Scaling IoT Services:

Benefits of Software-Defined Networking (SDN) and Service Function Chaining (SFC) Techniques

Christian Jacquenet

christian.jacquenet@orange.com

Outline

- Context
- IoT networking
- IoT service design
- SDN and SFC frameworks for optimized IoT service delivery and operation
- A drone-monitored multi-field irrigation system use case
- Conclusion

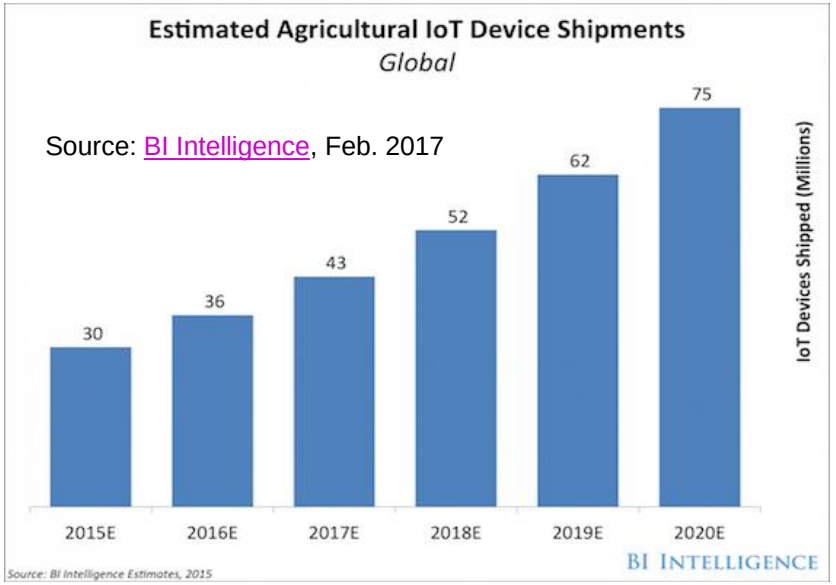
Context

- Foreseen tens of billions of objects (sensors, actuators, controllers, *etc*.) deployed for a plethora of usages
  - Objects communicate over a networking infrastructure by means of various, possibly service-inferred designs
- IoT infrastructure varies in scope (home, access, metro and beyond) and techniques (wired, wireless, a combination thereof)
  - Depending on the nature of the service

Agricultural IoT

- Est. 75M of shipped devices in 2020
- Many IoT projects are up and running, *e.g.*,:
  - EU's Food and Farm 2020 "IoF2020" 30M€-funded project
  - Sensor network to preserve water in Kansas
  - Bangladesh's e-Village project to boost agricultural production with sensors
  - *Etc*.

Source: e-agriculture.org , Feb. 2017



On IoT Networking

- Network scale is different
  - Up to several thousands of nodes, yielding path computation issues with current routing protocols
- Technology and environment are (often) constrained
  - Low powered devices, unstable communication links and outdoor installation raise new (security) issues
- Traffic patterns are very specific
  - Data are collected by sensors and forwarded to actuators and/or Internet “gateways” (*e.g.*, CPEs), yielding N:1 group communication scheme
  - Control traffic (*e.g.*, commands) is also very unidirectional, yielding 1:P group communication schemes
    - P2P traffic patterns remain somewhat marginal

Why IP?

- A federative layer
  - Cornerstone of the “Internet of Things” for the sake of interoperability and E2E paradigm
- IP is ubiquitous and scalable
  - Anything can be transported over IP, covering both wireless and mobile environments
    - In particular, IP is a more straightforward option than E164 numbering
- Current SoA proves IP implementations are lightweight, *e.g.*,:
  - TCP/IPv6 stacks can operate with 4kB RAM and 24 kB flash memory
  - ZigBee implementations require ~20 to 60 mW
    - For 1mW of power, a range of 10 to 100 meters and a data rate of 250 kbit/s

A Few Additional IoT Challenges

- Scalability and performance
  - Potential large scale requires efficient name resolution and forwarding paradigms
- Dynamic service discovery
  - Services for things must be identified for proper operation
  - Requires appropriate semantics to describe service capability
    - Can also be user-driven (*e.g.*, human/machine interaction for dynamically geo-located parking facilities)
- Mobility
  - Locate things and the services they support while in motion
- Security and privacy
  - Selective access to specific services (*e.g.*, field irrigation system)

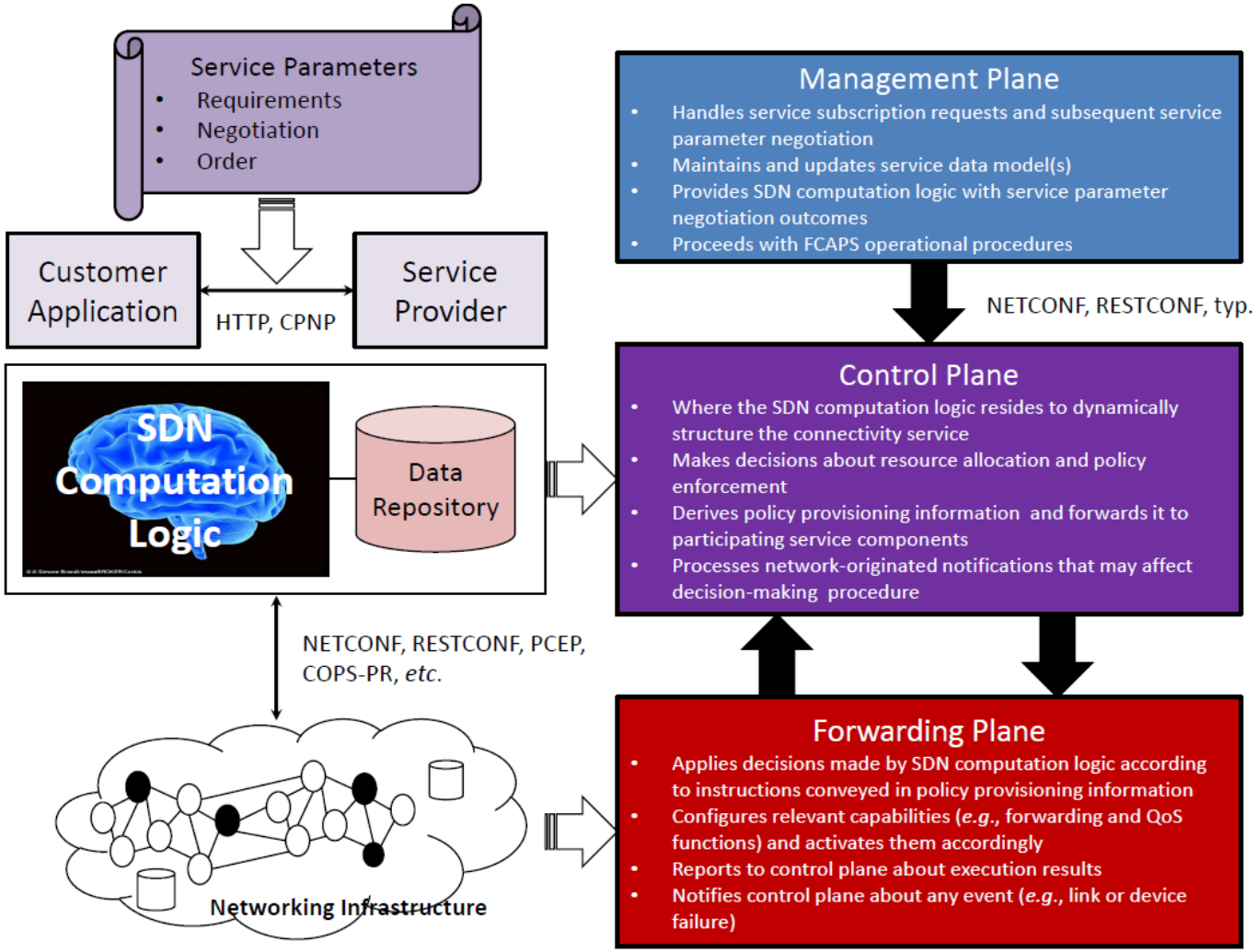
On IoT Service Design

- IoT service design assumes the combined and possibly ordered activation of elementary capabilities or Service Functions (SF)
  - Forwarding and routing, firewall, QoS, DPI, *etc*.
- IoT service complexity suggests robust mastering of chained SF activation and operation
  - For the sakes of optimized resource usage and reliable service delivery

Rationale for Software-Defined IoT Networking

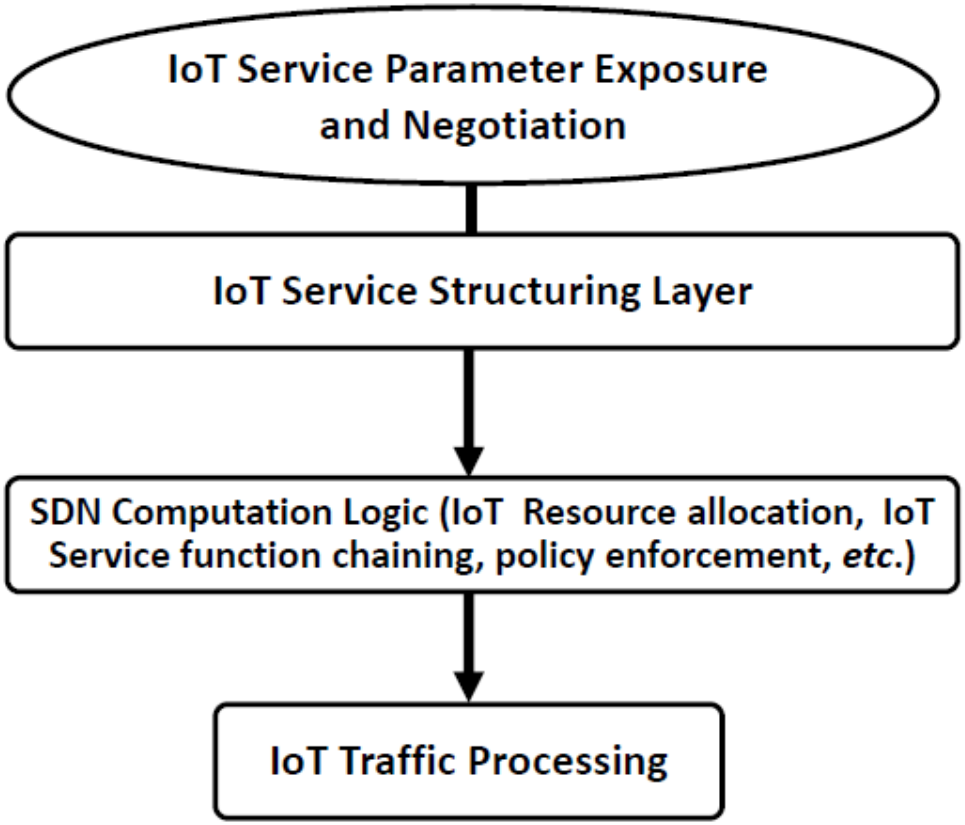
- Introduce *robust automation* in IoT service delivery for the sakes of cost optimization and improved service production times
  - Based upon a set of IoT service-specific policies
  - According to IoT customer/app requirements, possibly yielding a dynamic negotiation of IoT service parameters
- Use *dynamic resource allocation and policy enforcement schemes*
  - Likely based upon the use of various protocols and tools, given the broad heterogeneity of IoT technologies
- Activate *feedback mechanisms* to assess efficiency of IoT service delivery procedure
  - Verify that what has been delivered complies with what has been negotiated

SDN Framework



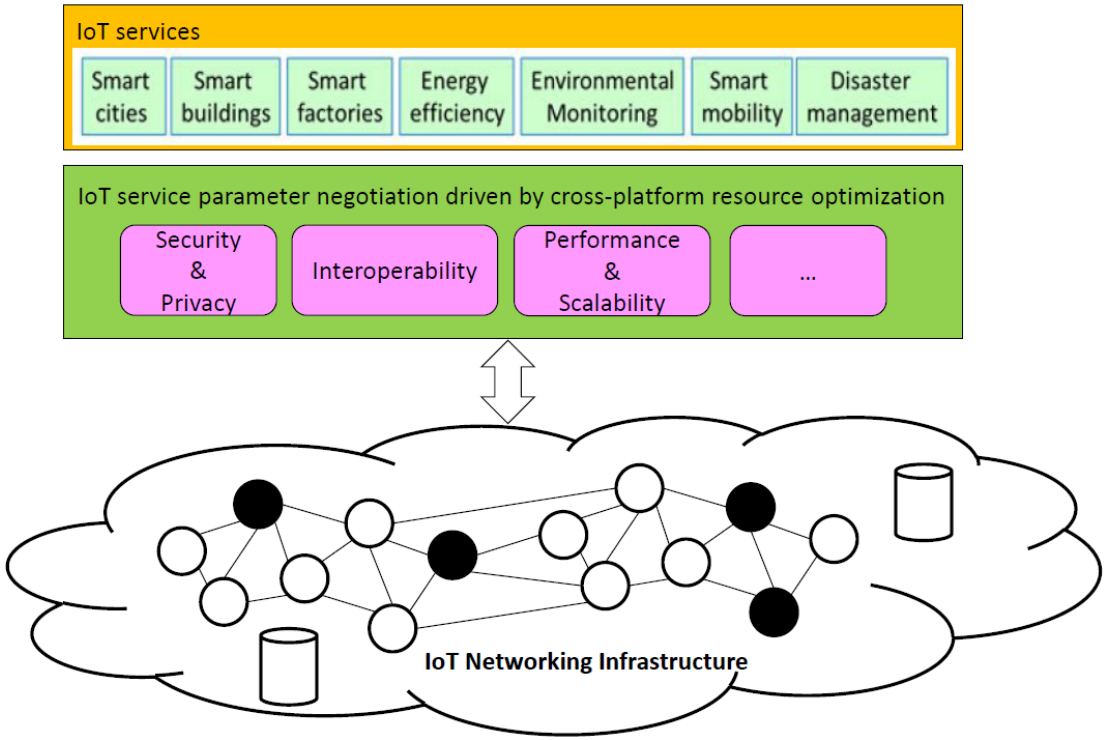
IoT Service Production Chain

- Outcomes of IoT service parameter negotiation feed SDN computation logic
  - Along with other inputs, like network-originated notifications and available resources
- IoT service is structured accordingly
  - Based upon abstract IoT service components depicted in (service-inferred) data models
- SDN computation logic then allocates IoT resources (network, storage, CPU)
  - Forwards policy decisions and configuration information to participating devices



IoT Service Parameter Exposure and Negotiation

- Multi-clause IoT service parameter template to
  - Accommodate specifics of a large variety of IoT services
  - Facilitate multiparty-operated resource integration
    - *E.g.*, cross-platform cooperation
- Clauses are manifold
  - Device geo-location
  - QoS requirements
  - Privacy requirements
  - Flow identification
  - *Etc*.



Negotiating IoT Service Parameters with CPP Template

- A [standard] (RFC 7297) , flexible format to list of (service-specific) various requirements for a connectivity service, including:
  - Service scope and flow characteristics
  - QoS and security parameters
  - Implicit or explicit service invocation means
- Facilitates dynamic service parameter negotiation
  - Outcomes to feed SDN computation logic for resource allocation and policy enforcement purposes

```
<CONNECTIVITY_PROVISIONING_DOCUMENT> ::=
                                    <Connectivity Provisioning Component> ...
<Connectivity Provisioning Component> ::=
                                    <CONNECTIVITY_PROVISIONING_PROFILE> ...
<CONNECTIVITY_PROVISIONING_PROFILE> ::=
                                    <Customer Nodes Map>
                                    <Scope>
                                    <QoS Guarantees>
                                    <Availability>
                                    <Capacity>
                                    <Traffic Isolation>
                                    <Conformance Traffic>
                                    <Flow Identification>
                                    <Overall Traffic Guarantees>
                                    <Routing and Forwarding>
                                    <Activation Means>
                                    <Invocation Means>
                                    <Notifications>
                                    <Optional Information Element> ...
<Customer Nodes Map> ::=  <Customer Node> ...
<Customer Node> ::=  <IDENTIFIER>
                     <LINK_IDENTIFIER>
                     <LOCALIZATION>
```

IoT Service Function Chaining (SFC)

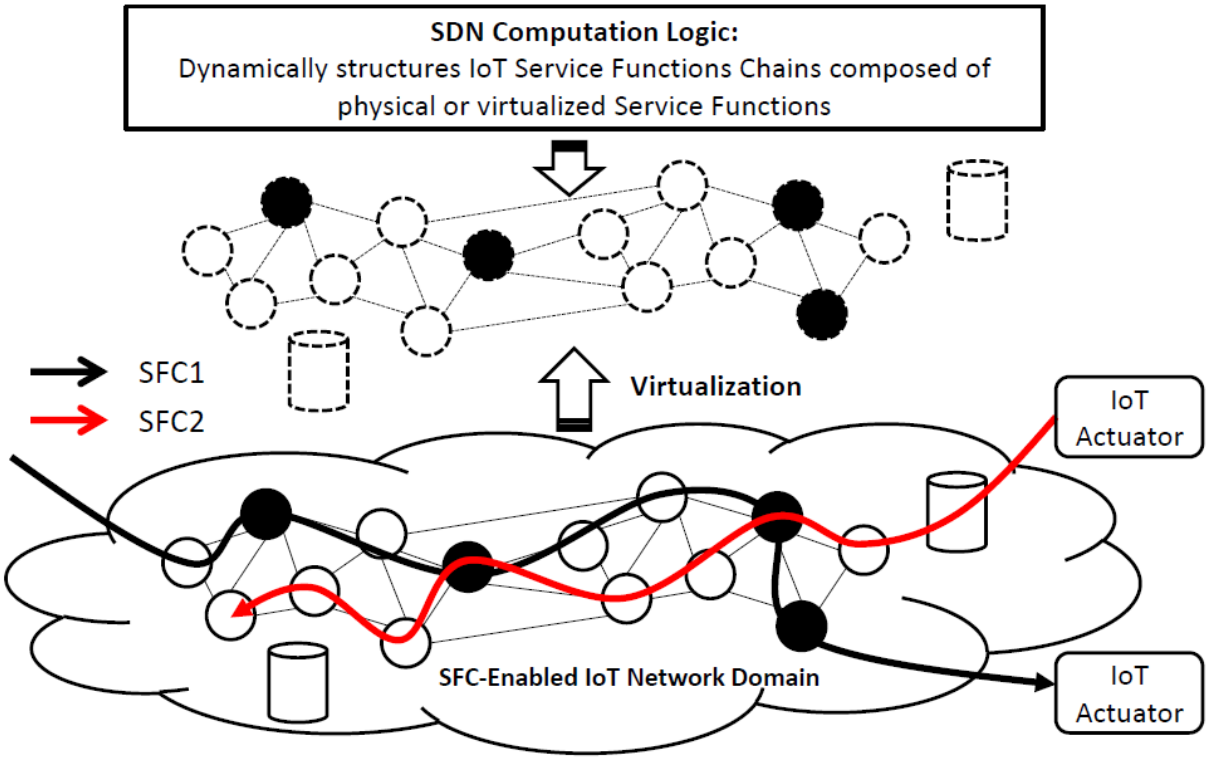
- Compute and establish IoT service-inferred forwarding paths
  - Thereby contributing to the optimization of overall service delivery and operation
- Master IoT Service Function (SF) chaining regardless of the underlying topology and routing policies
  - Yielding an IoT SF-based differentiated forwarding policy enforcement scheme
- Facilitate IoT SF operation while avoiding any major topology upgrade
  - Derive chronology of SF invocation according to the required service and associated parameters

IoT SFC Specifics

- Adaptation and mapping functions are required to accommodate a large variety of (proprietary) protocols and technologies
- A wide range of IoT-specific, tunable capabilities
  - Sleeping mode and duty cycle management (*e.g.*, Report Period settings)
  - MTU settings
  - IoT service-inferred routing metric settings (*e.g.*, ETX, latency, energy metric, header compression settings)
  - Objective Function (derived metric settings)
  - Security features
  - *Etc*.

IoT SFC Chaining Example

- Example chains reflect two different services (*e.g.*, networked humidity and water metering sensors) where:
  - SFC1 = {DPI; 6lo (WPAN) encapsulation; ETX setting; 6lo de-capsulation}
  - SFC2 = {DPI; 6lo (NFC) encapsulation; header compression; 6lo de-capsulation}

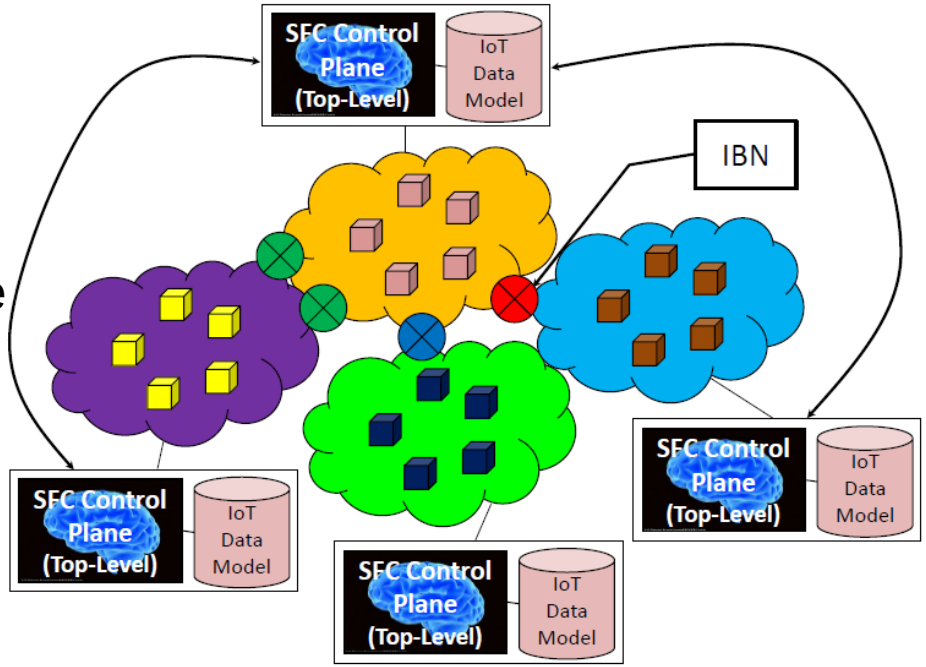


Design Considerations

- SFC control plane components can be centralized, distributed or a mix thereof, depending on:
  - Path computation capabilities
  - Number and nature/complexity of SFCs supported in the domain
  - Organization of SFC operation
  - SFC-inferred network taxonomy (access, backhaul, core)
- IoT Dimensioning figures may challenge single SFC control plane designs
  - Thereby introducing SFC control plane hierarchy

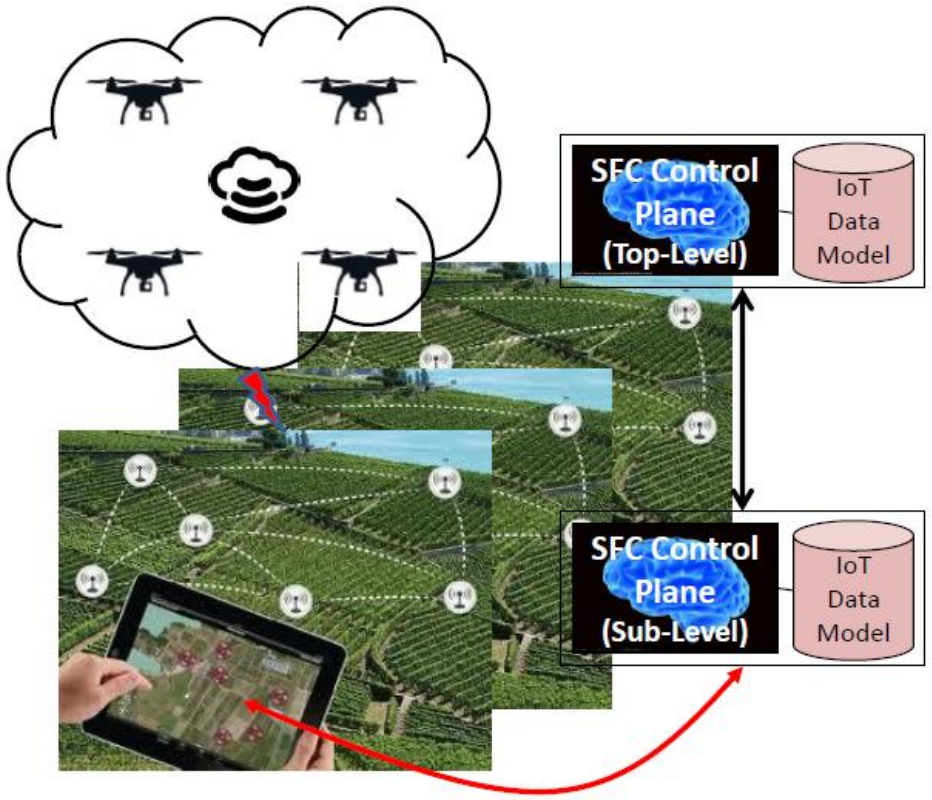
SFC Hierarchy for IoT

- Top-level SFC domain includes classifier and SF nodes
- Sub-domains are SFC domains too
  - Packets that enter a sub-domain already carry SFC instructions, unlike those that enter the top-level domain
  - Sub-domains interface with upper domain by means of Internal Boundary Nodes
- IBN nodes strip SFC encapsulation that pertains to upper domain
  - Then apply traffic classification rules specific of the sub-domain according to decisions made by sub-domain PDP



Drone-Monitored Multi-Field Irrigation Service

- Top-level SFC domain manages drone fleet and subsequent drone communication, *e.g.*,:
  - Controller-drone exchanges (*e.g.* where drones act as actuators for a set of fields)
  - Optimized Link State Routing (OLSR) routing function
- Field-specific networked humidity sensors compose sub-level SFC domain
  - 6lo encapsulation, Static context Header Compression (SCHC) function, *etc*.
- Optimizes flow management and traffic forwarding efficiency



Implications

- Decisions made by sub-domain control planes need to be consistent with those made by the top-level SFC control plane
  - IBN nodes are configured and seen as a single SF by the top-level IoT SFC control plane
- IBN nodes maintain states for flows that enter sub-domains
  - No need for sub-level SFC control planes to alter SFPs whenever upper level SFPs are changed
- Glue between upper and lower domains is the SFC metadata

Conclusion

- SDN and SFC techniques facilitate the automated delivery and operation of IoT services
  - But they assume a complex, smart combination of protocols, algorithms and (standard) data models
  - IoT service design for agriculture may also suggest combination of various techs (*e.g.*, WSNs connected to networked unmanned flying devices)
- Signaling traffic may affect expected overall performance, flexibility and scalability
  - Given typical IoT service dimensioning figures
  - Suggests considerations on hierarchical SDN and SFC designs
    - Service- or geographically-driven
- Cross-platform IoT infrastructures with a bit of virtualization are privileged designs
  - Improved mastering of resource usage

Thank You!