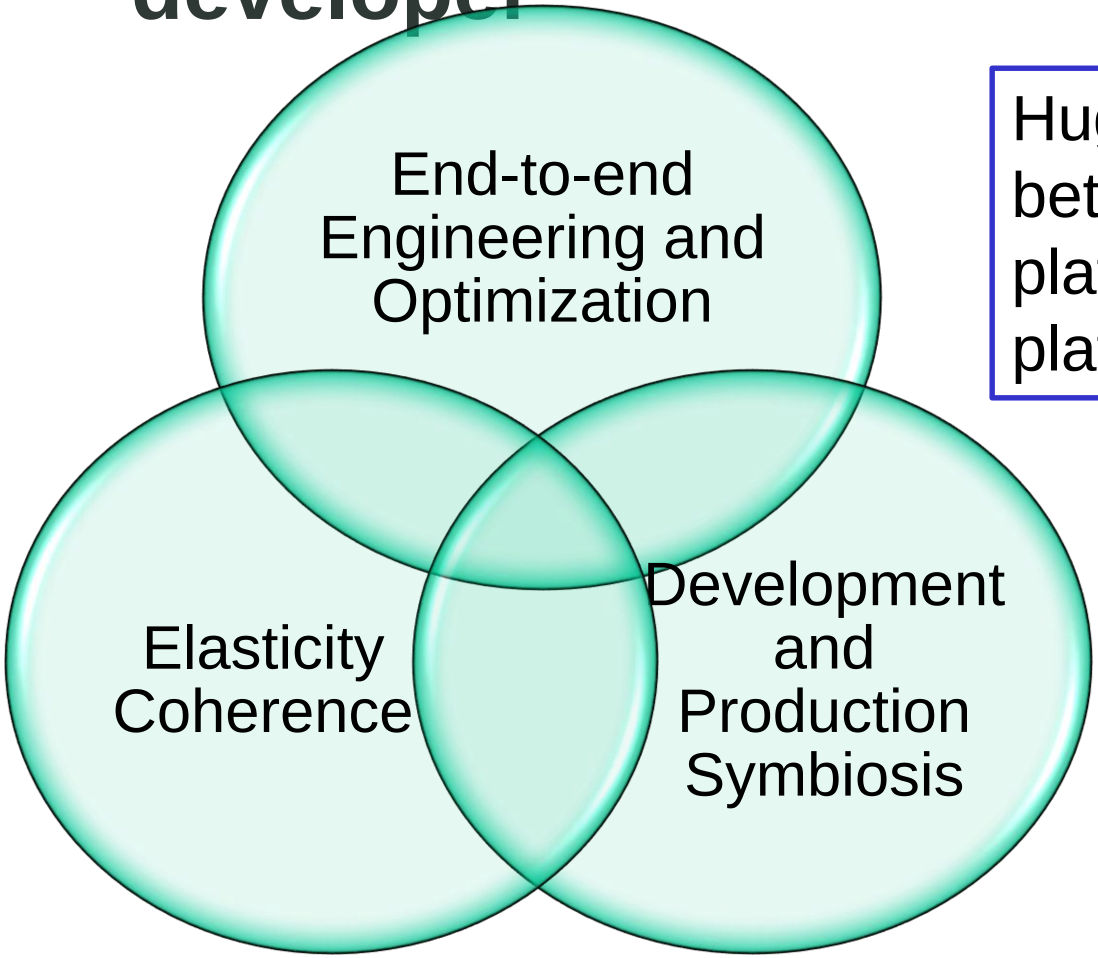
<u>Adapted from: </u>Duc-Hung Le, Nanjangud C. Narendra, Hong Linh Truong:
HINC - Harmonizing Diverse Resource Information across IoT, Network Functions, and Clouds. FiCloud 2016: 317-324

IoT Cloud Systems development patterns

- Type 1 **mainly focus** on IoT/edge networks:
  - sensors, Things connectivity, sensor data models, IoT gateways, IoT-to- cloud connectivity
- Type 2 **mainly focus** on (public/private) services in data centers:
  - e.g., IoT data hubs, time serties data management, NoSQL databases, big data ingest systems, and data integration
- Type 3 **equally focus** on across IoT and cloud sides and have the need to control at both sides:
  - Highly interactions between the two sides, including the network functions in the middle

Check: Hong Linh Truong, Georgiana Copil, Schahram Dustdar, Duc-Hung Le, Daniel Moldovan, Stefan Nastic: **On Engineering Analytics for Elastic IoT Cloud Platforms**. ICSOC 2016: 267-281

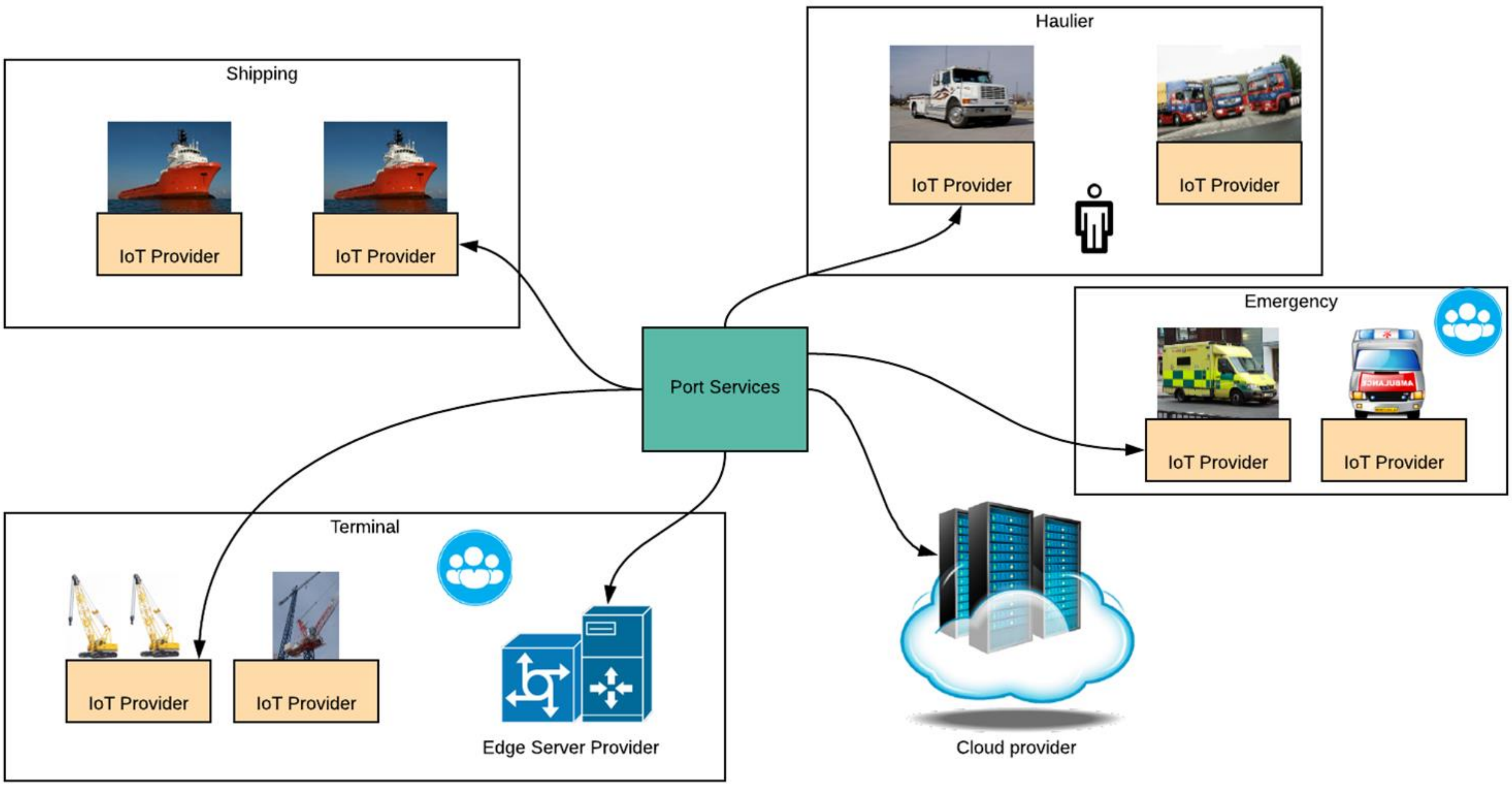
Engineering perspectives for the developer



Huge differences between single platform & multiple platforms

Check: Hong Linh Truong, Schahram Dustdar: **Principles for Engineering IoT Cloud Systems**. IEEE Cloud Computing 2(2): 68-76 (2015)

Example of IoT Cloud systems



Check: the H2020 EU INTER-IoT project -http://www.inter-iot-project.eu/

Diversity and Complexity

- Programming APIs
  - REST API, client libraries (Javascript, Python, Java, etc.), WebSocket
  - They might/might not support standard application protocols
- Protocols: MQTT, AMQP, CoAP, Modbus, etc.
- Data format:
  - JSON, AVRO, BSON, CSV
- Data syntax & semantics
  - Known and specific data models

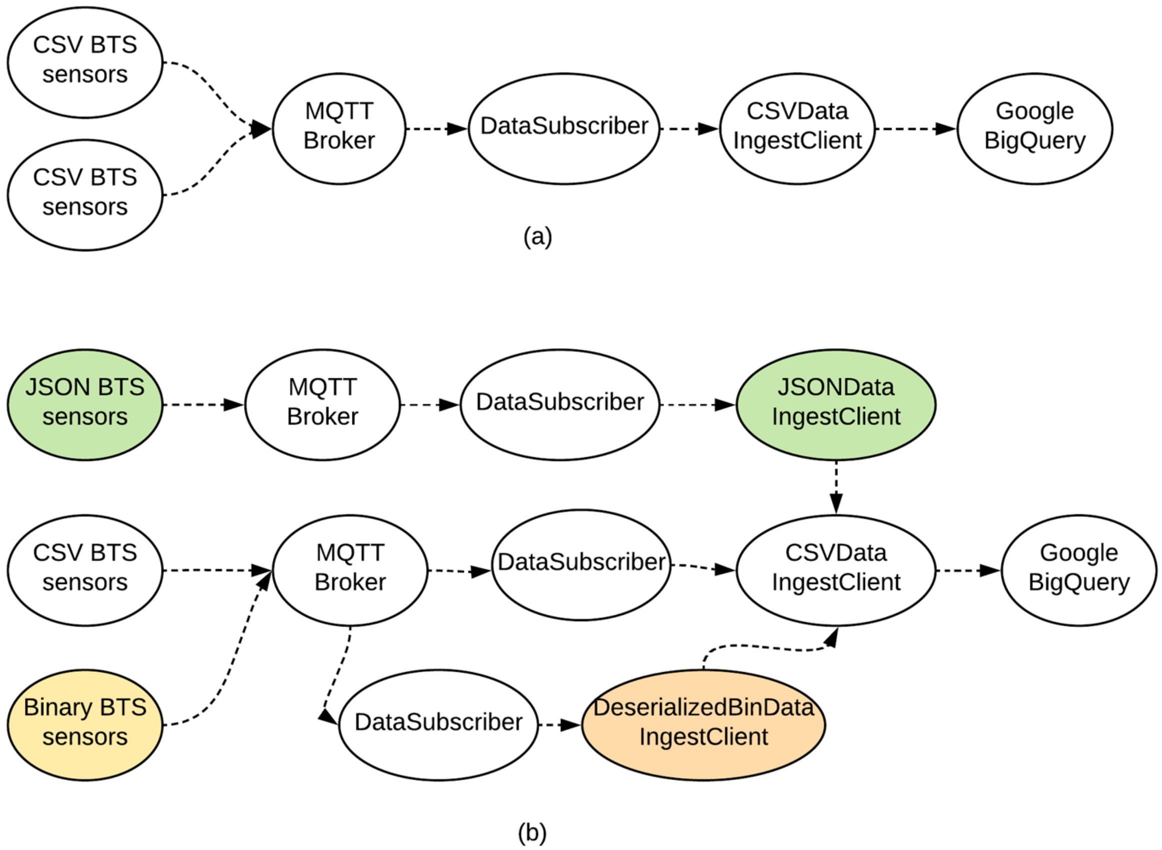
Examples of data models and APIs

Platform	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

Dealing with diversity and complexity

- **API Integration**
  - Use REST APIs/common message protocols for obtaining resources and data
- **Data Transformation**
  - *Processes and services* for transforming data among different models
- **Building multi-protocol integrator/datahubs at application layers**
  - AMQP, MQTT, CoAP, etc.
- **Building protocol bridge**
  - MQTT-Kafka, MQTT-AMQP, etc.

Examples of typical tasks



Source : Hong Linh Truong: **Towards a Resource Slice Interoperability Hub for IoT**. IC2E 2018: 310-316

Example of typical tools for building IoT Cloud software

- Sensors can be written in Java/Javascript/Python
  - Simulated/emulated sensors just read data from sample files
  - Sensors to gateways communication: TCP/IP, MQTT, LoRa, BLE, etc.
- Gateways:
  - Raspberry Pi, lightweight virtual machines, vendor-specific devices
  - Gateway-to-cloud: MQTT/AMQP, Web socket, CoAP,
  - Network layer connectivity: IP, 4G, LoRaWAN, Sigfox, etc.
- Cloud platforms:
  - Using own clouds or Google, Microsoft, and Amazon

Identifying IoT interoperability issues

Interoperability

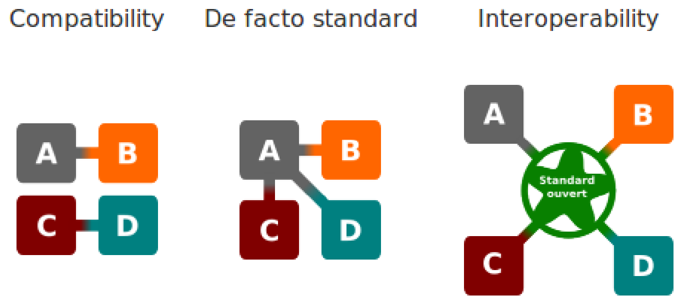
From http://interoperability-definition.info/en/

DEFINITION:

Interoperability is a characteristic of a product or system, whose interfaces are completely understood, to work with other products or systems, present or future, in either implementation or access, without any restrictions.

Translation: GDT Interop, John McCreesh, Ed Daniel

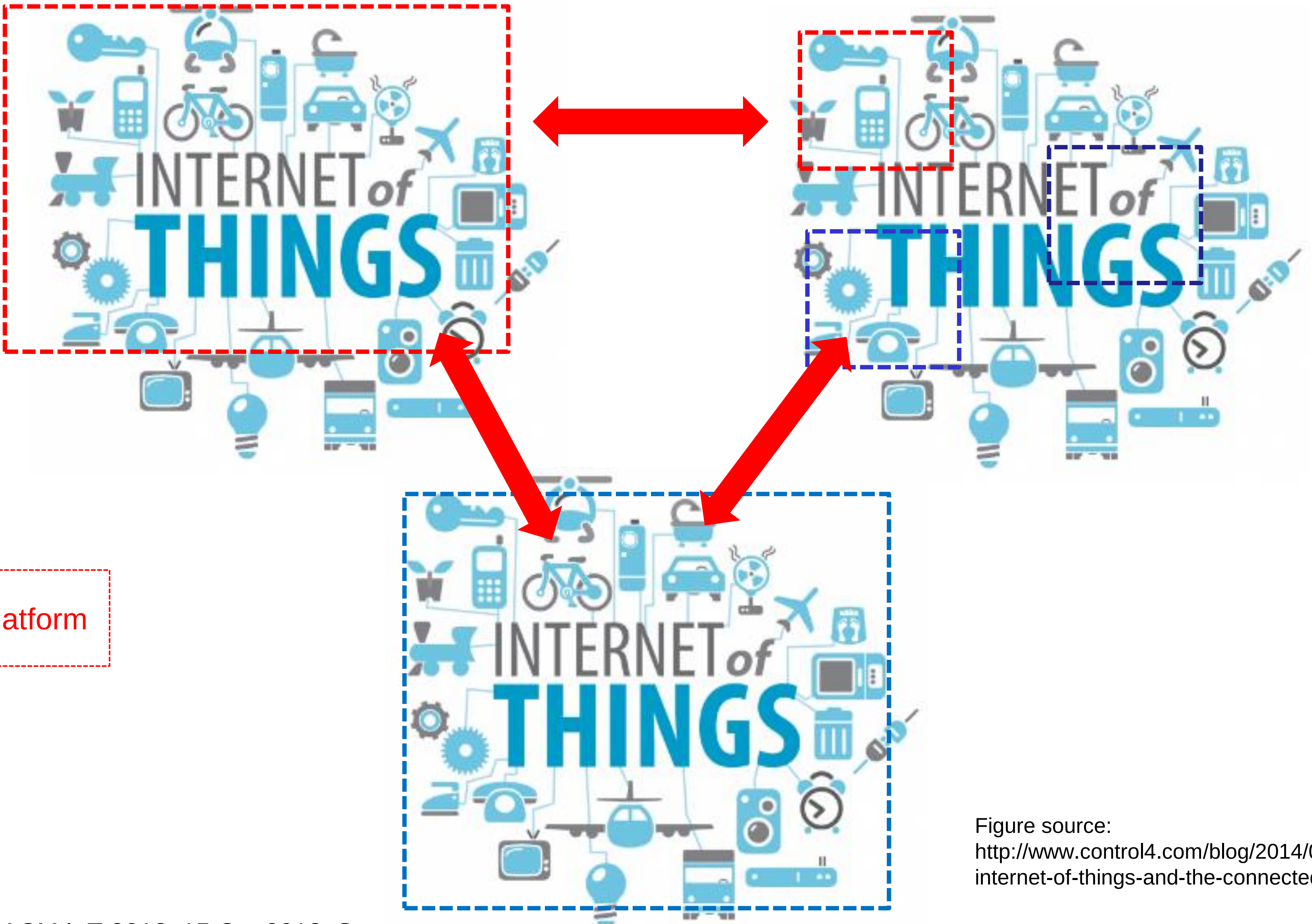
DEGREE OF OPERABILITY



However, we do not strictly follow the definition!

Interoperability issues

- **On IoT/edge side**
  - devices integration, sensor connectivity, sensor data models, discovery, etc.
  - Interoperability among Things, within a platform
- **On (public/private) services in data centers for IoT**
  - Protocol and data integration, data transformation, etc.
  - Interoperability in accessing multiple IoT platforms
- **On both IoT and cloud sides**
  - Platform to platform, IoT middleware, IoT-Cloud
  - Interoperability across multiple IoT platforms



Platform

Figure source:
http://www.control4.com/blog/2014/03/the-internet-of-things-and-the-connected-home

Some observations (1)

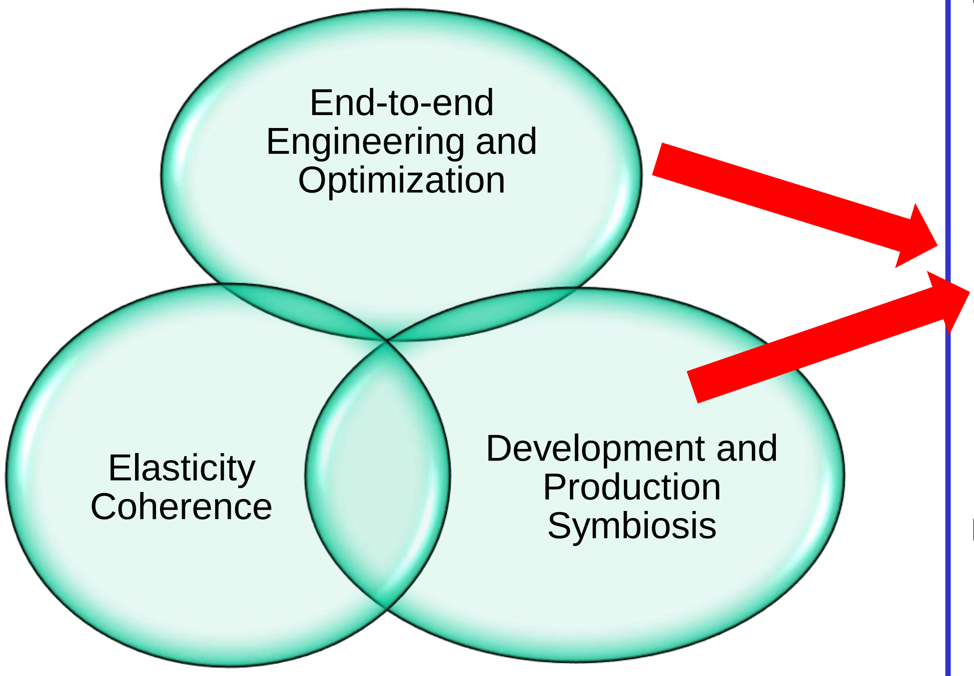
- Obviously there are many concepts and software for data models, protocols, middleware, etc.
- Software components are increasingly virtualized/dockerized and developed as microservices
  - thus can be dynamically provisioned on-demand

→ *how will this impact on interoperability and how do we leverage them for solving interoperability*

- IoT components and data continue to be introduced in a fast way

→ *should we consider solving interoperability as a typical programming tasks (besides working on standard models)*

Some observations (2): engineering perspectives for the developer



- Huge number of interoperability issues to be solved during the development!
- The scale of the deployment is an important factor as well

Some observations (3): governance and compliance

- Even the same technology stack but deployed and controlled by different organizations
  - Different policies and regulations, e.g. the same technology stack for different application domains
  - The same technology but many parts are customized (e.g., data format of sensors)

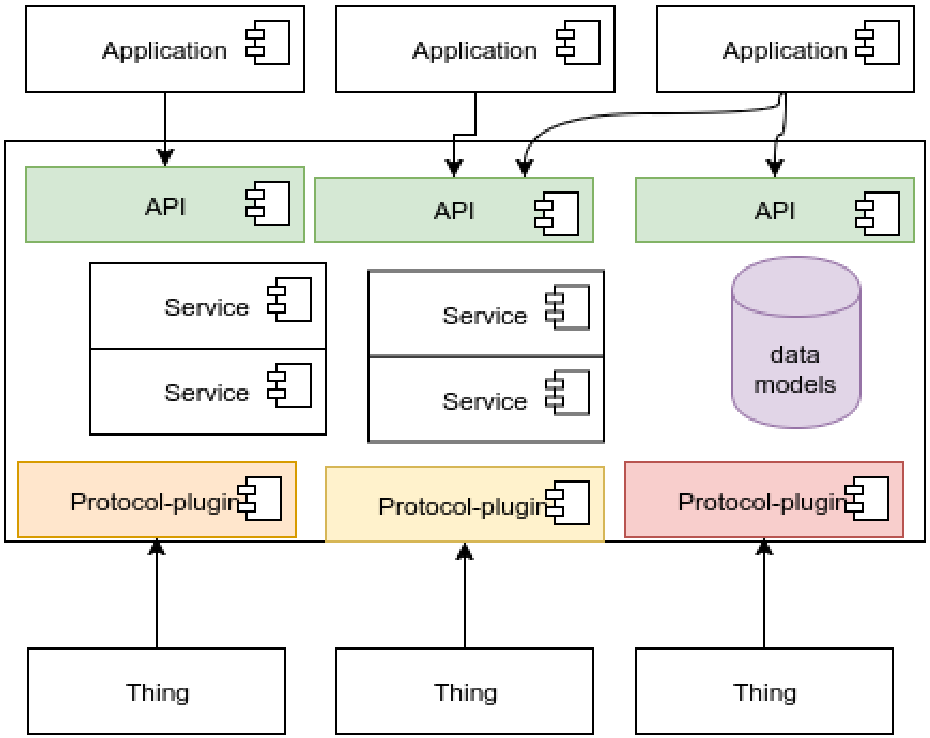
→ Don’t expect that everything can be done in advance and we want to minimize changes in software development!

Interoperability among Things, within an IoT platform (1)

- Key requirements
  - How to make Things work together within an IoT system
    - Often physical infrastructures belong to the same organization
  - How to make Things exchange data for different applications

- Interoperability issues:
  - connectivity, application protocols, service discovery, data exchange, etc.

Interoperability among Things, within an IoT platform (2)



- Key focuses
  - device discovery interoperability,
  - network interoperability
  - data interoperability

For some survey and classifications:
Noura, M., Atiquzzaman, M. & Gaedke, M. , **Interoperability in Internet of Things: Taxonomies and Open Challenges**, Mobile Network and Application (2018). https://doi.org/10.1007/s11036-018-1089-9

Example: OpenHab

- https://www.openhab.org
- Binding with various underlying hardware/things
  - Support various protocol interoperability
- Simple transformation
  - Can be used for data interoperability
- API for accessing data
- Connectors for pushing data to the cloud
- OSGI based enables dynamics but still within a platform

Example: oneM2M

- http://www.onem2m.org
  - Distributed architecture
  - Data interoperability
  - Integration with different application protocols (HTTP/CoAP)
  - Able to support cross application domains

- Specifications:
  - http://onem2m.org/technical/partner-transpositions

Example: Open Connectivity Foundation

- https://openconnectivity.org
- Interoperable REST interfaces & data models:
  - https://openconnectivity.org/developer/oneiota-data-model-tool
  - http://oneiota.org/
- Bridge models: define a set of bindings

  https://openconnectivity.org/specs/OCF_Bridging_Specification.pdf
- Reference implementations
  - IoTivity: https://iotivity.org/downloads
  - AllJoyn: https://github.com/alljoyn/alljoyn.github.com/wiki

Interoperability among IoT platforms

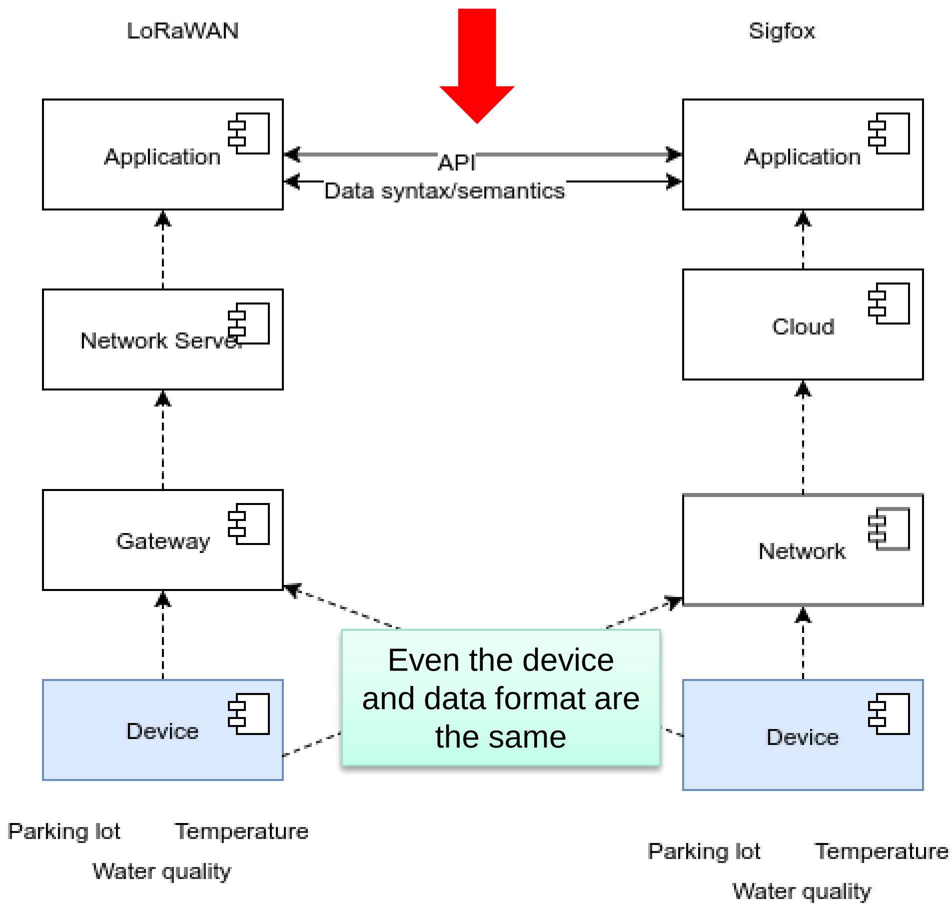
- Interoperability among IoT Platforms
  - How to exchange data of and controls for resources between IoT platforms
    - Dealing with high level of data/controls abstraction

- Key issues: data interoperability, APIs, application connectivity protocols in edge/cloud environments
  - Protocols exposed by a platform are usually common ones used in current edge/cloud computing
  - Access control is another related issue

- Complexity of data is very high because typically we use multiple IoT platforms for integrating different application domains

Example: only able to deal at the application layer

Interoperability w.r.t API and data



The TNT:

https://www.thethingsnetwork.org/docs/network/architecture.html

Sigfox:

https://ask.sigfox.com/questions/606/is-there-a-general-network-architecture-descriptio.html

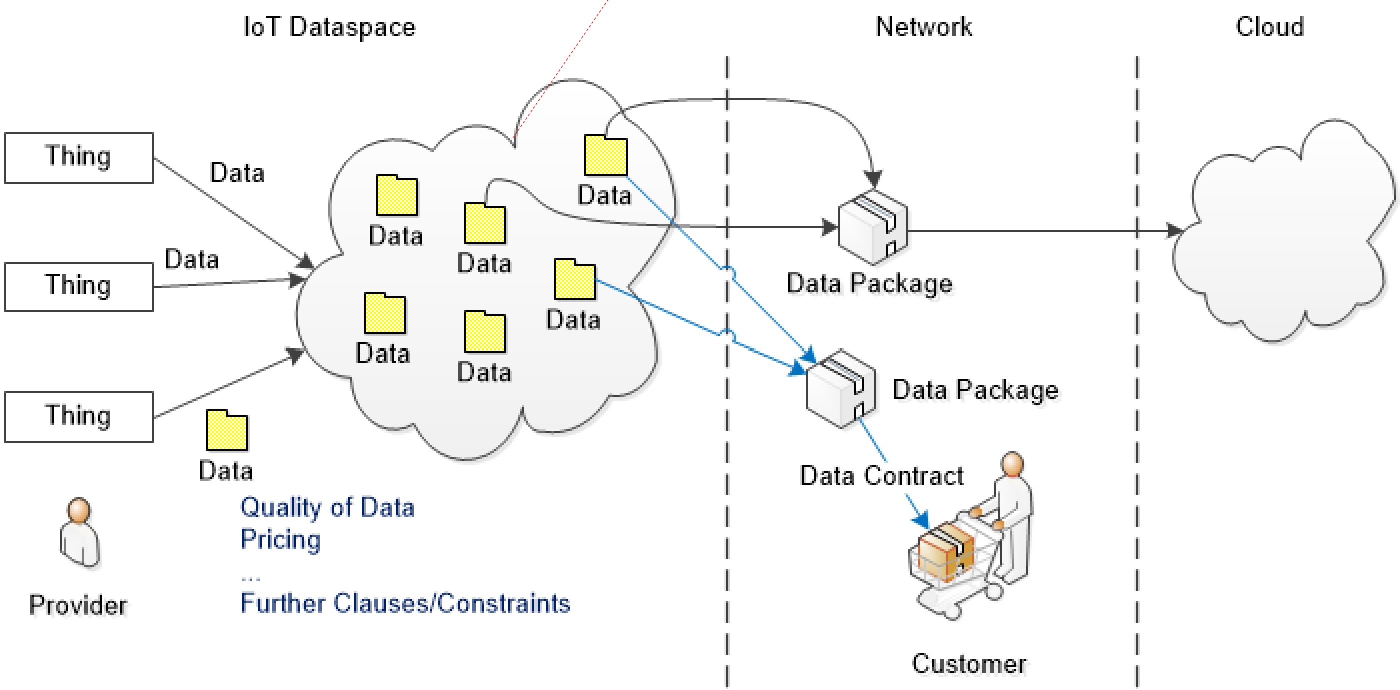
Parking sensor:

http://www.libelium.com/products/smart-parking/

Examples of data and semantics issues

Centralized marketplaces/datahub models: IoT data as a service offered by different types of providers

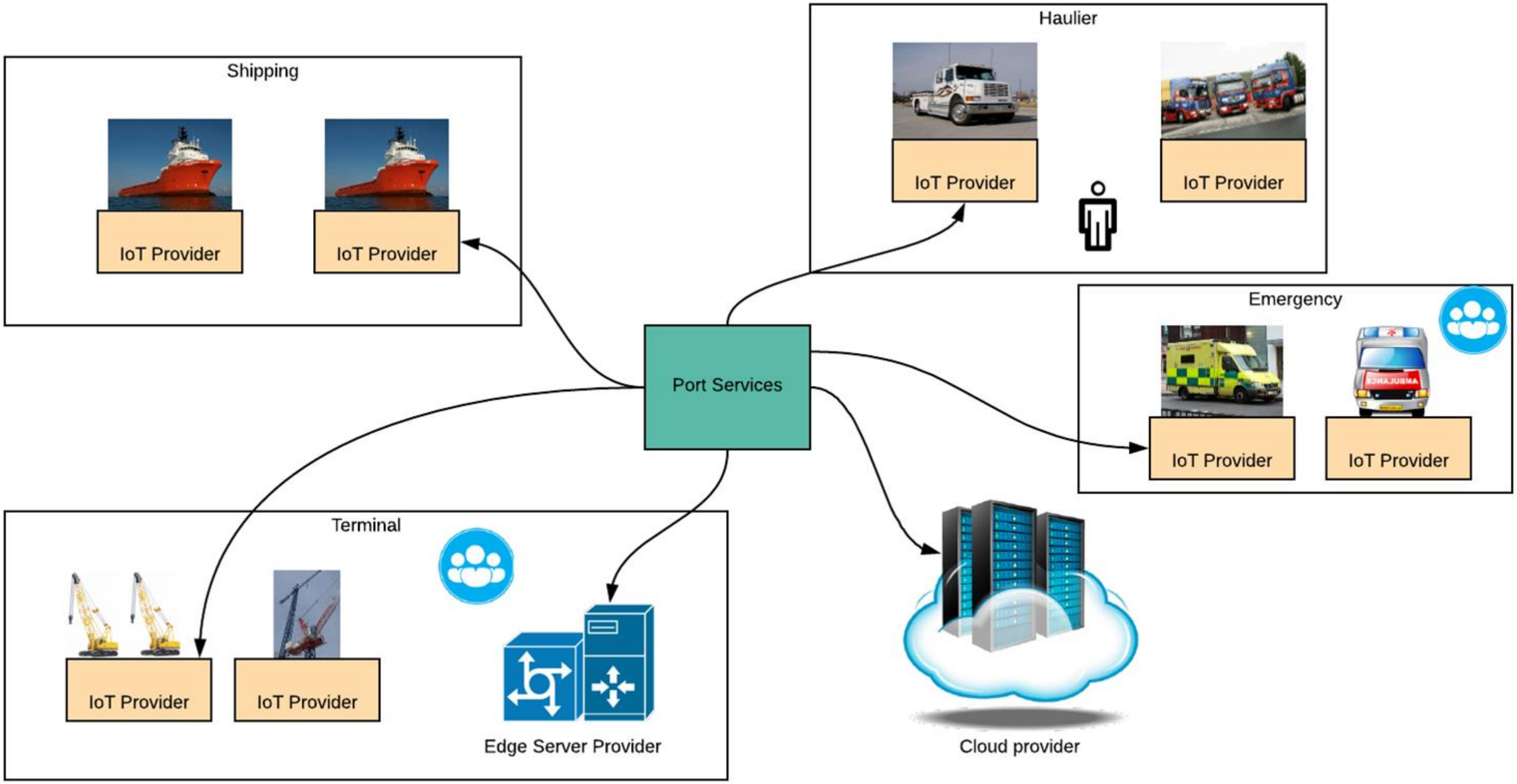
Interoperability w.r.t data semantics, syntax and regulation



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)

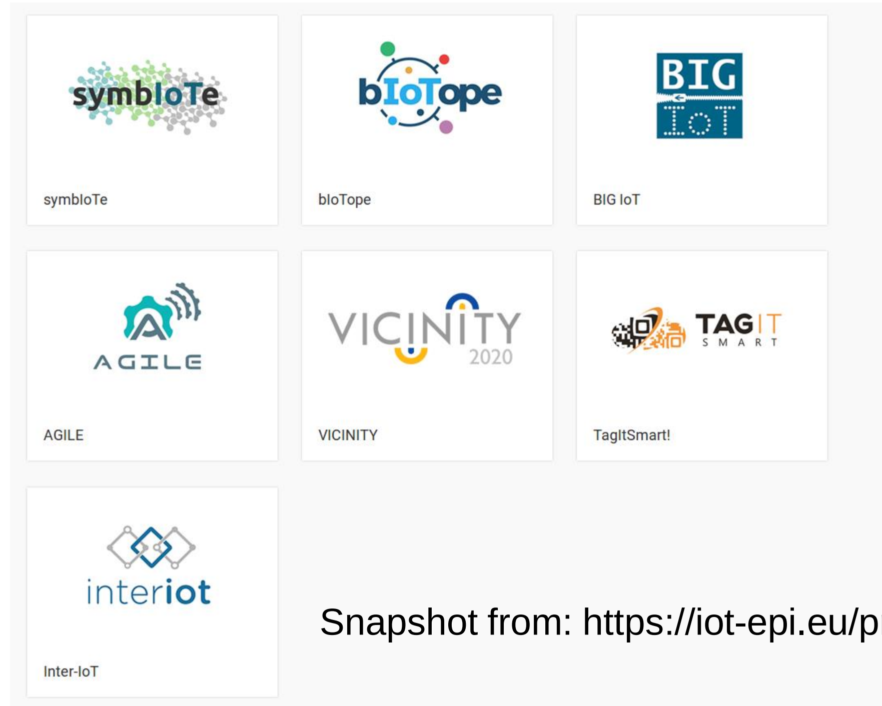
Complex case: example of SeaPort

Data format, data semantics, APIs, middleware services, and regulations



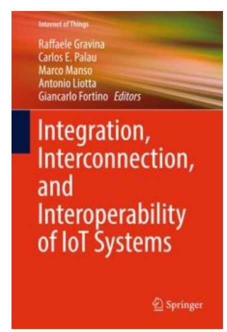
Some EU projects

- Address various aspects of IoT cross-platform interoperability



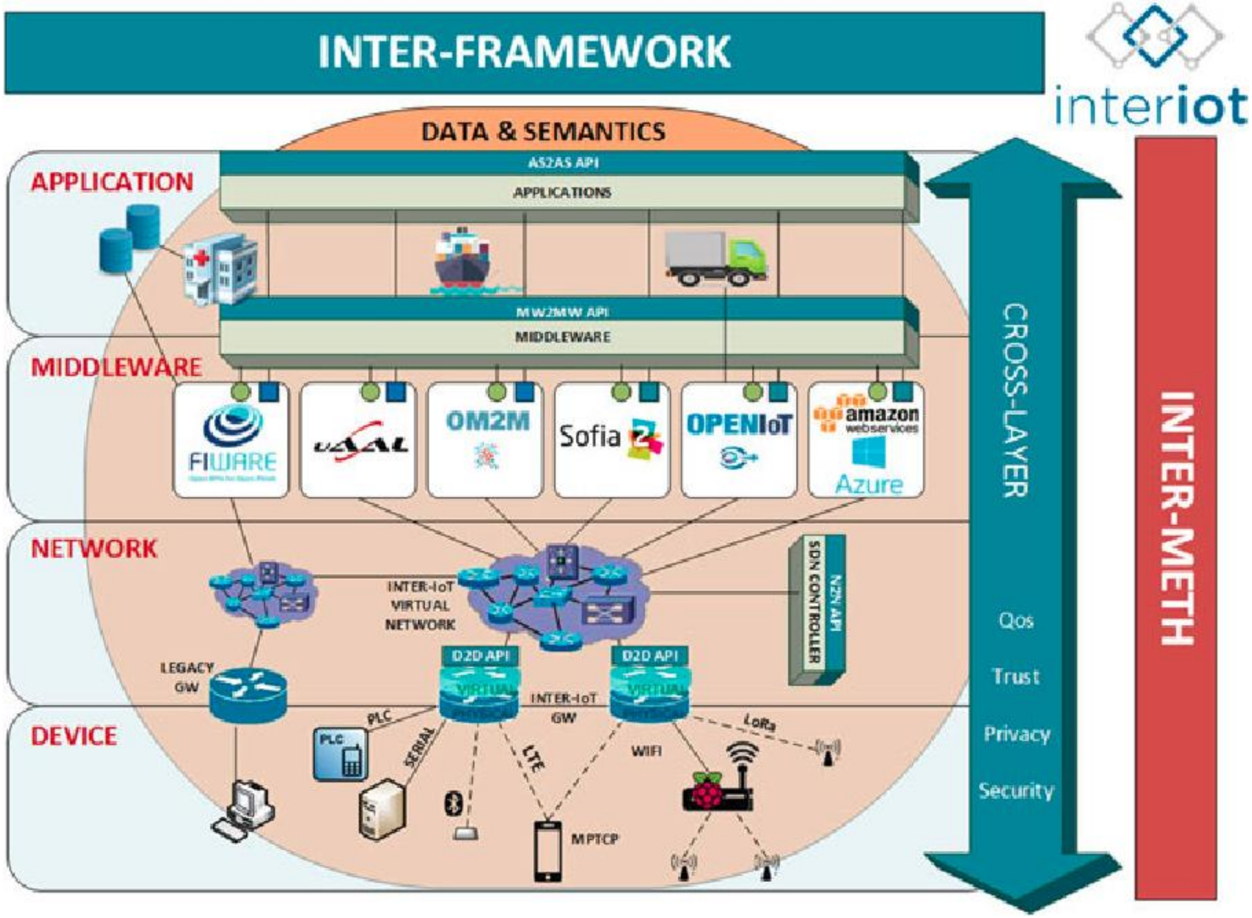
Snapshot from: https://iot-epi.eu/projects/

Check: https://www.computer.org/web/computingnow/archive/interoperability-in-the-internet-of-things-december-2016-introduction



Example: H2020 INTER-IoT

Web site: http://www.inter-iot-project.eu/



Fortino G. et al. (2018) **Towards Multi-layer Interoperability of Heterogeneous IoT Platforms**: The INTER-IoT Approach. In: Gravina R., Palau C., Manso M., Liotta A., Fortino G. (eds) Integration, Interconnection, and Interoperability of IoT Systems. Internet of Things (Technology, Communications and Computing). Springer

Example: H2020 BigIoT

- http://big-iot.eu/
- Similar to centralized data marketplace/datahub model
- Integrate data from platforms through API integration

- Examples of development tasks: https://big-iot.github.io/

Big-IoT figure and paper: A. Bröring et al., **"Enabling IoT Ecosystems through Platform Interoperability**," in IEEE Software, vol. 34, no. 1, pp. 54-61, Jan.-Feb. 2017

Is that interoperability among IoT platforms like typical cross-platform interoperability?

→ Yes! especially when we look at high-level application protocols, API integration, common data models, data semantics, service composition

Move to a more dynamic, runtime approach

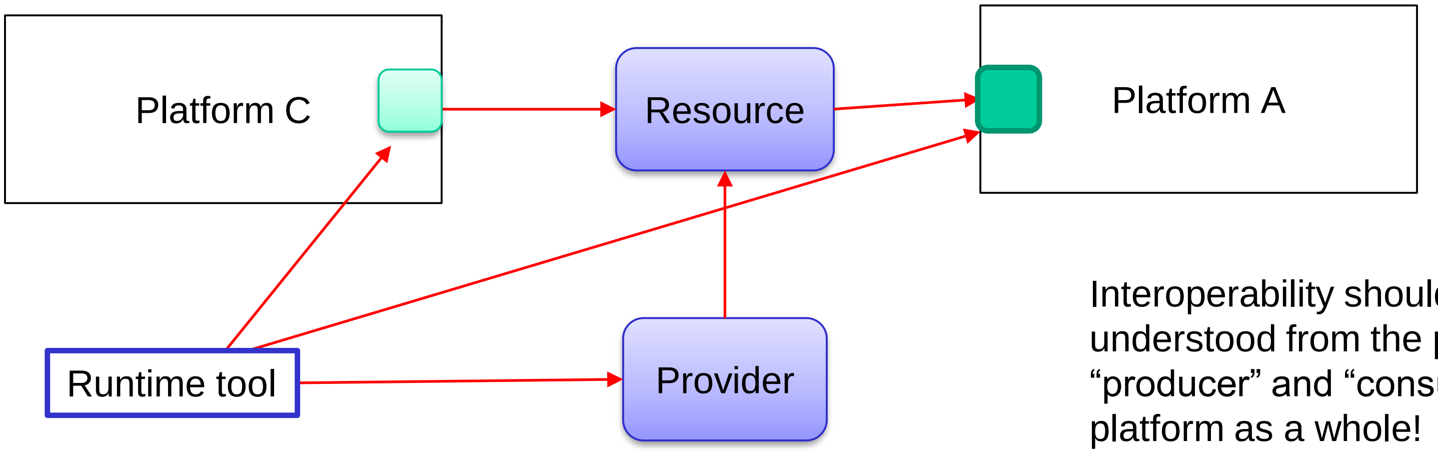
- Static interoperability approach:
  - Statically configured, fixed, not extensible, long-development cycle, hard to reconfigured, lack of microservices mindset
- Dynamic interoperability approach
  - Focus on dynamicity and runtime for dealing with interoperability
    - configuration of existing static software
    - adaptation, change in a short time.

→ Fit well with DevOps “Interoperability-Solution-as-Code”: Not waiting for standard models but fast code development for solving interoperability issues

What would be our focus in IoT interoperability?

- IoT data and controls characteristics at high-level abstractions
  - as low-level abstractions have been addressed heavily
- Think about developer perspectives
  - solving interoperability issues is not based on “standards”
- Think about current infrastructures
  - we are living in edge and cloud dynamics and virtualization of everything

Focus 1: virtualization and edge/cloud orchestration



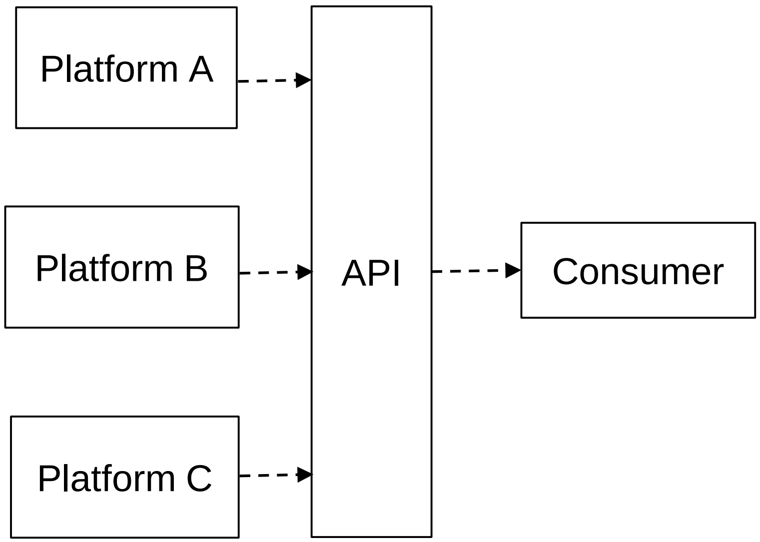
Interoperability should be understood from the perspective of "producer" and "consumer" not the platform as a whole!

- Discover the required features of IoT resources that one needs
  - Data and controls (not sensors or actuators)
- *Acquire* and *control* relevant resources for obtaining the required features
  - Beyond typical "service composition" due to cross-system and cross-layer virtualized resources

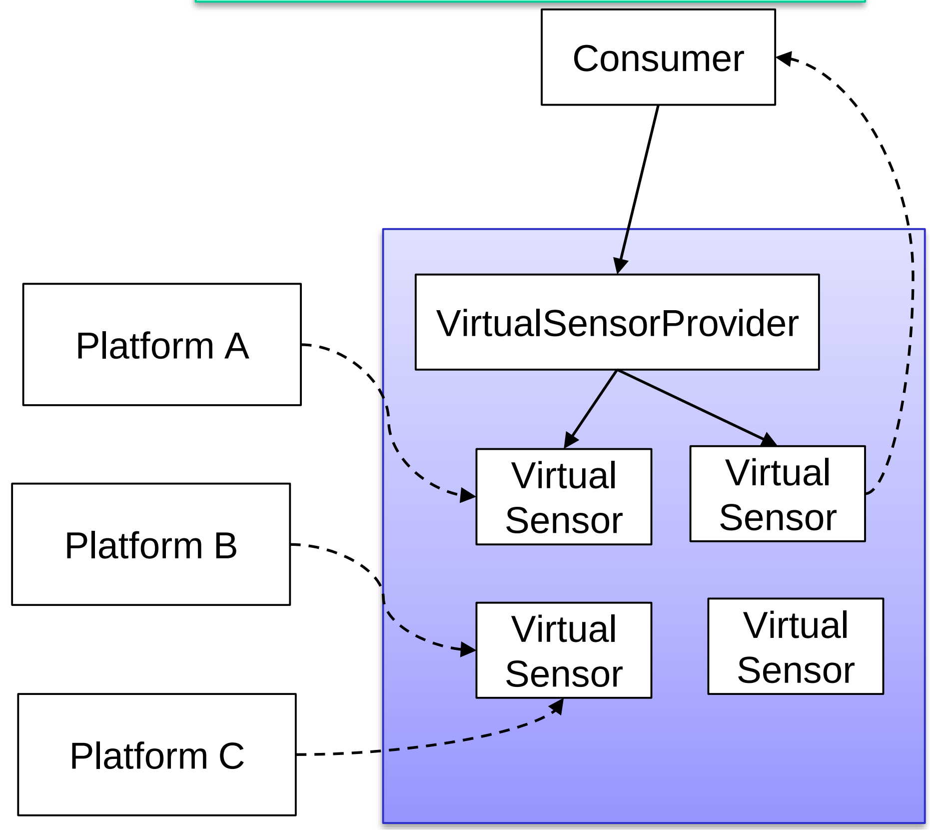
Focus 2: data-as-a-service model for IoT interoperability

From Centralized Data Marketplace/datahub with VirtualIoTData-as-a-Service

Marketplace/data hub Principles

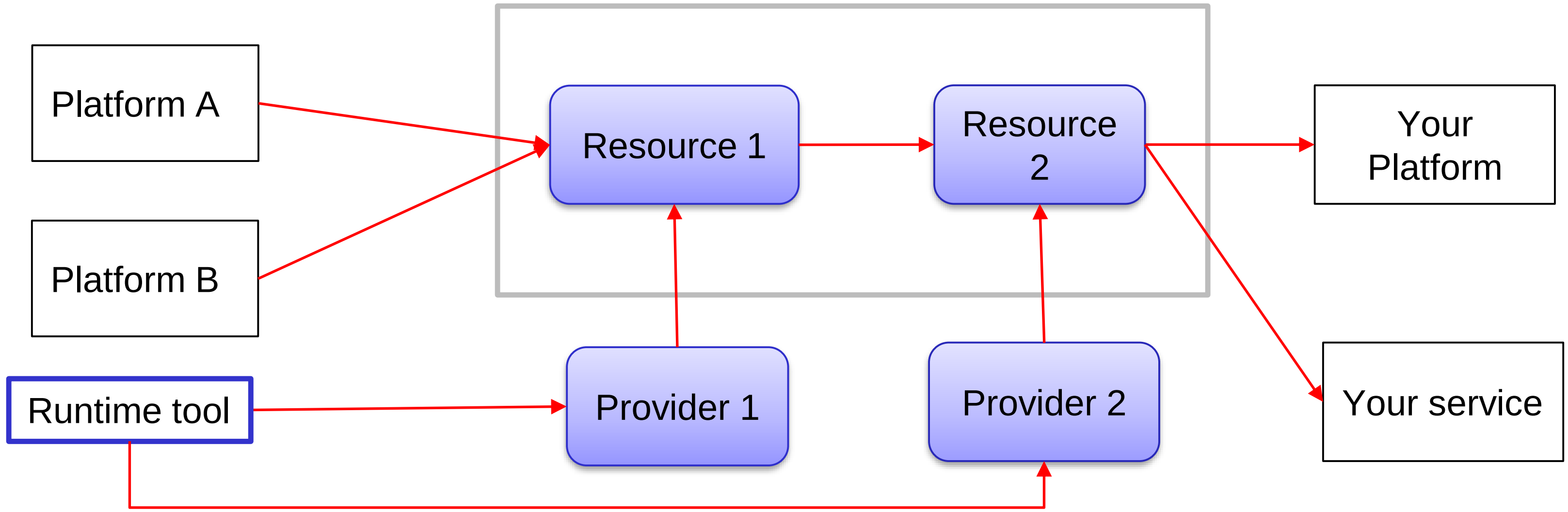


Virtual IoTData-as-a-Service Example



<u>Relevant papers: </u> Cao et al. :
MARSA: A Marketplace for Realtime Human Sensing Data. ACM Trans. Internet Techn. 16(3): 16:1-16:21 (2016)
A. Bröring et al., "**Enabling IoT Ecosystems through Platform Interoperability**," in IEEE Software, vol. 34, no. 1, pp. 54-61, Jan.-Feb. 2017.
M. Blackstock and R. Lea, "**IoT interoperability: A hub-based approach,"** 2014 International Conference on the Internet of Things (IOT), Cambridge, MA, 2014, pp. 79-84.

Focus 3: dynamic runtime pipelines for interoperability bridges



- Support fast development and deployment of service pipelines to solve interoperability issues
- Leverage state-of-the-art dynamic provisioning, service mesh, and K8s IoT/edge/Cloud software orchestration
- Leverage network functions from NFV

Focus 4: interoperability from regulation and quality view

- Platforms support the same technology stack but provided by different providers
  - e.g., cloudmqtt.com/cloudamqp.com versus self-deployed Mosquitto/RabbitMQ

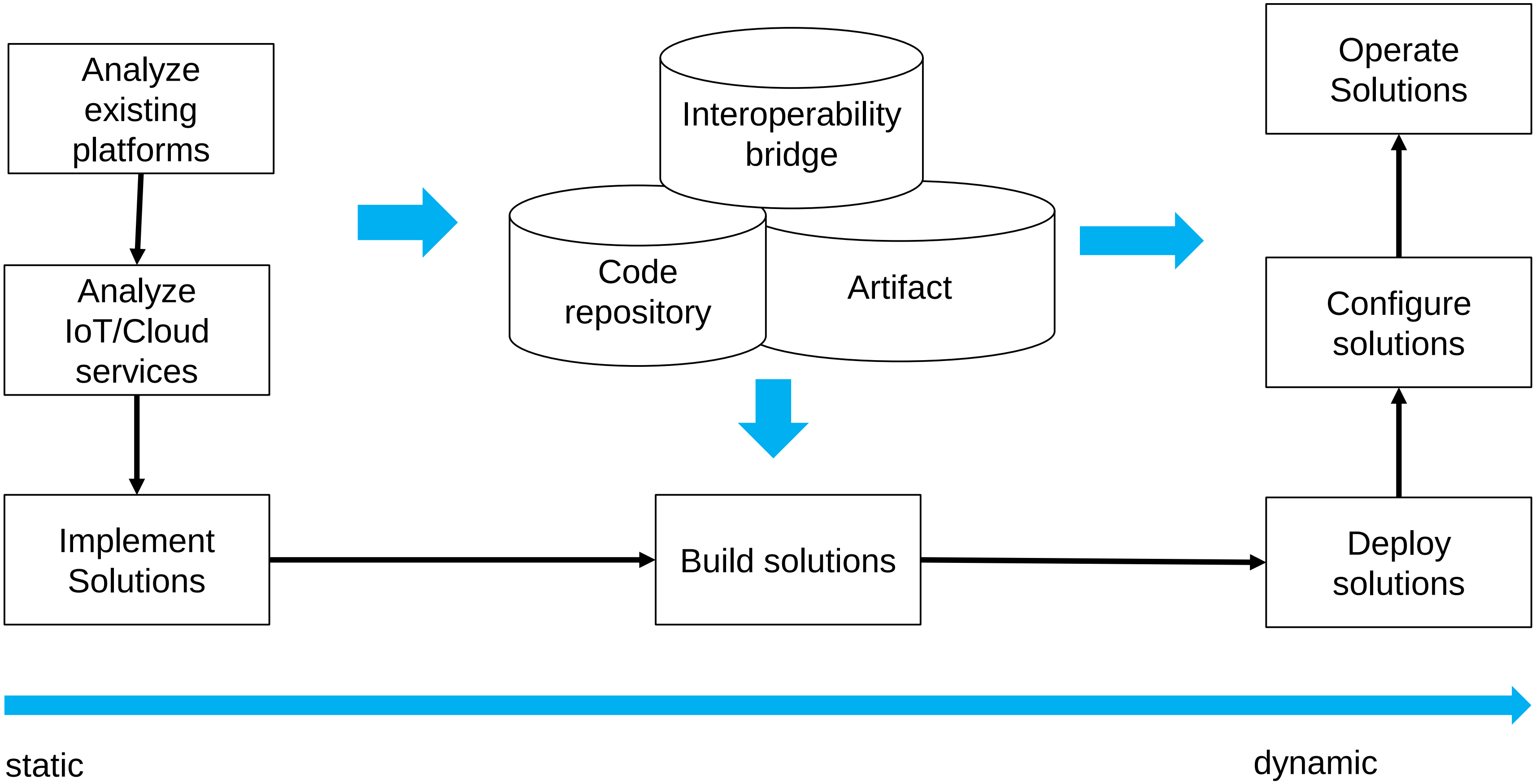
- How does deployment influence runtime interoperability?
    - E.g. regulation? Quality of services?

- How do different providers deal with interoperability w.r.t. regulation?

Focus 5: DevOps impact on solving interoperability issues

- DevOps is widely used for developing IoT Cloud software systems
- Tasks are related to IoT interoperability
  - building protocol translations (e.g., MQTT-to-Kafka), developing tasks for data transformation (e.g., using Logstash/Node-RED), etc.
- Many tasks for solving IoT interoperability are part of the software development lifecycle
  - Searching software artefact, automatic deployment of software, Infrastructure-as-Code for IoT platforms integration

Develop interoperability solutions as part of the software development - DevOps (not just an agreement)



Part 2

DevOps and Resource Slice for Dynamic IoT Interoperability

Resource slice

Resource slice and ensembles of IoT, edge and cloud resources

- Definition of resource slice:

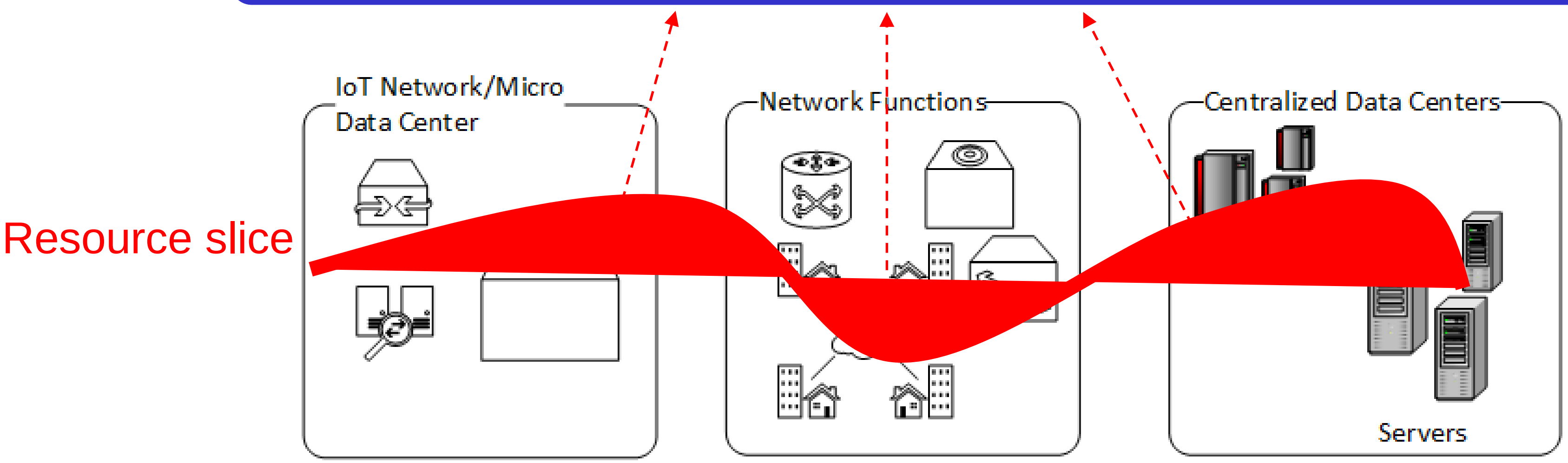
  a set of resources used in a specific context

- Context-specific resource slice
  - Types of resources
    - resources specifically built for the application
    - platform-specific resources
    - common resources
  - Layers: application, platform, networks, etc.

Source :
Resource slice concept and related papers: http://sincconcept.github.io
Hong Linh Truong, Nanjangud C. Narendra:
SINC - An Information-Centric Approach for End-to-End IoT Cloud Resource Provisioning. ICCCRI 2016: 17-24

Ensemble and its resource slice

Ensemble ={Resource slice, Requirements, Metrics, Policies, ..}

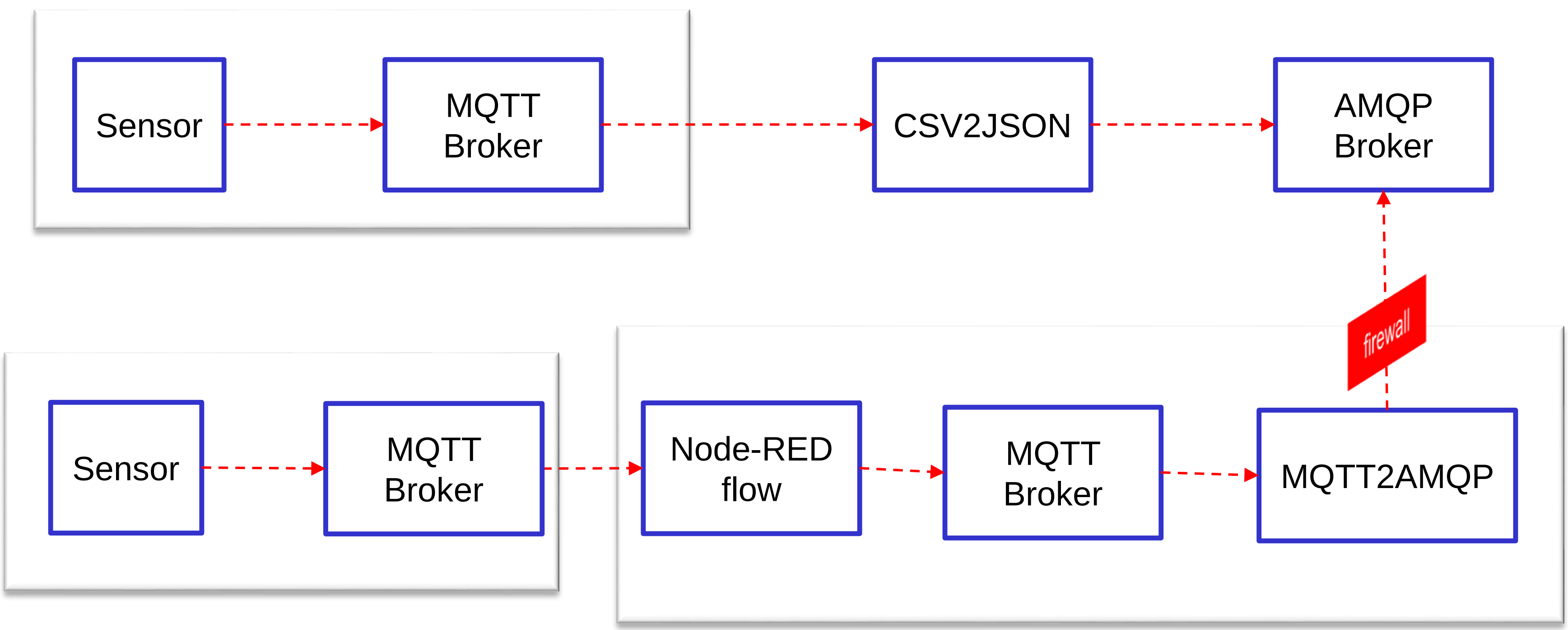


- Virtual resources as services: data streams, data transformers/analytics, messaging brokers, storage, firewall, etc.

Hong Linh Truong, Nanjangud C. Narendra, Kwei-Jay Lin:
Notes on ensembles of IoT, network functions and clouds for service-oriented computing and applications.
Service Oriented Computing and Applications 12(1): 1-10 (2018)

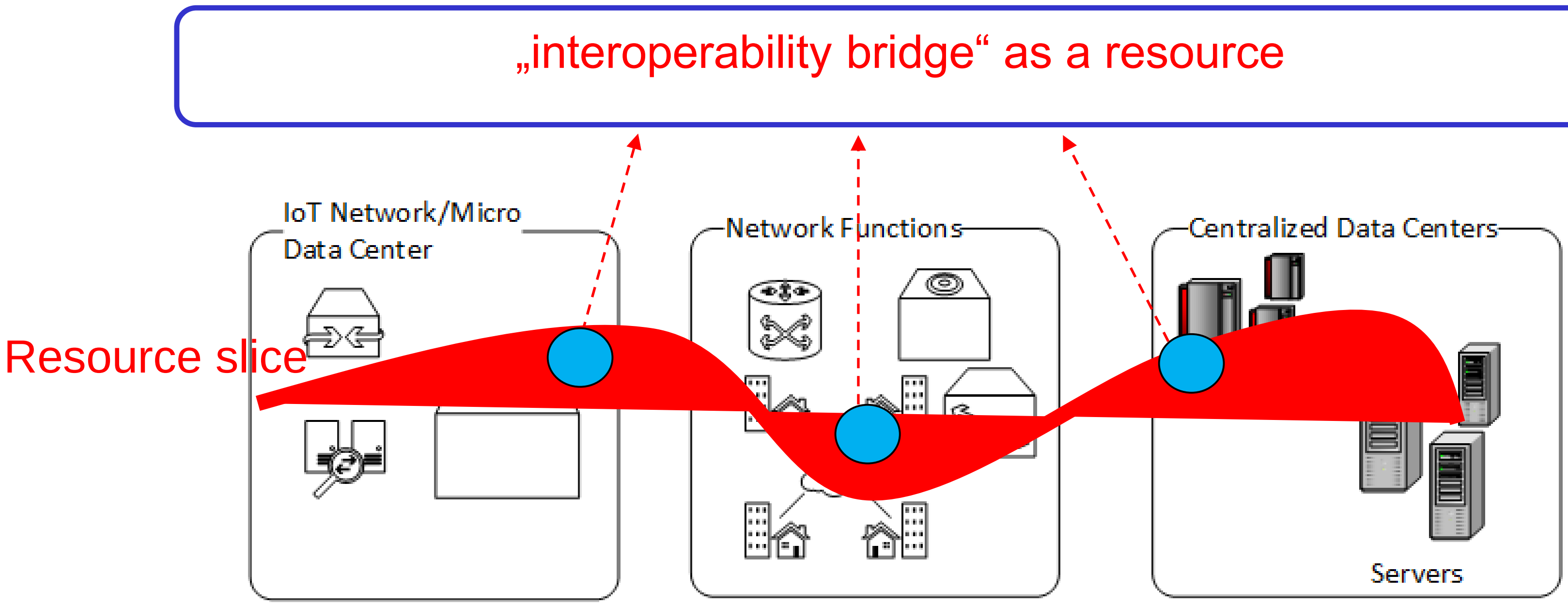
Examples:

For data transformation



Ensemble for solving interoperability issues

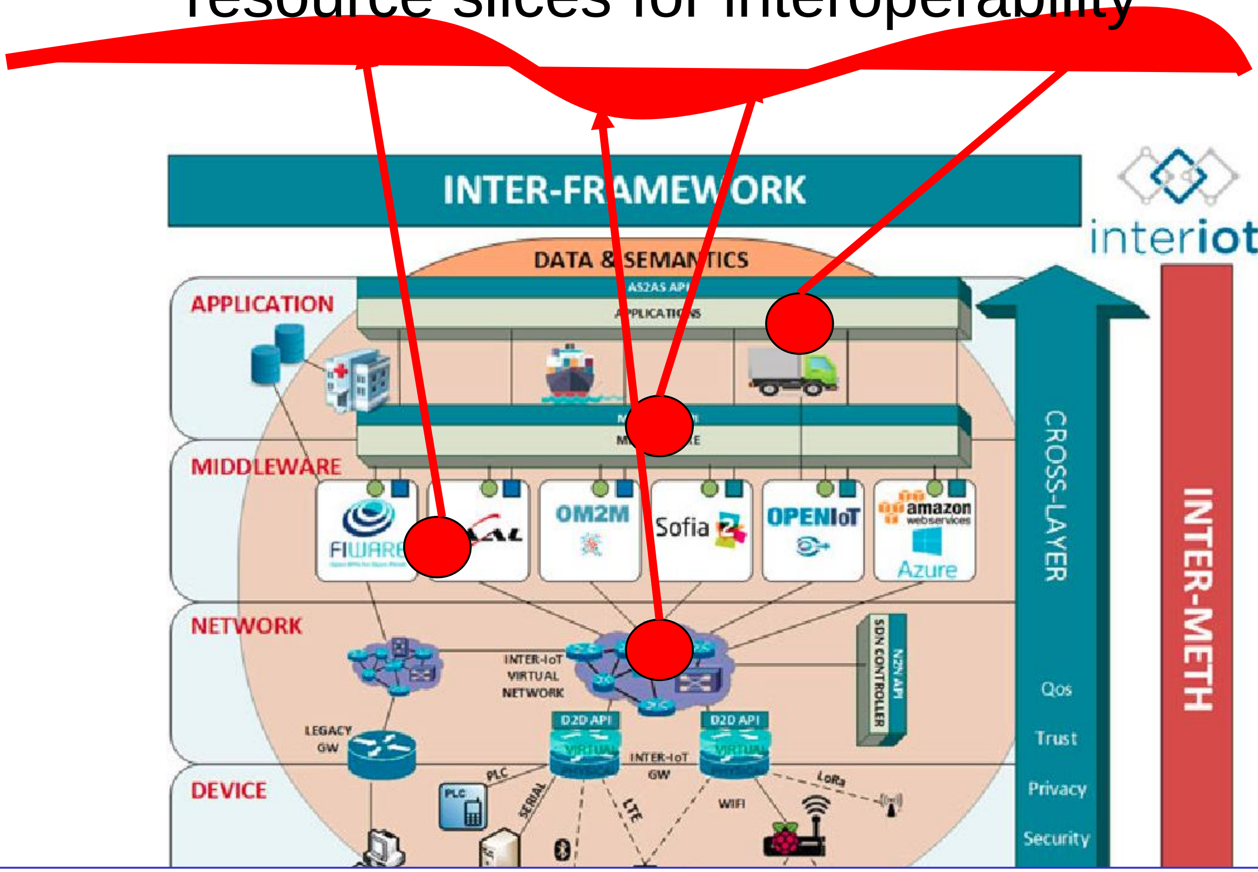
- A composition of “no-Interoperability-specific components“ can create a virtual resource for interoperability solutions



- Hong Linh Truong: **Towards a Resource Slice Interoperability Hub for IoT**. IC2E 2018: 310-316

Example: applying resource slice into INTER-IoT

resource slices for interoperability



INTER-IoT Figure source: Fortino G. et al. (2018) **Towards Multi-layer Interoperability of Heterogeneous IoT Platforms**: The INTER-IoT Approach. In: Gravina R., Palau C., Manso M., Liotta A., Fortino G. (eds) Integration, Interconnection, and Interoperability of IoT Systems. Internet of Things (Technology, Communications and Computing). Springer, Cham

Much work need to be done for dynamics

- Resources have to be in the form that can be virtualized and instantiated on-demand
  - Also require detailed, low-level information about resources

- Some static solutions must be adapted!
  - Just making metadata available is not enough
  - Dockerize static solutions, as an example of technical implementation

High-level principles for resource slice and interoperability

- 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, if needed

Examples

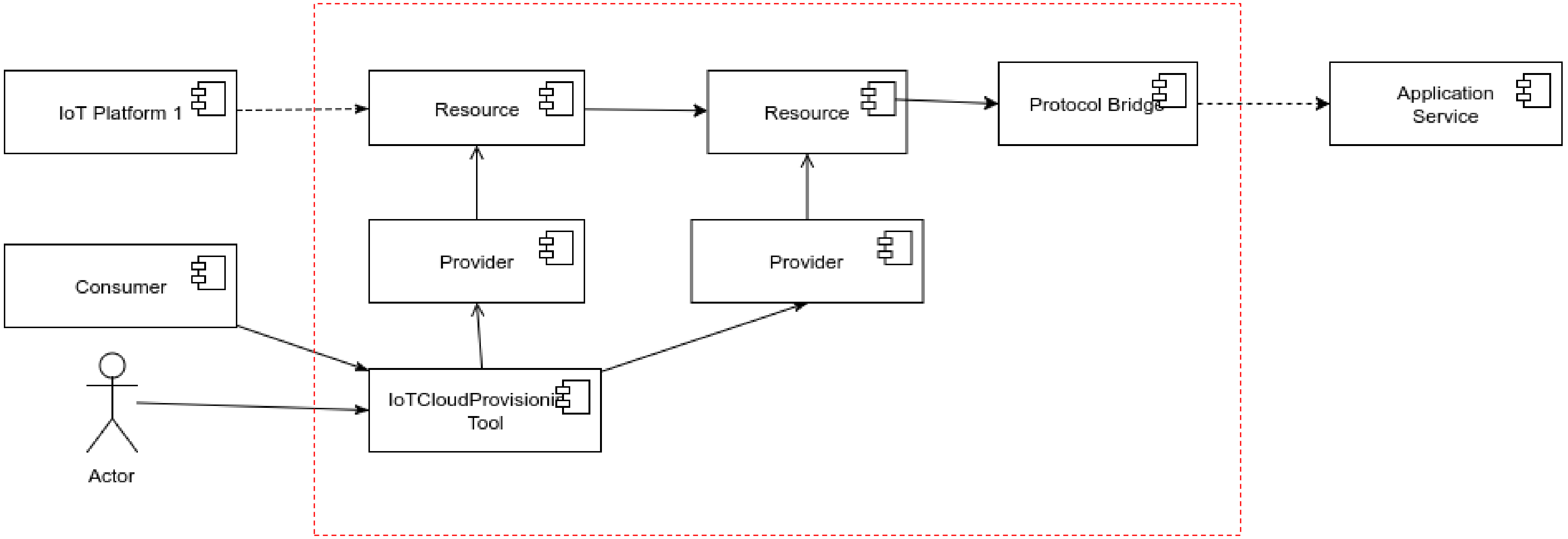
- `DP(r)= {dpa,dpc}` return all videos and the current video, and `CP(r)={cpp }` put video to a storage
- A service `S` can provide a set of $R=\{r_i\}$ allow to use `DP(r)` and `CP(r)`
- Examples of slices
  - `RS(c) ={ri, GoogleStorage, FaaS}`
  - `RS(c) ={ri, Kafka, Trigger, Container}`
- Our interoperable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

Integration requirements for dynamic interoperability

- **Developments**
  - 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
- **Operations**
  - Resource providers provision resources
  - Resources and providers can be controlled at runtime

IoT interoperability DevOps

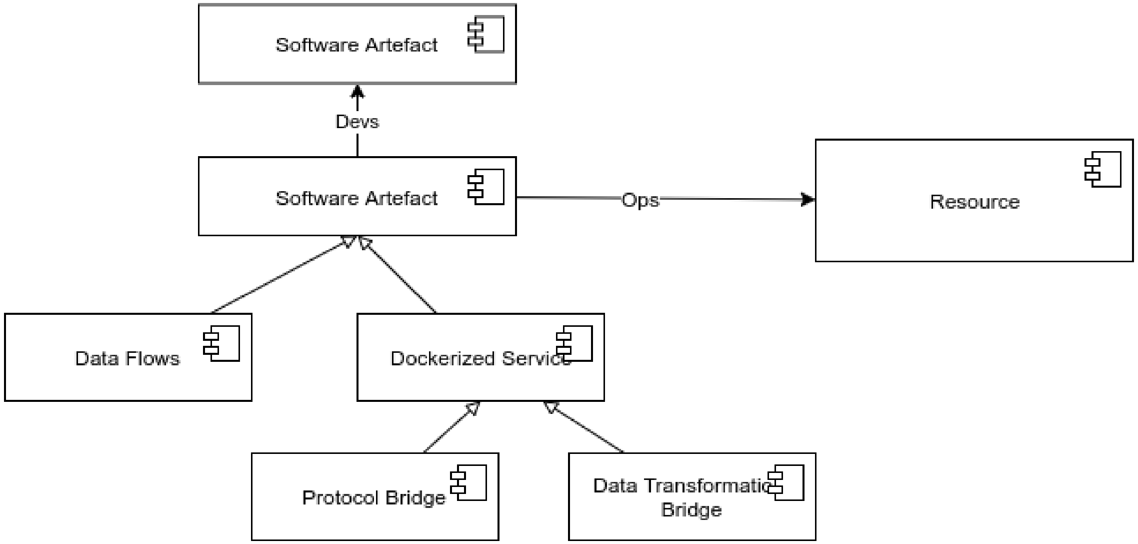
What are our typical tasks? (1)



Composition and deployment

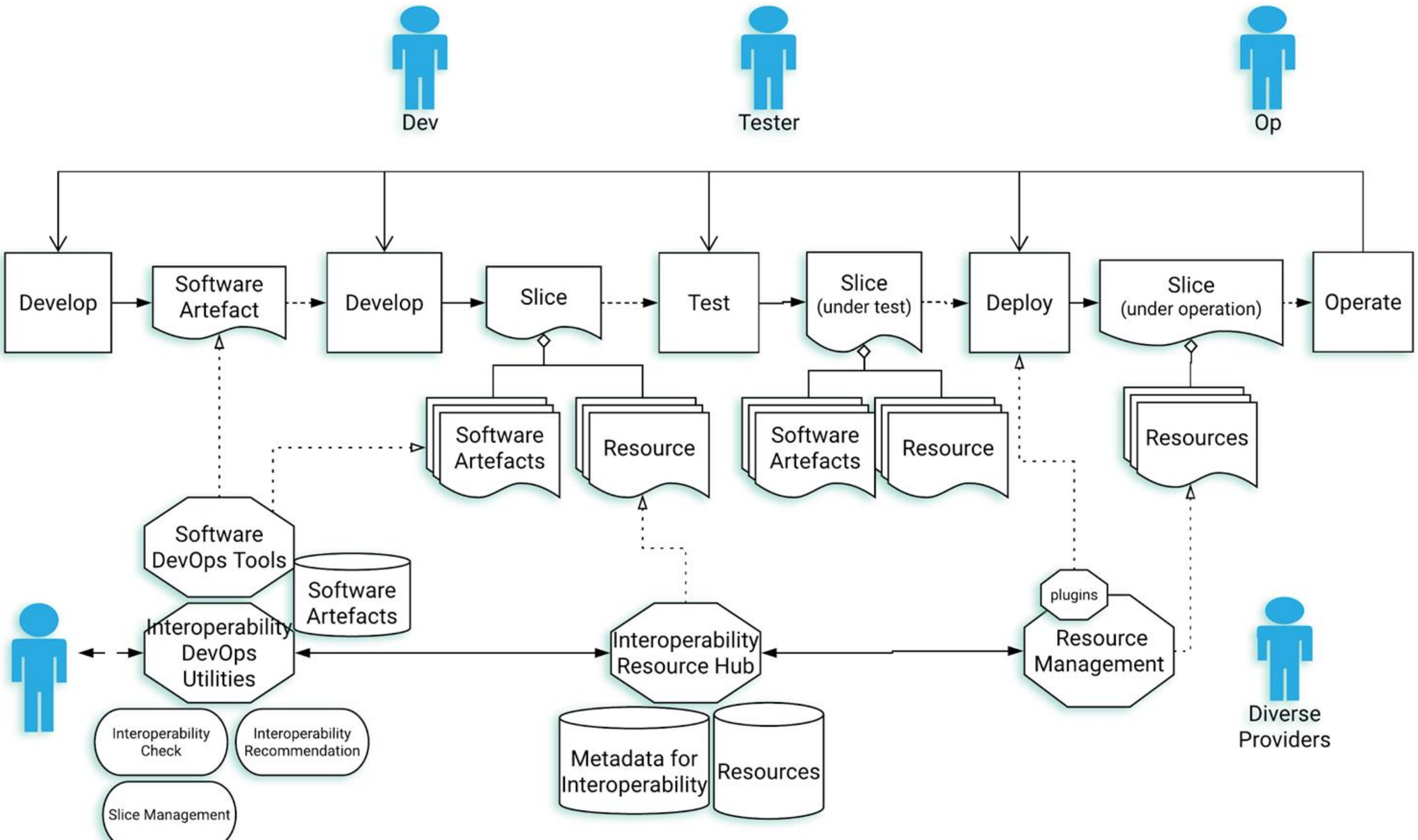
What are our typical tasks? (2)

- Create resources and reuse existing ones
- We do not want to create new artefacts, but if we have to, we want to do it fast



Dockerized bridge: https://github.com/rdsea/IoTCloudSamples/tree/master/IoTCloudUnits/csvToJson
Flow-based bridge:
https://github.com/rdsea/IoTCloudSamples/tree/master/IoTCloudUnits/node_red_dataflows

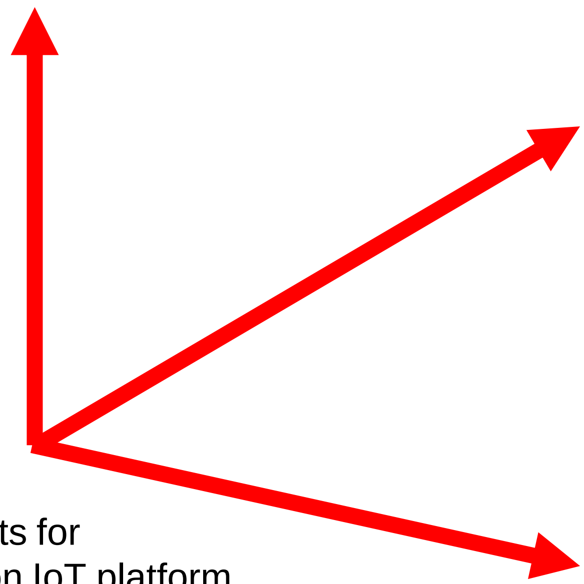
Our IoT interoperability DevOps process



Working paper: Hong-Linh Truong, Michael Hammer: **IoT Interoperability DevOps**. October, 2018

Important aspects

Metadata about resources and artefacts



Dynamic resource provisioning

Building and suggesting interoperability Bridge

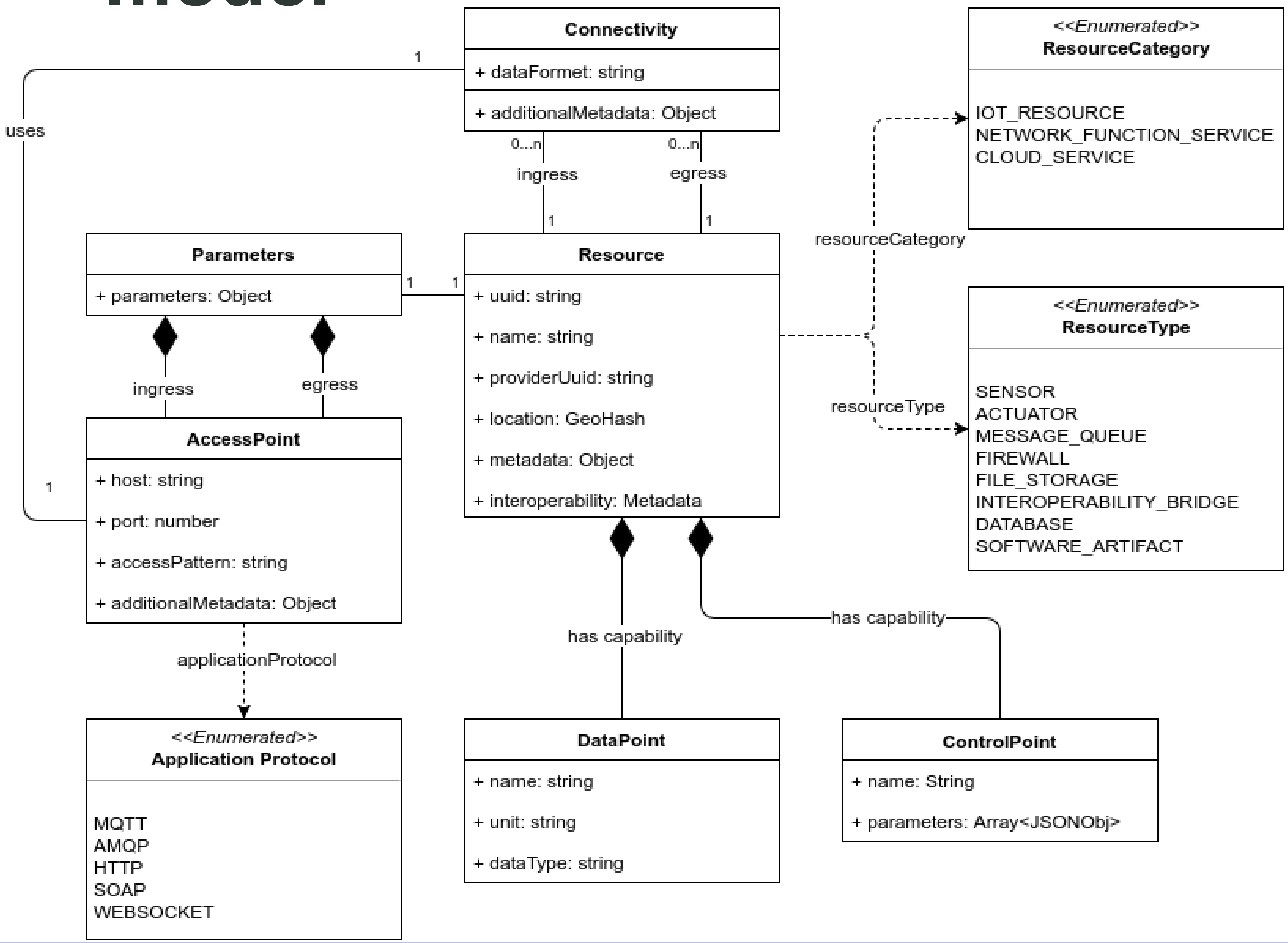
Key ingredients for implementation IoT platform interoperability solutions

Metadata for interoperability

Metadata

- Dynamic solutions need a lot of metadata for making dynamic provisioning
- Typical resource metadata and “interoperabity metadata” but in an extensible model
- We should focus more on novel aspects of *metadata reflecting dynamics for interoperability solutions* , e.g., quality, contract, delivery frequency (related to V* of data and scale of deployment)

Detail: core resource information model



Lingfan Gao, *"On Provisioning and Configuring Ensembles of IoT, Network Functions and Cloud Resources"*, Master thesis, TU Wien, Oct 2018
https://github.com/SINCConcept/HINC

Example of core resource information

- Resource types
  - SENSOR, ACTUATOR; MESSAGEQUEUE, FIREWALL, FILE_STORAGE, VPN, CONTAINER, VIRTUALMACHINE
  - FAAS, INTEROPERABILITY_BRIDGE, DATA_TRANSFORMATION, PROTOCOL_TRANSLATION
- Platform connectivity protocols:
  - REST, MQTT, AMQP, CoAP
- Underlying network connectivity for platform protocols
  - IP, LoRaWAN, NB-IOT

Resources information

Why it has to be detailed?

→ If we want to enable dynamic instantiation and configuration at runtime!

Example of a resource slice:

https://github.com/SINCConcept/HINC/tree/master/scenarios/btssensors

Interoperability metadata (1)

- Answering the question of which types of metadata are needed for IoT interoperability

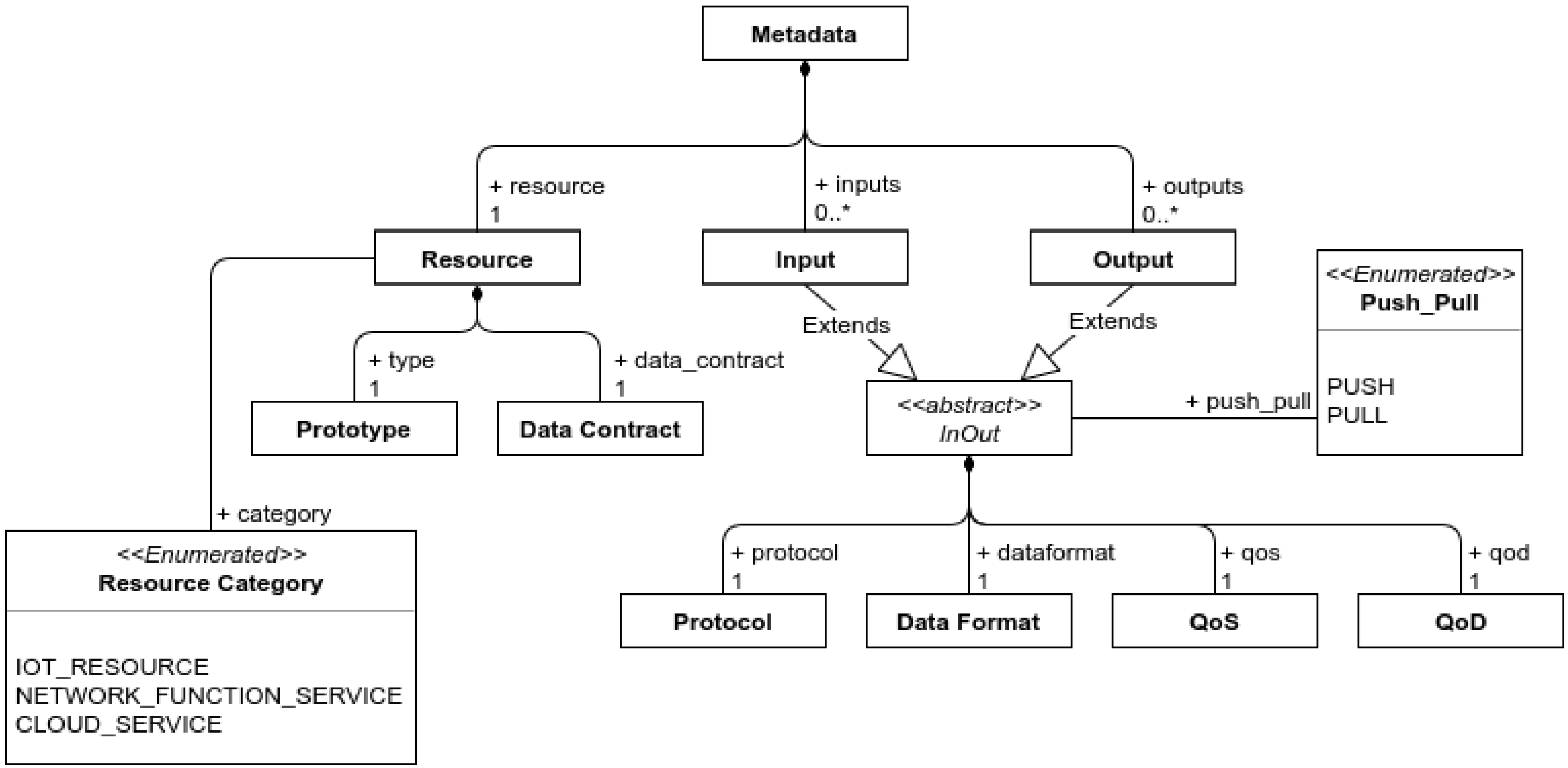
- Conventional metadata about IoT platforms
  - Data access patterns: PubSub (fan-out), ReqResp, Queue
  - Data format: CSV, JSON, AVRO, RDF
  - Data models (by existing standards)
  - Possible service/artefacts for dealing with data transformation and transfer
  - Possible service/artefacts for dealing with protocol bridges

Interoperability metadata (2)

- Novel types related to contract and regulation:
  - Data contract: ownership/regulation
  - Quality of data: completeness, accuracy, timelineness, etc.
  - Data delivery: frequency (e.g. sampling rates)
  - Execution policy and data regulation: e.g., within EU

- Other interesting metadata
  - Price if the interoperability bridge is from marketplaces
  - License, e.g. for software artifact

Our Interoperability metadata



Working paper: Hong-Linh Truong, Michael Hammer: **IoT Interoperability DevOps**. October, 2018

Artefacts metadata examples

- Multi-protocol CVS2JSON Dockerized service

  https://github.com/rdsea/IoTCloudSamples/blob/master/IoTCloudUnits/csvToJson/metadata.json

- HTTP-to-Google Storage

  https://github.com/rdsea/IoTCloudSamples/blob/master/IoTCloudUnits/http2datastorage/metadata.json

Dynamic resource provisioning for building interoperability solutions

How do we leverage the state of the art service provisioning and configuration for IoT interoperability

What will be provisioned?

- Possible services for data conversion, data message, protocol bridge at the platform level
- Possible user-defined services for data transformation and data transfer
- Provisioning
  - Request resource providers to do this
  - Take artefacts and ask suitable providers to run artefacts, e.g, flows and containers

Resource provisioning and configuration

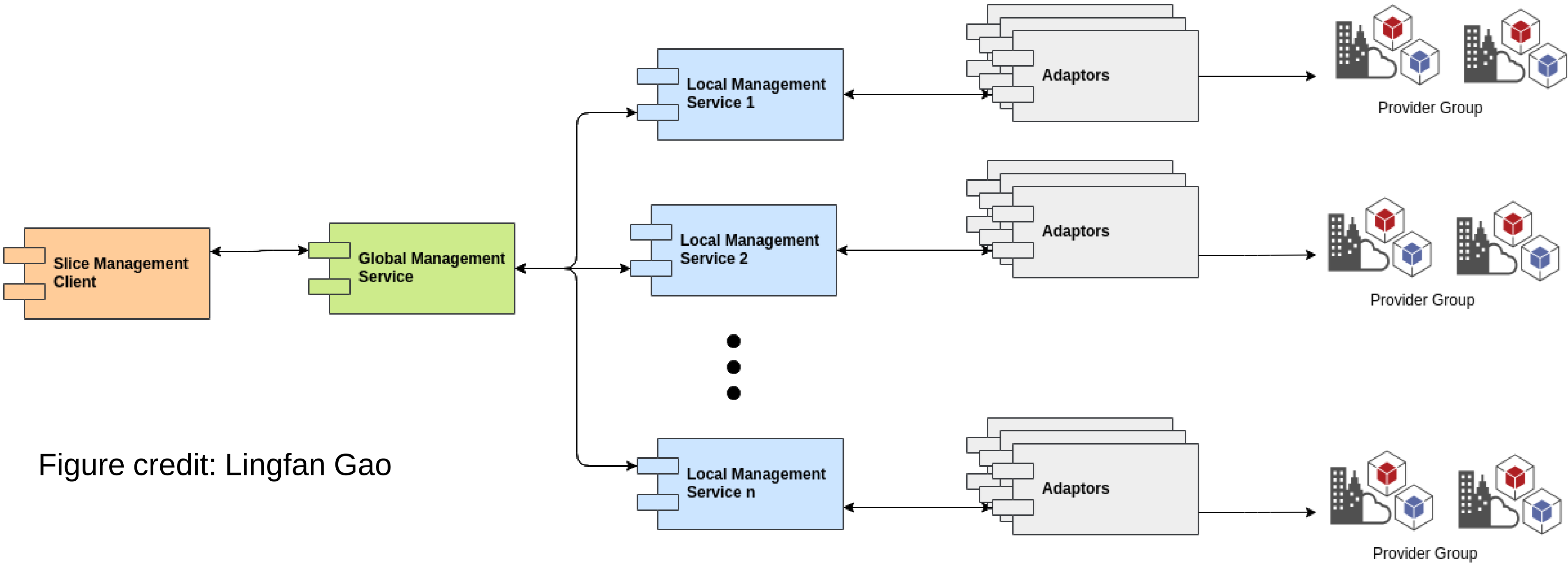


Figure credit: Lingfan Gao

- Resource slice concept and related papers: http://sincconcept.github.io
- Hong Linh Truong: **Towards a Resource Slice Interoperability Hub for IoT**. IC2E 2018: 310-316
- Lingfan Gao, *"On Provisioning and Configuring Ensembles of IoT, Network Functions and Cloud Resources"*, Master thesis, TU Wien, Oct 2018

Provisioning situations

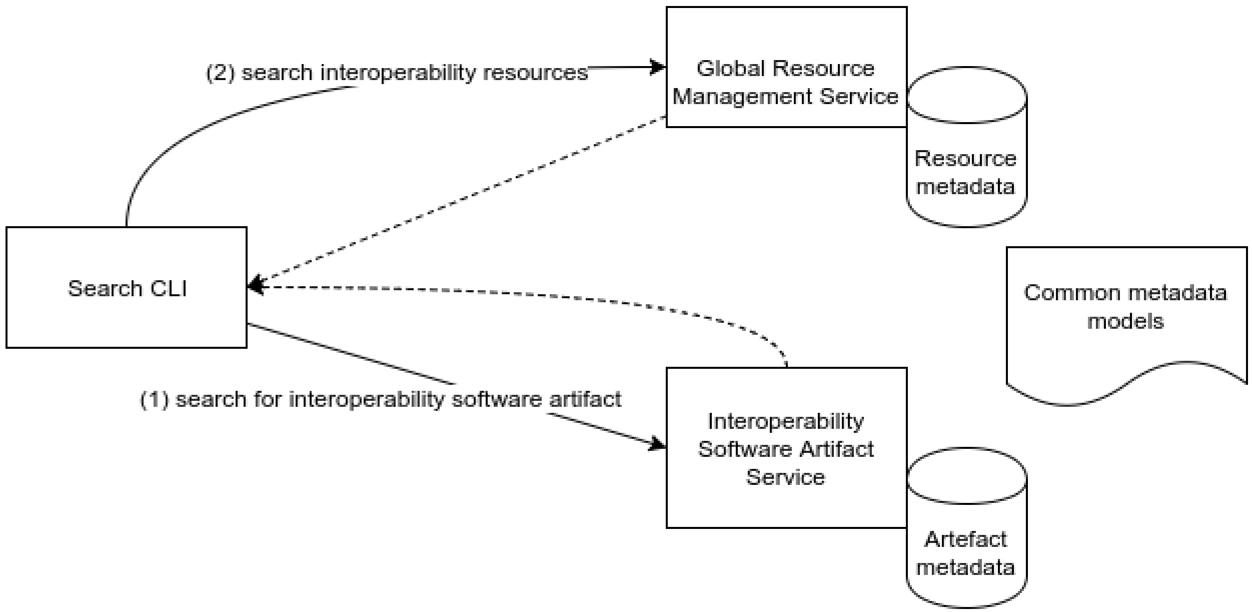
- Case 1: after selecting suitable components → deploying → solutions
  - but it is different from static because we leverage dynamic features to enable runtime solutions based on runtime needs

- Case 2: runtime dynamic selection and control of providers → solutions
  - It is possible to adapt/reconfigure resources/artefacts

Providers

- Known providers
  - Protocol bridges
  - Message brokers: e.g., MQTT, AMQP
  - Service engine: e.g., Docker and Tomcat/Jetty
  - Process engine: e.g., Node-RED, KSQL, and Flink
- Many resources can be dynamically provisioned
  - But they have not put into the resource-as-a-service model
  - Deploy your own Node-RED versus ask a Node-RED provider to do this

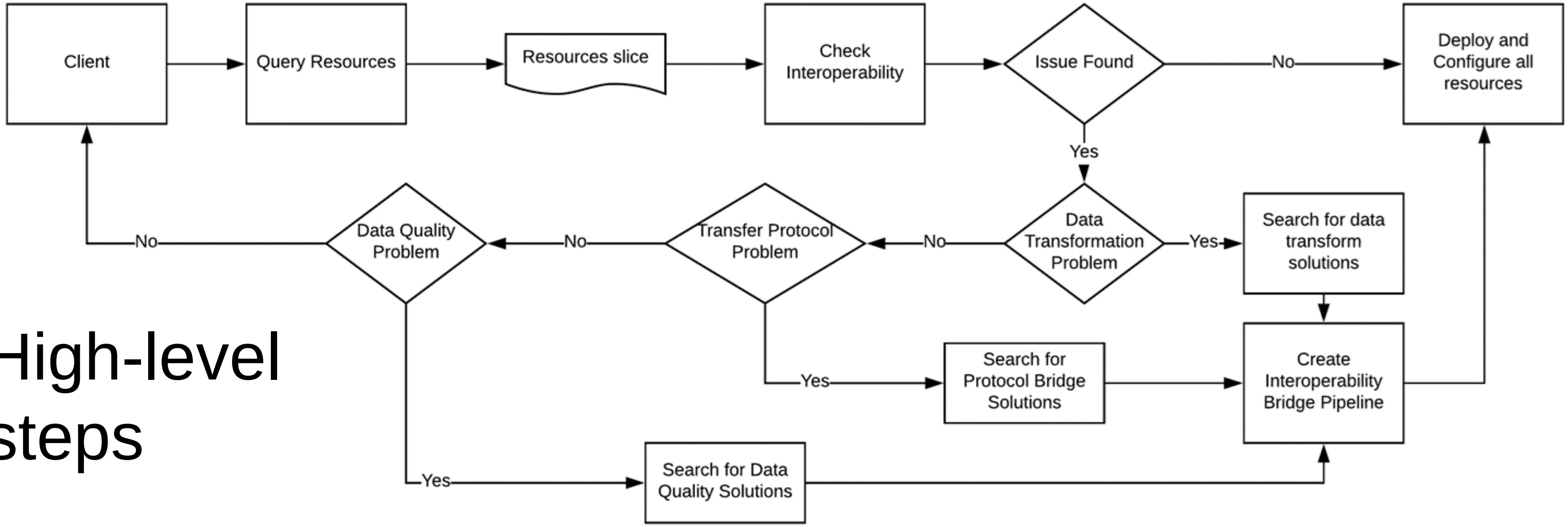
Search for interoperability artefacts and resources



- Leverage JSON-based/document based search.
- Currently based on MongoDB features but will be extended
- Requires up-to-date and rich metadata
- Many software artefacts have been developed but we need to make the right metadata for them

Interoperability Check

Check using metadata and the graph of resource slices



High-level steps

rsiHub - Resource Slice Interoperability Hub

- Resource Providers
  - Within IoTCloudSamples + third parties
- Adaptors
  - If you want resources to be controlled by rsiHub

- Local Resource Management
- Global Resource Management
- Interoperability Software Artifact Service
- Interoperability Recommendation Service

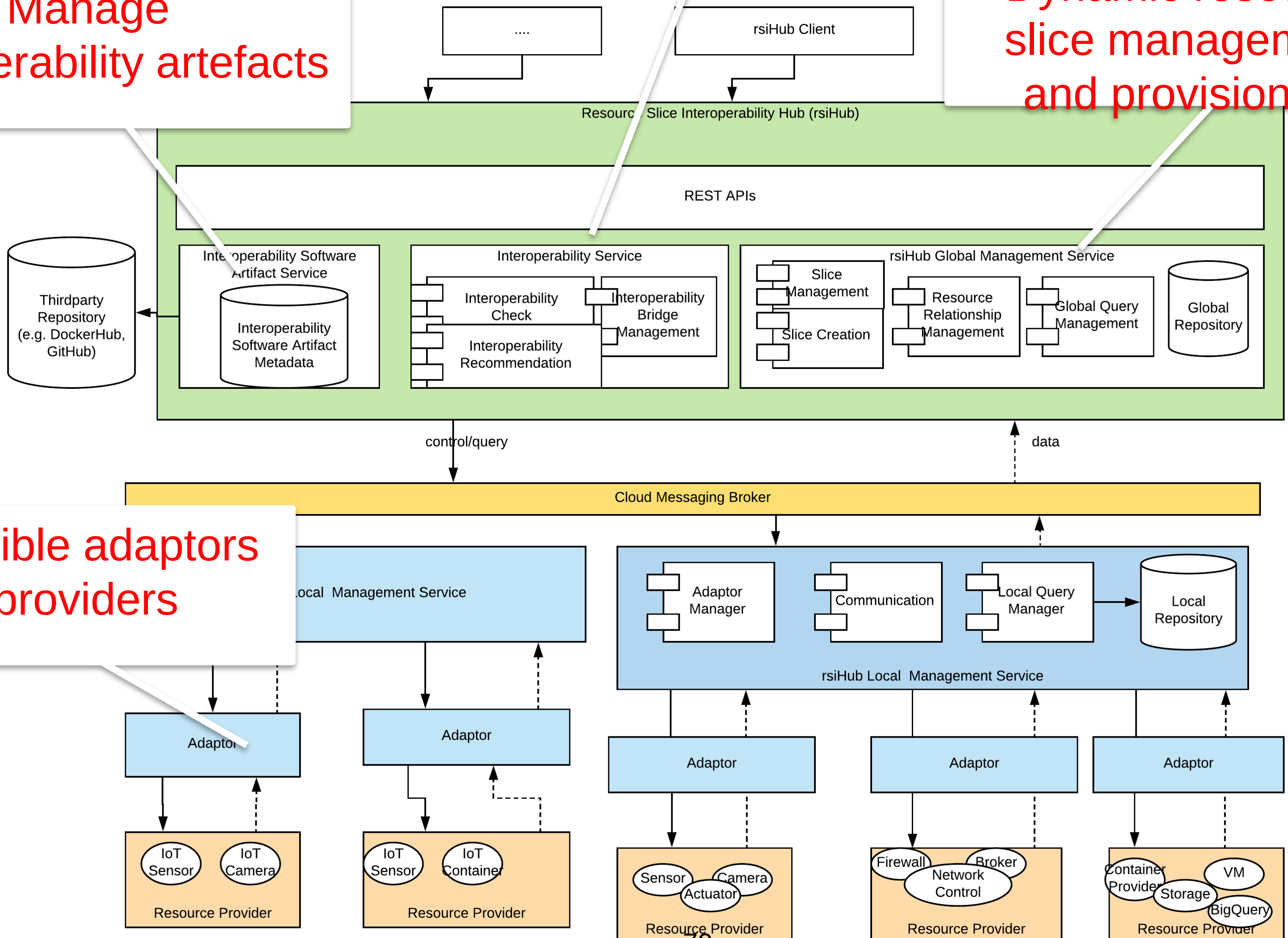
- rsiHub CLI: pizza

Search, check and recommend interoperability artefact/resources and bridges

Manage Interoperability artefacts

Dynamic resource slice management and provisioning

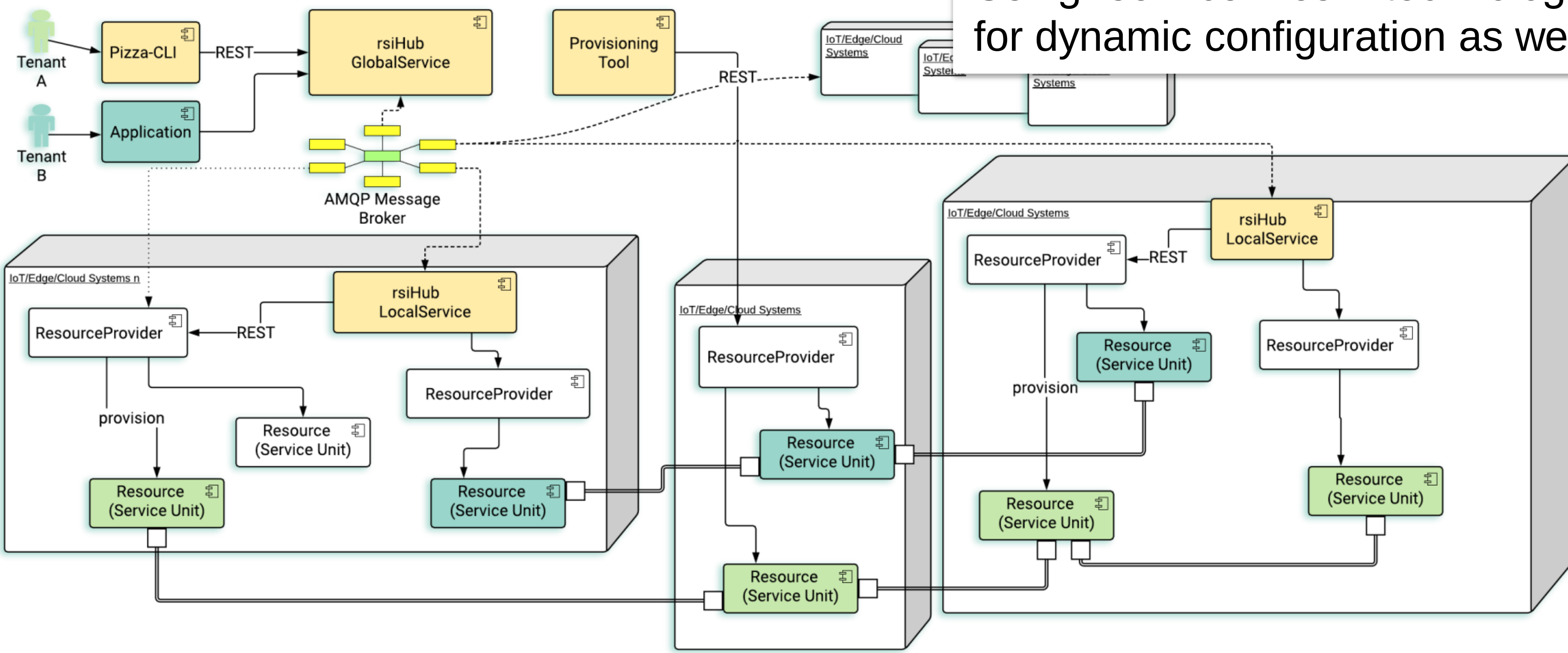
Extensible adaptors to providers



ACM I

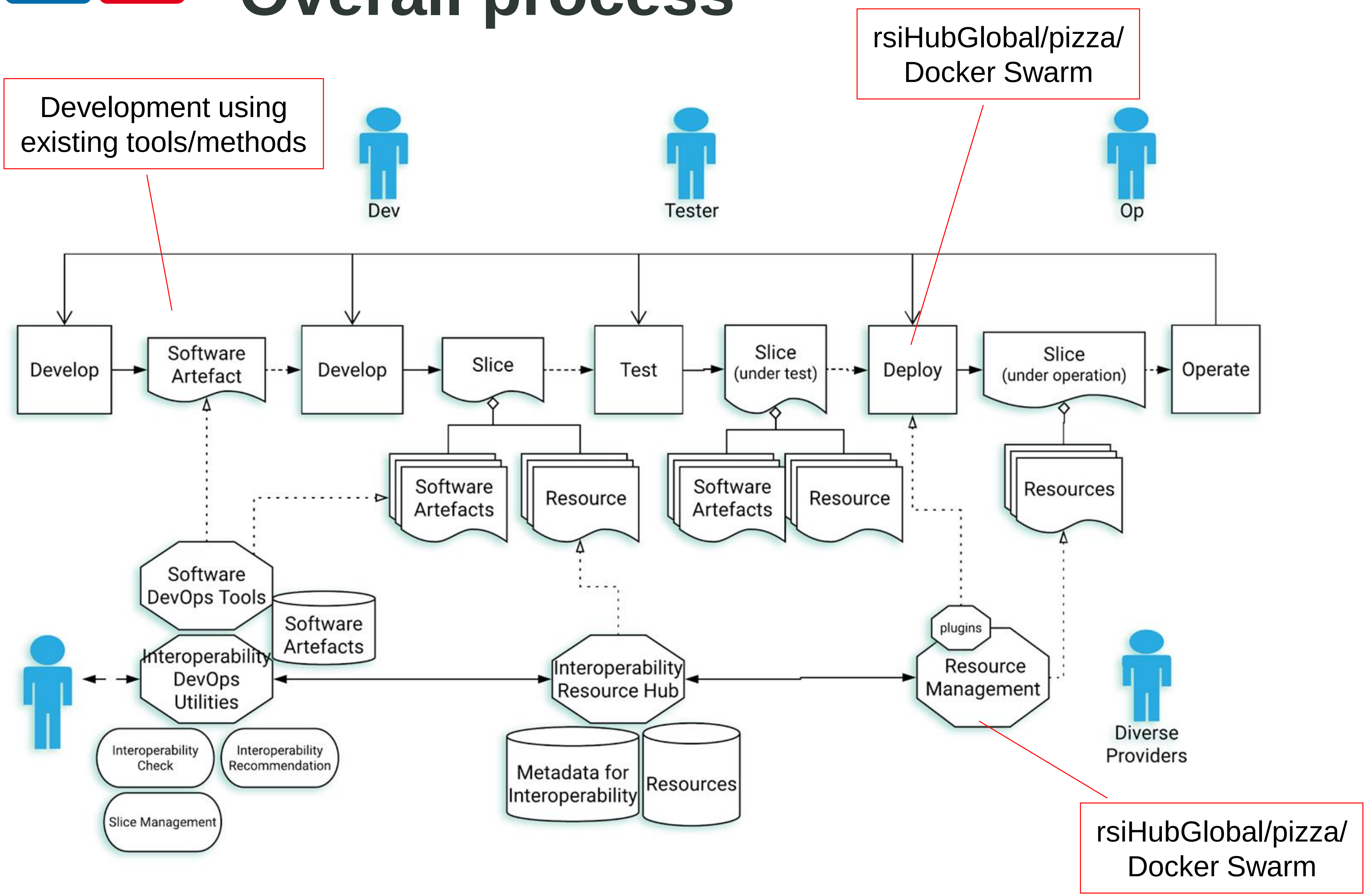
In Action

Using “service mesh” technologies for dynamic configuration as well



- Check our prototypes and examples
- rsiHub: https://github.com/SINCConcept/HINC
  - IoTCloudSamples: https://github.com/rdsea/IoTCloudSamples
  - RDSEA docker hub: https://hub.docker.com/u/rdsea/
  - IoT 2018 Tutorial: https://github.com/rdsea/iot2018tutorial

Overall process



Running use cases and examples

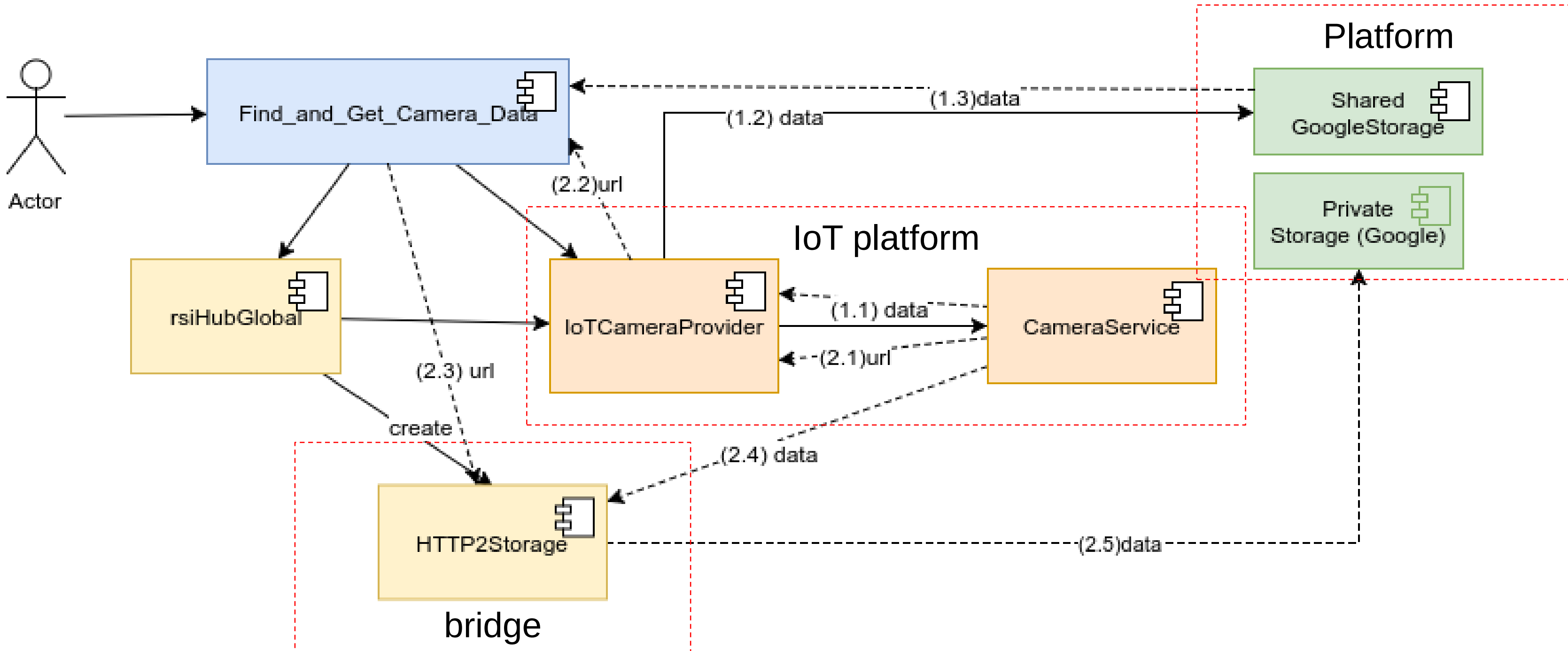
Prototypes and testbed

- Two prototypes
  - rsiHub: https://github.com/SINCConcept/HINC/
  - IoTCloudSamples: https://github.com/rdsea/IoTCloudSamples
- Testbed
  - Google Cloud Platform
    - for Cloud services and for emulating edge/IoT platforms
    - for emulating IoT sensors
- Realistic dataset or emulating dataset for sensors
- Real service providers and emulating services for application domains

Example: dynamic platform and application protocol bridges

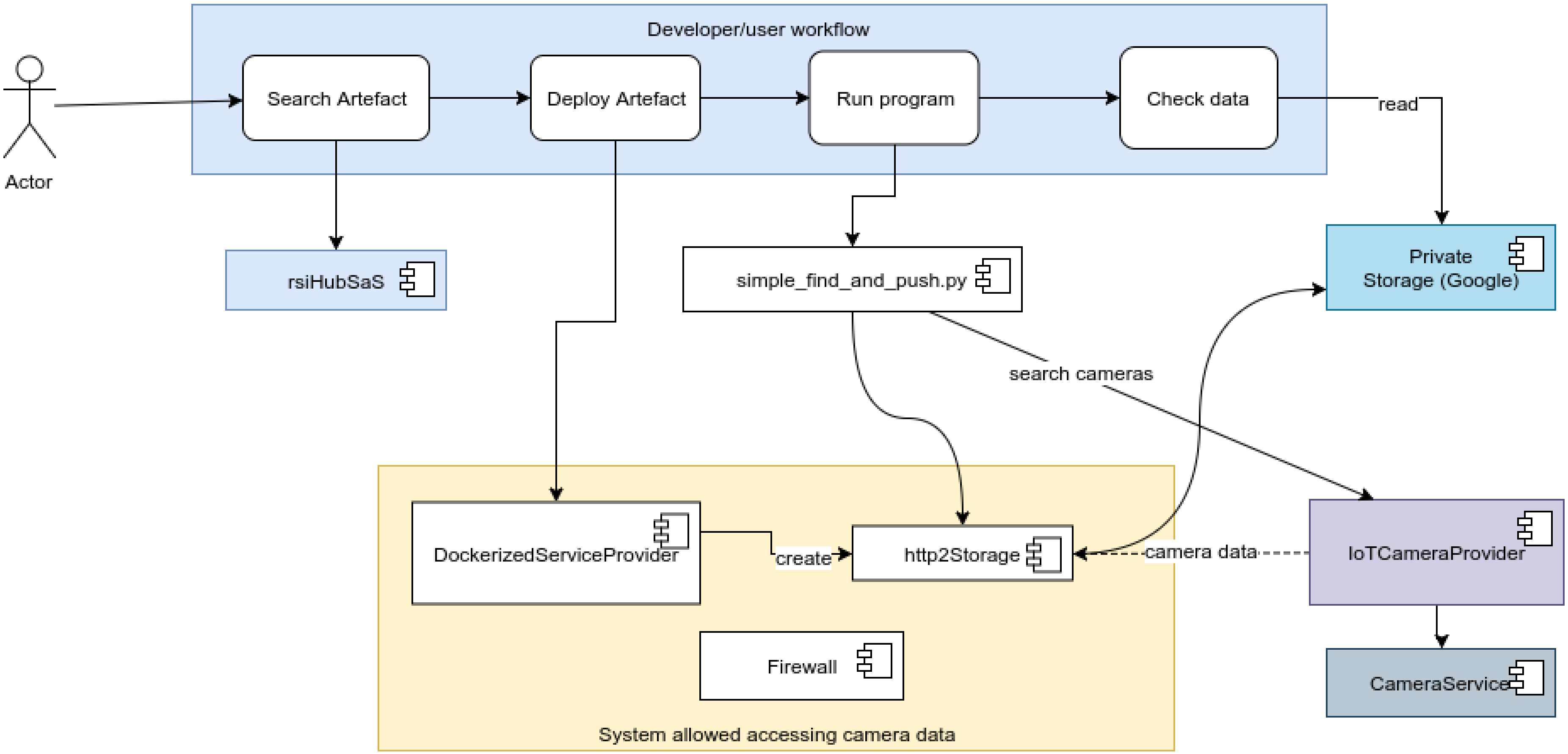
- Inside a city we have a lot of cameras, there is an IoTCamearaProvider manages such cameras
  - Only REST GET for downloading camera data for clients running in certain systems
- Given a situation we need push camera data to a storage of another IoT system (e.g., for running combined analytics)
  - Search suitable bridges
  - Deploy interoperability bridges
  - Control bridges to perform data movement

Protocol bridge and other services



Similar use cases when we need to bridge different protocols on-demand: CSV2JSON, MQTT2AMQP, PubSub2CoAP, INTER-IoT Gateways etc.

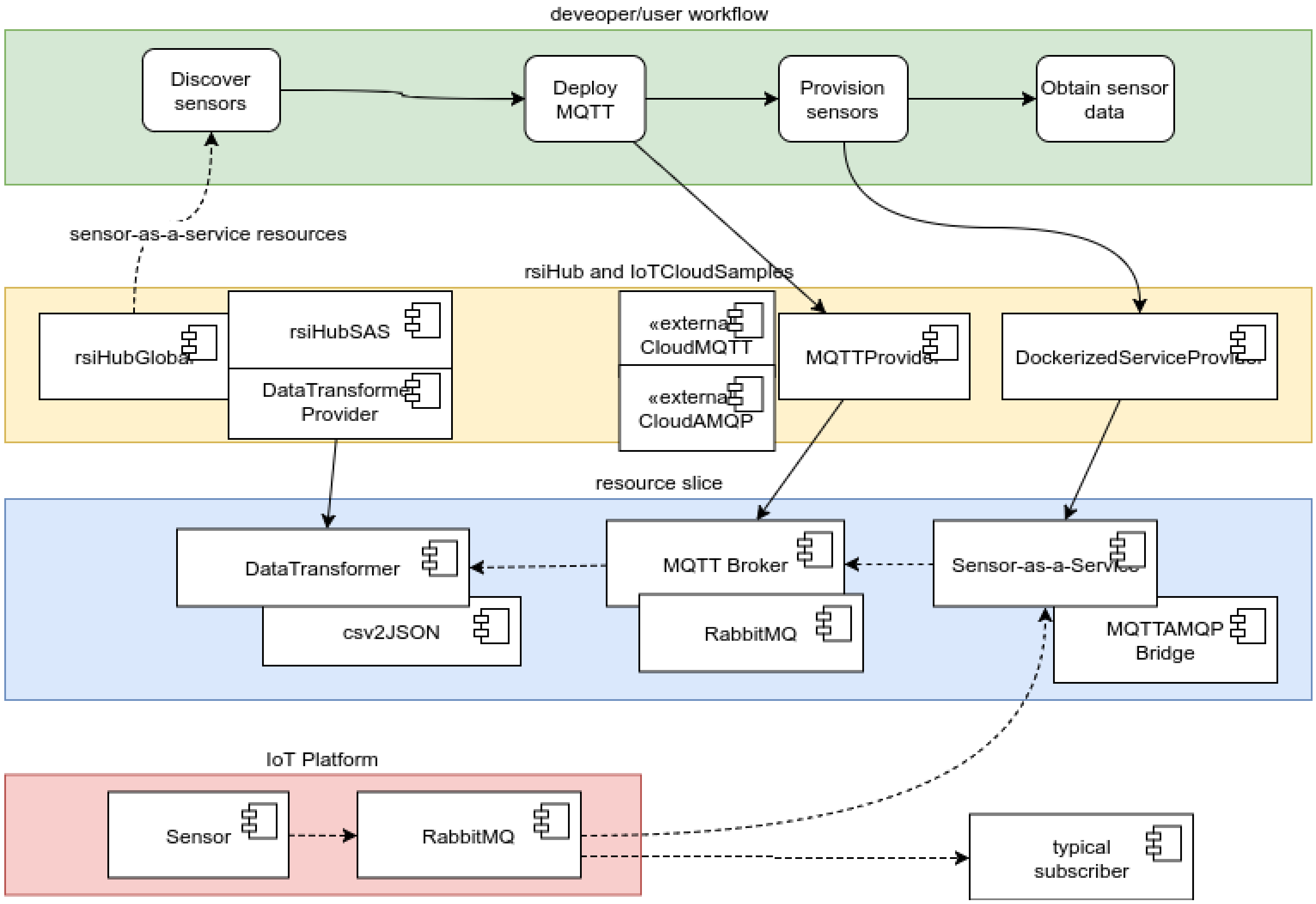
User actions and resource slice



Example: IoTData-as-a-service and dynamic service provisioning

- Our goal
  - Turn IoT platforms present data streams into on-demand data-as-a-service but at the fine-grained access
  - Enable dynamic usage
- IoTdata-as-a-service implementations
  - Different types of lightweight dockerized services
  - Flows deployed as a service
- The consumer decides to create, connect, and receive data

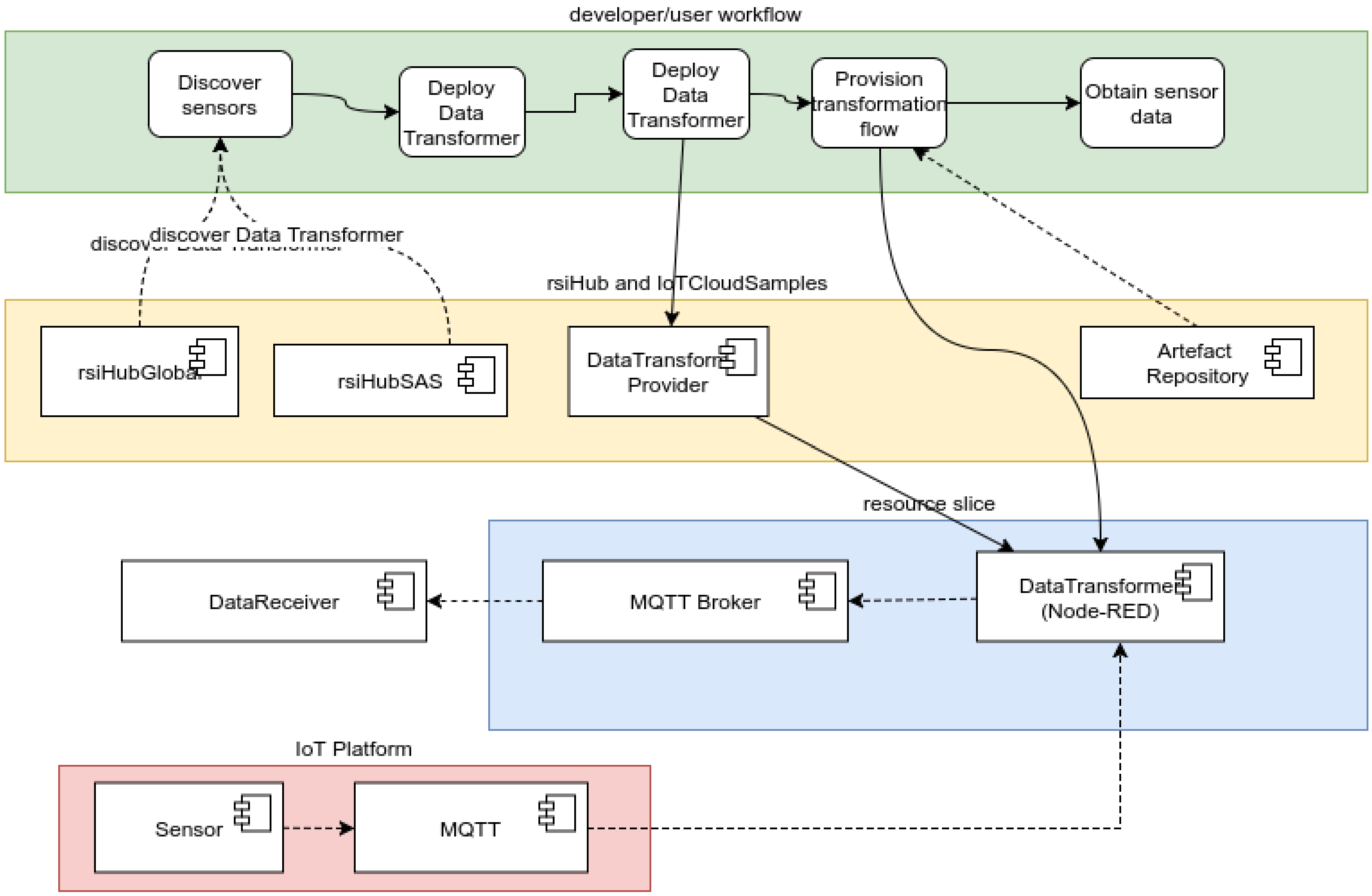
User actions and resource slices



Example: dynamic data transformation

- For interoperability of data (semantics or syntax): data transformation and process are needed at runtime
- Finding artefacts and deploy artefacts for transformation
  - Dependent also on the availability of process engine
- The principle can be applied for different types of engines

User actions and resource slice



Example: complex dynamic resource slice creation and adaptation

- Creating entire resource slices and update slices
- Video https://storage.cloud.google.com/rdsea-public/rsihub-demo_ecsa_final.mp4

Hong-Linh Truong, Lingfan Gao, Michael Hammerer:
Service architectures and dynamic solutions for interoperability of IoT, network functions and cloud resources.
ECSA (Companion) 2018: 2:1-2:4

Summary

Conclusions

- IoT interoperability among platforms are complex
- Different approaches: static approach versus dynamic one
  - We need to deal with new characteristics from IoT platforms and IoT

- Our approach – solving interoperability issues through the development and operation:
  - Ensembles of IoT, network functions and clouds
  - Dynamic provisioning and configuration
  - DevOps and microservices engineering

Future work

- Tutorial is still built on on-going research work
  - Make ideas and tools robust
    - metadata models about resources for interoperability resource slices and bridges
  - Extensive tests

- Incorporating security and access control features for resource slices for interoperability solutions
- Check http://rdsea.github.io for further development and https://github.com/rdsea/iot2018tutorial for further update

Thanks for your attention!

Hong-Linh Truong

Faculty of Informatics
TU Wien
rdsea.github.io