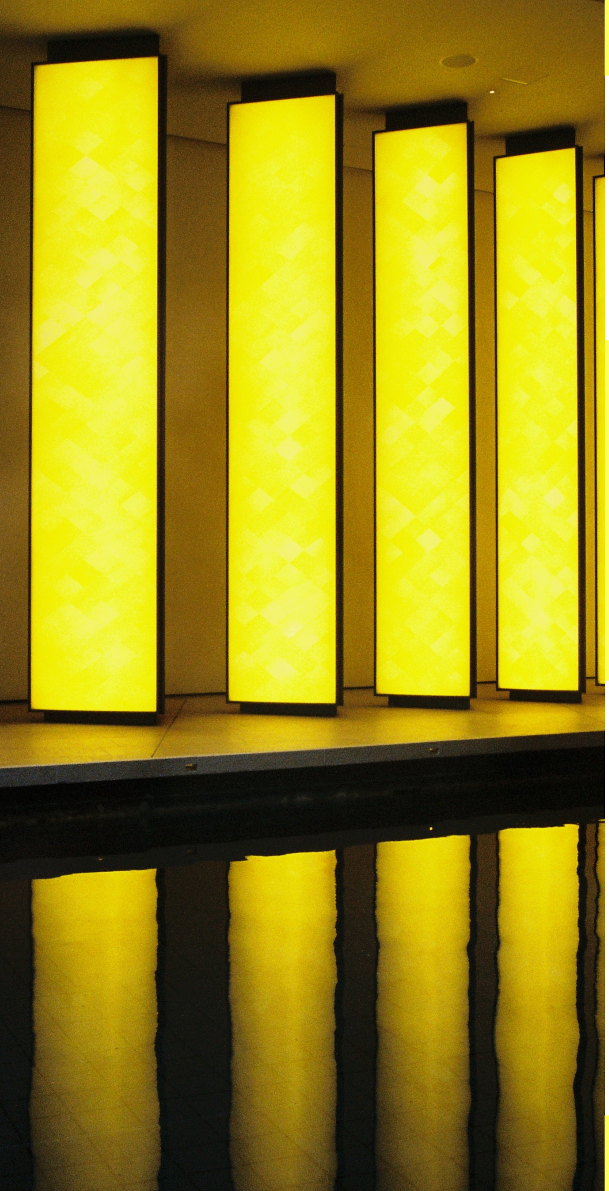


Distribution of Components in Event-Driven Sensor Networks

Final project presentation

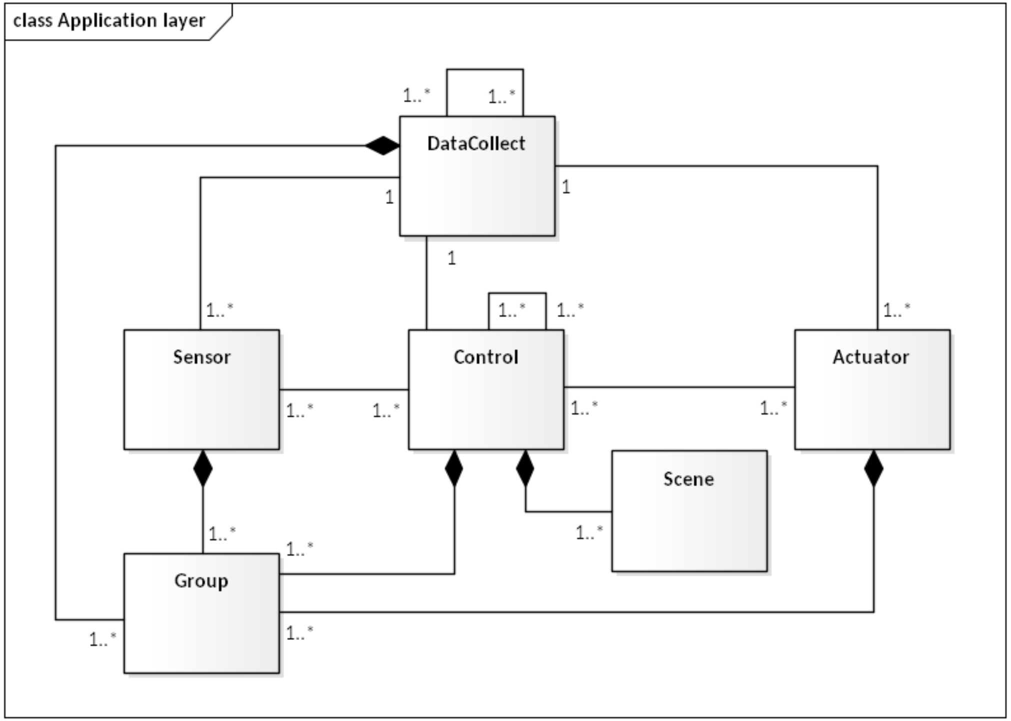
Bc. Miroslav Hájek
Software Architecture

Topics outline

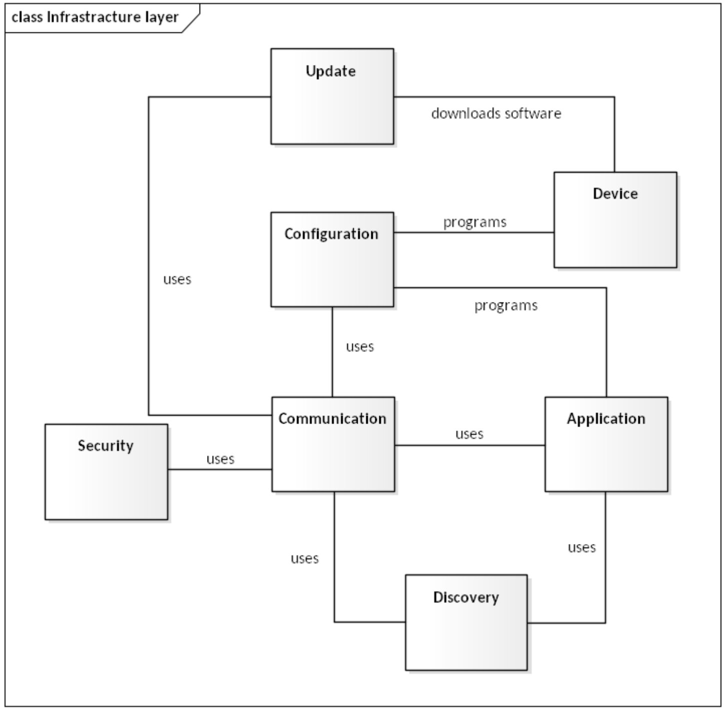
- **OpenAIS Object model overview**
- Demo application based on OpenAIS
  - Structure
  - API Interface
  - Rust implementation
  - Simulation – Light switch

OpenAIS – Object model

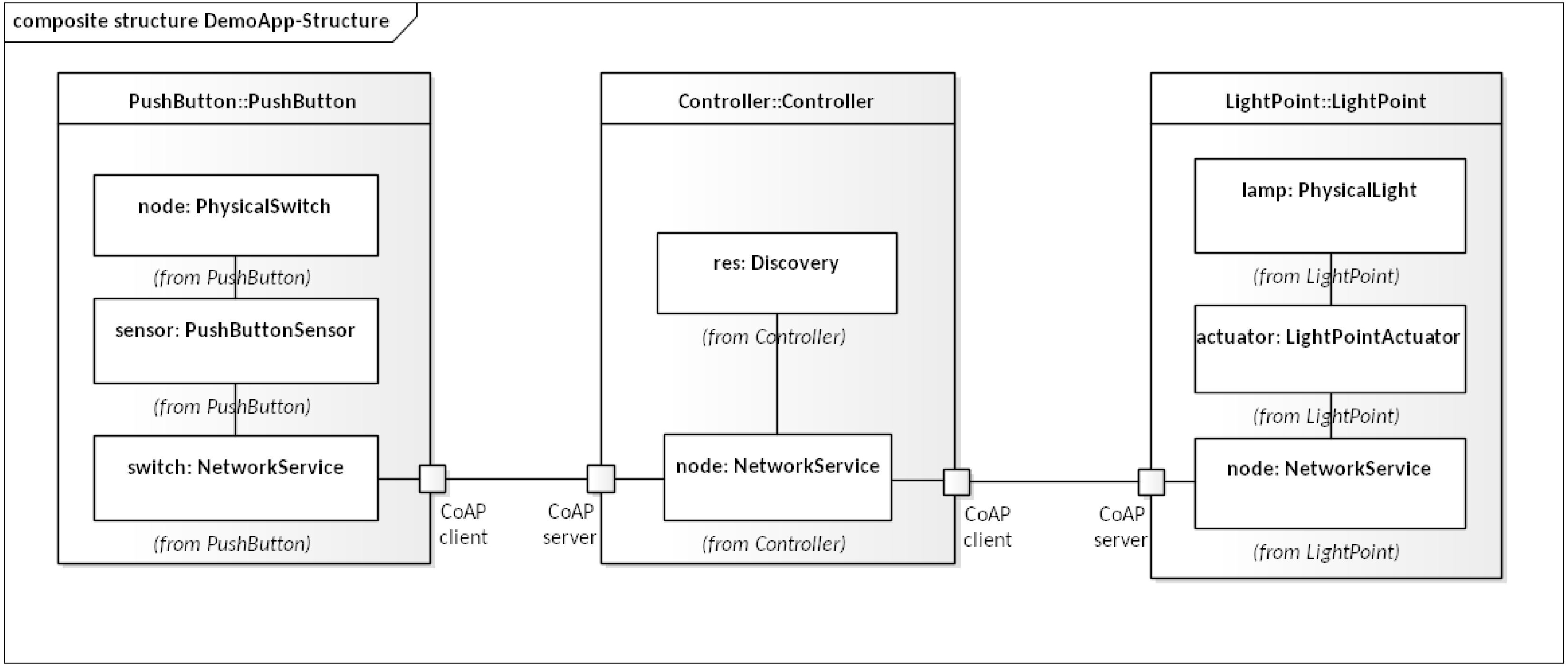
Application layer



Infrastructure layer

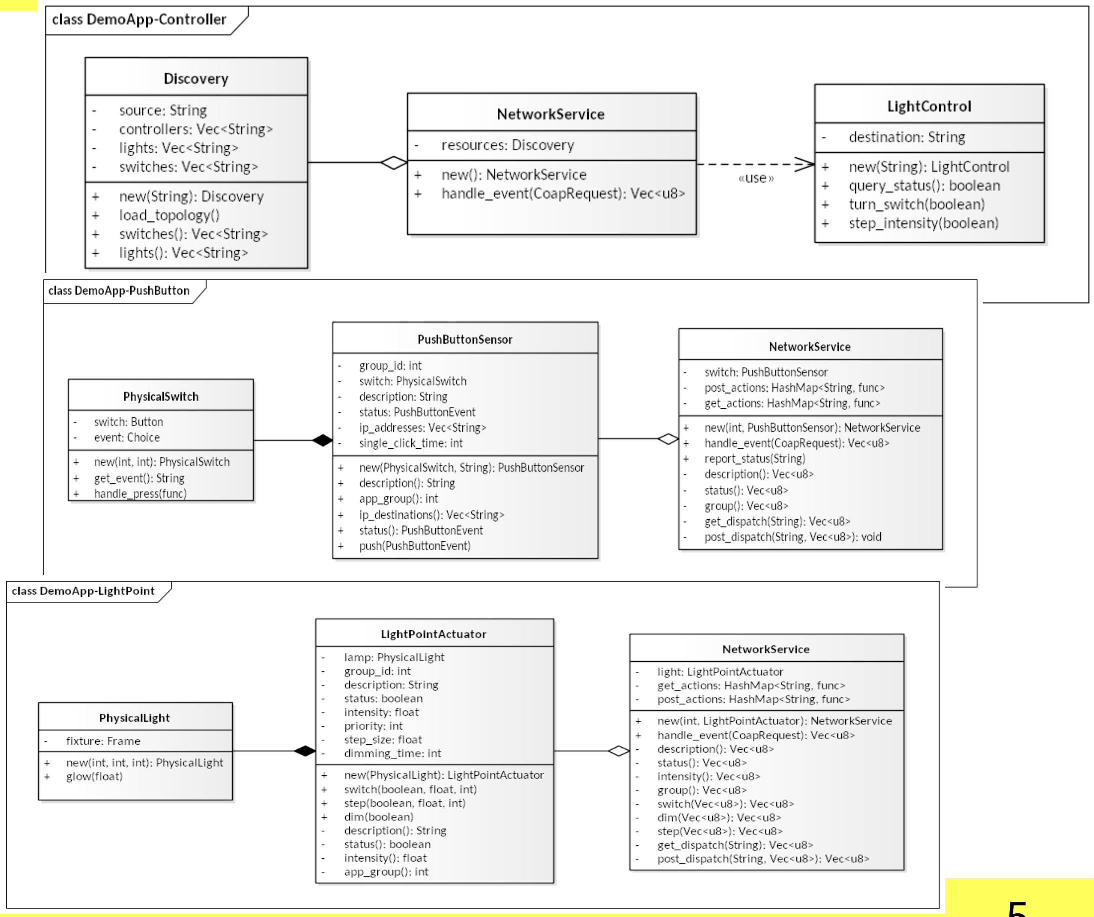


Distributed app based on OpenAIS



Class diagrams of services

- Sensor & Actuator: 3 layers
  - **Device** *(GUI, HW)*
  - **Configuration** *(PhysicalSwitch)*
  - **Communication** *(NetworkService)*
  - **Sensor / Actuator; + Group** *(PushButtonSensor)*
- Controller
  - **Discovery**
  - **Communication** *(NetworkService)*
  - **Control + Group** (LightControl)



class DemoApp-Controller

```
Discovery
-   source: String
-   controllers: Vec<String>
-   lights: Vec<String>
-   switches: Vec<String>
+   new(String): Discovery
+   load_topology()
+   switches(): Vec<String>
+   lights(): Vec<String>
```

```
NetworkService
-   resources: Discovery
+   new(): NetworkService
+   handle_event(CoapRequest): Vec<u8>
```

```
LightControl
-   destination: String
+   new(String): LightControl
+   query_status(): boolean
+   turn_switch(boolean)
+   step_intensity(boolean)
```

class DemoApp-PushButton

```
PhysicalSwitch
-   switch: Button
-   event: Choice
+   new(int, int): PhysicalSwitch
+   get_event(): String
+   handle_press(func)
```

```
PushButtonSensor
-   group_id: int
-   switch: PhysicalSwitch
-   description: String
-   status: PushButtonEvent
-   ip_addresses: Vec<String>
-   single_click_time: int
+   new(PhysicalSwitch, String): PushButtonSensor
+   description(): String
+   app_group(): int
+   ip_destinations(): Vec<String>
+   status(): PushButtonEvent
+   push(PushButtonEvent)
```

```
NetworkService
-   switch: PushButtonSensor
-   post_actions: HashMap<String, func>
-   get_actions: HashMap<String, func>
+   new(int, PushButtonSensor): NetworkService
+   handle_event(CoapRequest): Vec<u8>
+   report_status(String)
-   description(): Vec<u8>
-   status(): Vec<u8>
-   group(): Vec<u8>
-   get_dispatch(String): Vec<u8>
-   post_dispatch(String, Vec<u8>): void
```

class DemoApp-LightPoint

```
PhysicalLight
-   fixture: Frame
+   new(int, int, int): PhysicalLight
+   glow(float)
```

```
LightPointActuator
-   lamp: PhysicalLight
-   group_id: int
-   description: String
-   status: boolean
-   intensity: float
-   priority: int
-   step_size: float
-   dimming_time: int
+   new(PhysicalLight): LightPointActuator
+   switch(boolean, float, int)
+   step(boolean, float, int)
+   dim(boolean)
-   description(): String
-   status(): boolean
-   intensity(): float
-   app_group(): int
```

```
NetworkService
-   light: LightPointActuator
-   get_actions: HashMap<String, func>
-   post_actions: HashMap<String, func>
+   new(int, LightPointActuator): NetworkService
+   handle_event(CoapRequest): Vec<u8>
-   description(): Vec<u8>
-   status(): Vec<u8>
-   intensity(): Vec<u8>
-   group(): Vec<u8>
-   switch(Vec<u8>): Vec<u8>
-   dim(Vec<u8>): Vec<u8>
-   step(Vec<u8>): Vec<u8>
-   get_dispatch(String): Vec<u8>
-   post_dispatch(String, Vec<u8>): Vec<u8>
```

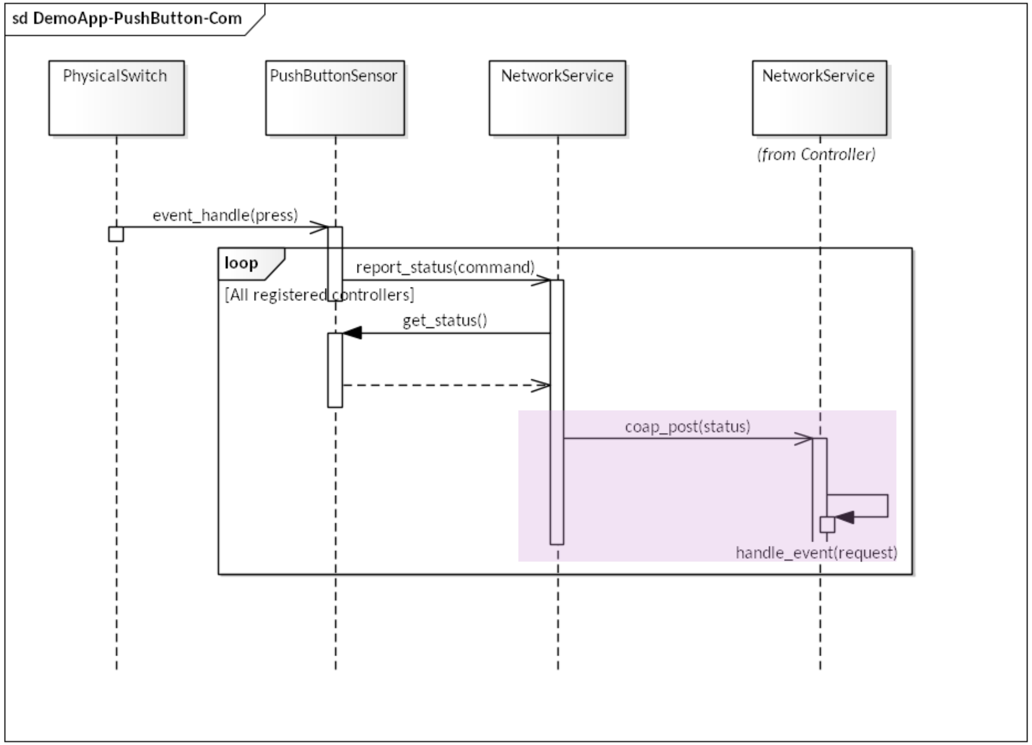
CoAP API Interface

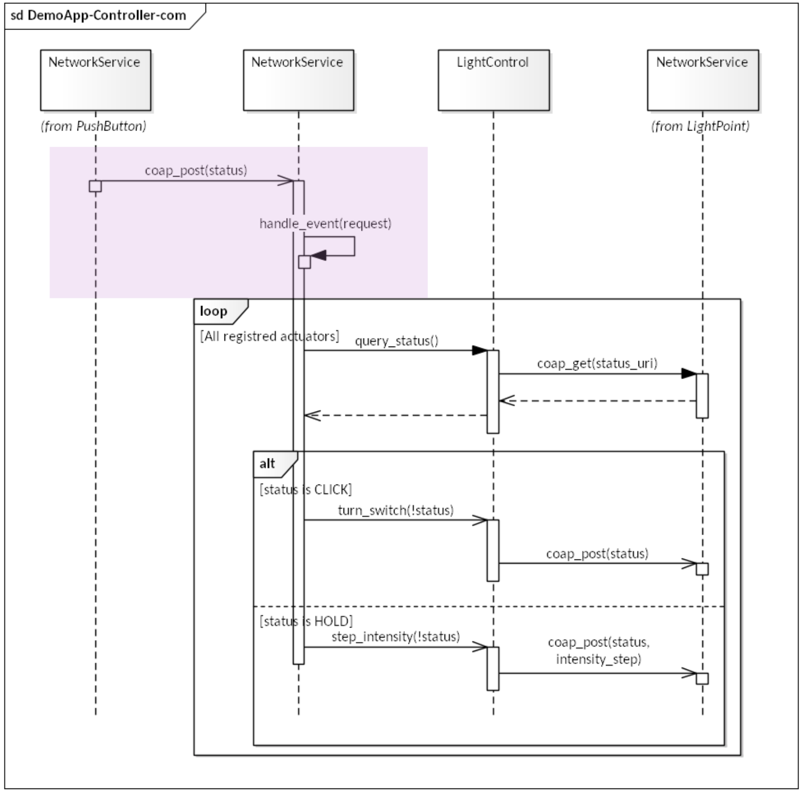
IDATA, ICONFIG, ICONTROL

- **4001 "oA Logical Light-Point Actuator"**
  - 901 "Documentary Description" string get
  - 100 "Target ON/OFF" Boolean get
  - 101 "Target Intensity" float get
  - 903 "Application Group ID" uint16 get
  - 117 "Switch" structure post
  - 118 "Dim" boolean post
  - 120 "Step" structure post

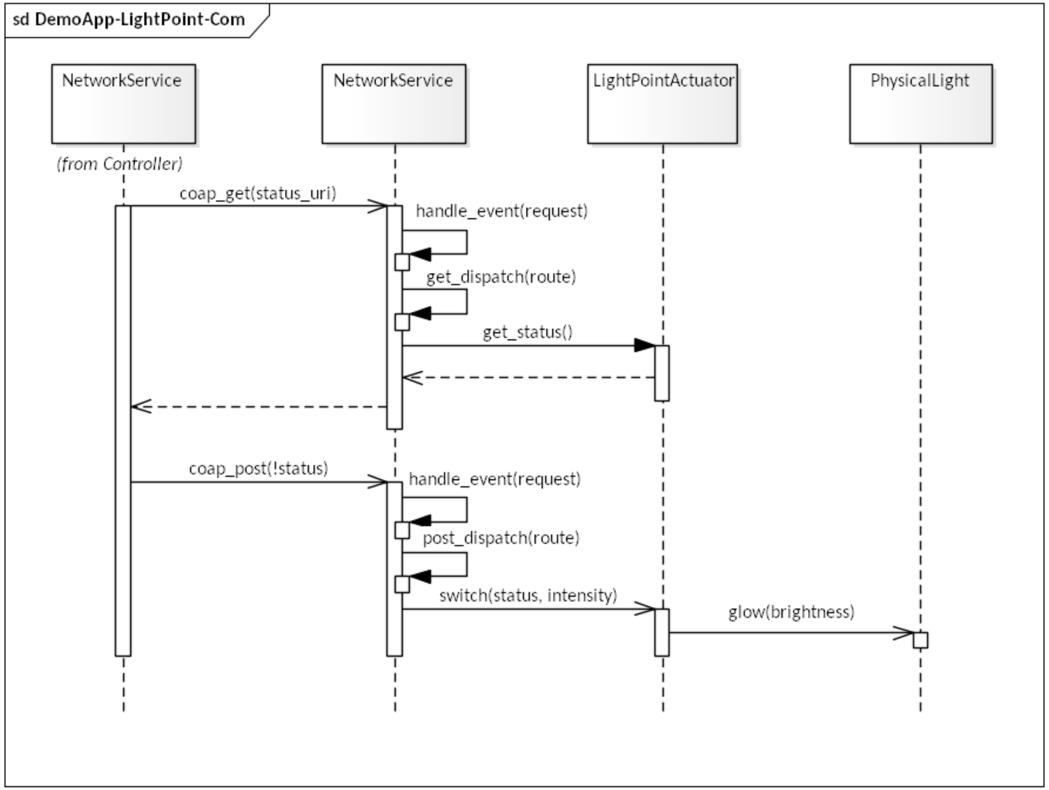
- **4002 "oA Logical Push-Button Sensor"**
  - 901 "Documentary Description" string get
  - 903 "Application Group ID" uint16 get
  - 202 "Push-Button Event Value" enumeration get

Scenario - Turn on / Dim light *(1)*





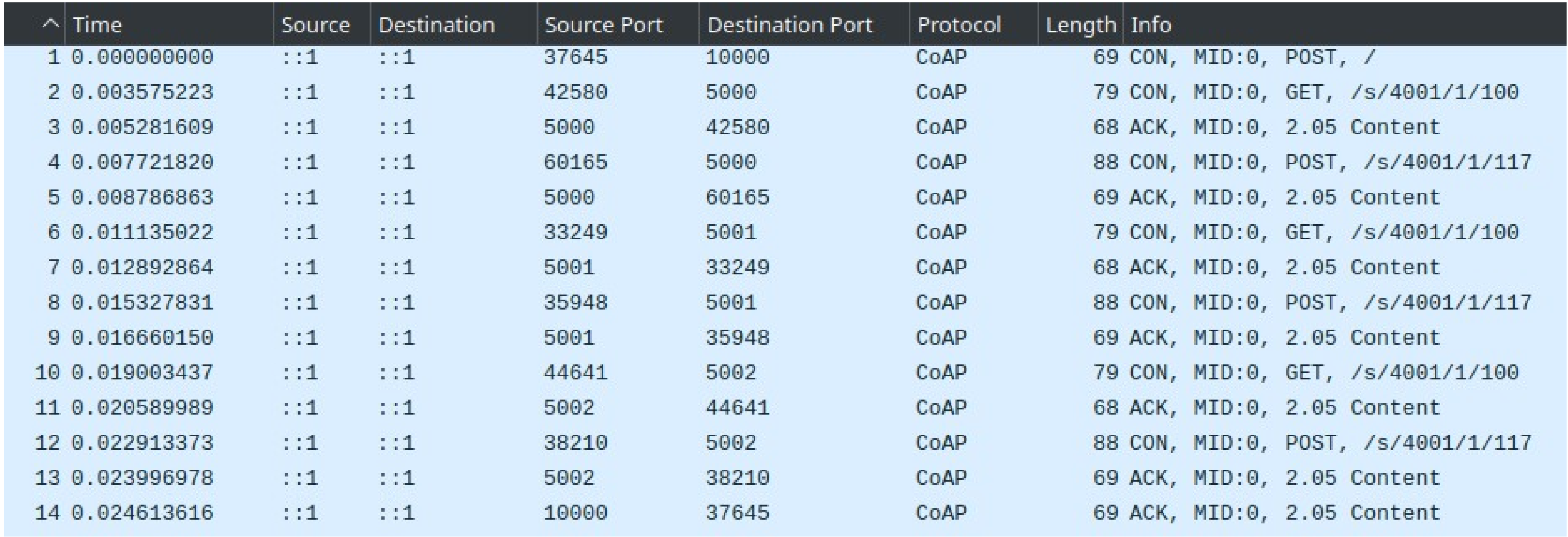
Scenario - Turn on light *(2)*



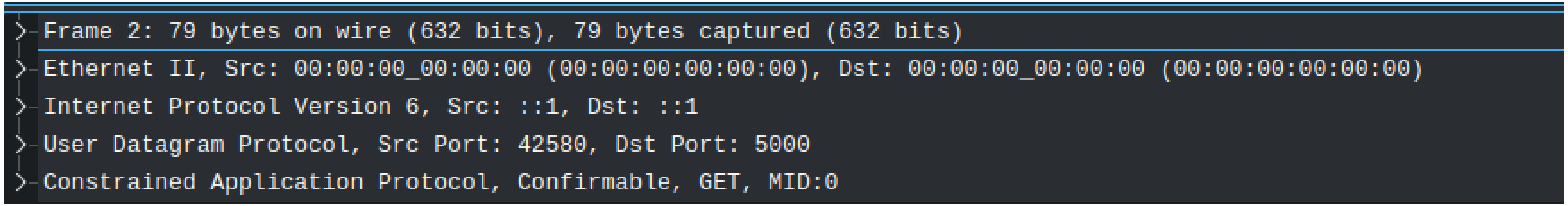
Changes for various types of deployments

- Sensor
  - List of controllers
- Controller
  - List of actuators

Communication - Network traces



	Time	Source	Destination	Source Port	Destination Port	Protocol	Length	Info
1	0.000000000	::1	::1	37645	10000	CoAP	69	CON, MID:0, POST, /
2	0.003575223	::1	::1	42580	5000	CoAP	79	CON, MID:0, GET, /s/4001/1/100
3	0.005281609	::1	::1	5000	42580	CoAP	68	ACK, MID:0, 2.05 Content
4	0.007721820	::1	::1	60165	5000	CoAP	88	CON, MID:0, POST, /s/4001/1/117
5	0.008786863	::1	::1	5000	60165	CoAP	69	ACK, MID:0, 2.05 Content
6	0.011135022	::1	::1	33249	5001	CoAP	79	CON, MID:0, GET, /s/4001/1/100
7	0.012892864	::1	::1	5001	33249	CoAP	68	ACK, MID:0, 2.05 Content
8	0.015327831	::1	::1	35948	5001	CoAP	88	CON, MID:0, POST, /s/4001/1/117
9	0.016660150	::1	::1	5001	35948	CoAP	69	ACK, MID:0, 2.05 Content
10	0.019003437	::1	::1	44641	5002	CoAP	79	CON, MID:0, GET, /s/4001/1/100
11	0.020589989	::1	::1	5002	44641	CoAP	68	ACK, MID:0, 2.05 Content
12	0.022913373	::1	::1	38210	5002	CoAP	88	CON, MID:0, POST, /s/4001/1/117
13	0.023996978	::1	::1	5002	38210	CoAP	69	ACK, MID:0, 2.05 Content
14	0.024613616	::1	::1	10000	37645	CoAP	69	ACK, MID:0, 2.05 Content



```
> Frame 2: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 6, Src: ::1, Dst: ::1
> User Datagram Protocol, Src Port: 42580, Dst Port: 5000
> Constrained Application Protocol, Confirmable, GET, MID:0
```

- **Topology.yaml (Discovery)**

```
controllers:
  - "[::]:10000"
lights:
  - "[::]:5000"
  - "[::]:5001"
  - "[::]:5002"
switches:
  - "[::]:4000"
```

Rust implementation

Initialize actuator components

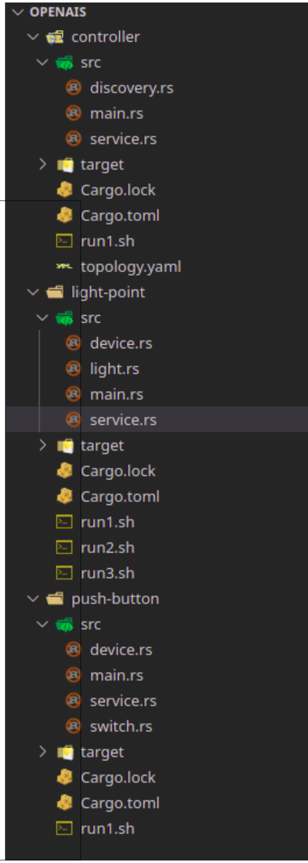
- `light-point/main.rs`

```
let COAP_ADDRESS =
     var("COAP_ADDRESS").unwrap();
let app = App::default();

let lamp = Arc::new(Mutex::new(
    device::PhysicalLight::new(500, 300, 50)
));
let actuator = light::LightPointActuator::new(
     Arc::clone(&lamp)
);
let node = Arc::new(Mutex::new(
    service::NetworkService::new(1, actuator)
));

thread::spawn(move || { … });  // CoAP server
app.run().unwrap();
```

Get light status via CoAP

- `light-point/service.rs`

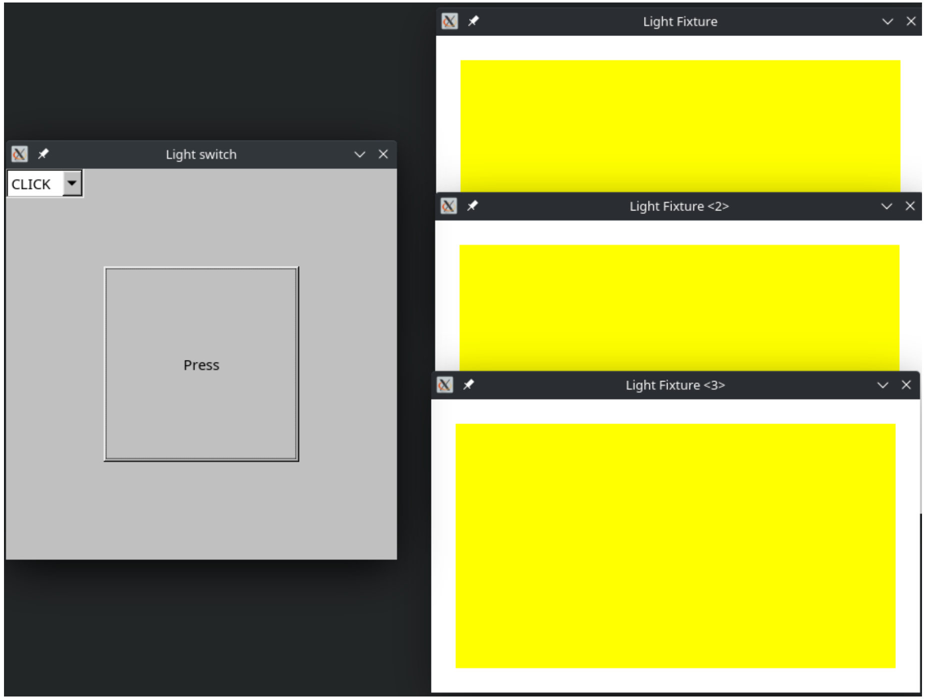
```
pub struct NetworkService {
  light: LightPointActuator,
  get_actions: HashMap<String, GetEndpoint>
}

get_actions: HashMap::from([
   (format!("s/4001/{}/100", id),
   NetworkService::status as GetEndpoint)])

impl NetworkService {
    fn status(&self) -> Vec<u8> {
        let cbor = CborBuilder::default()
        .encode_bool(self.light.status());
        return cbor.into_vec();
    }
}
```



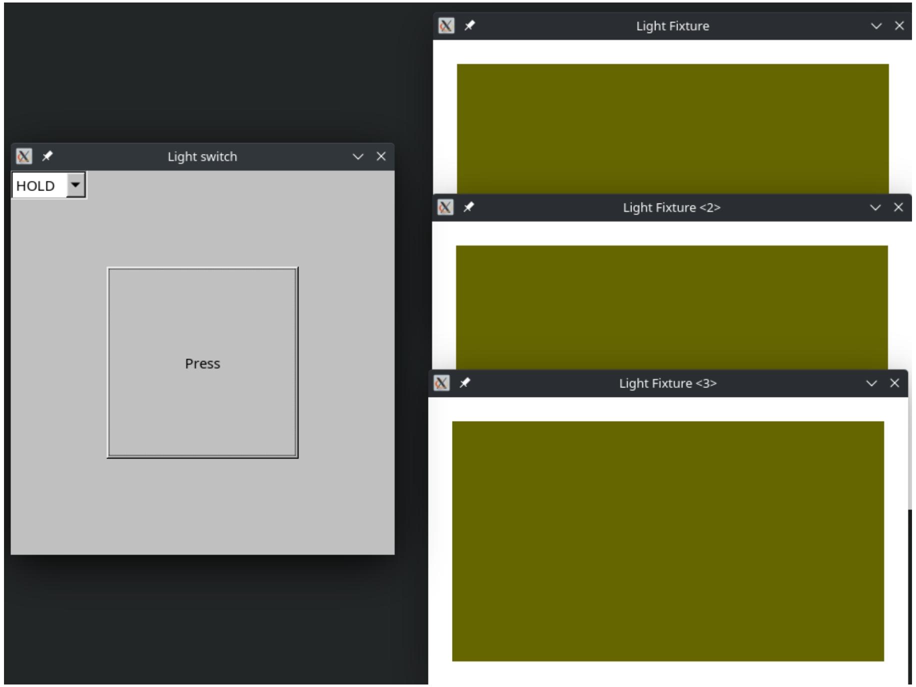
```
OPENAIS
  controller
    src
      discovery.rs
      main.rs
      service.rs
    target
    Cargo.lock
    Cargo.toml
    run1.sh
    topology.yaml
  light-point
    src
      device.rs
      light.rs
      main.rs
      service.rs
    target
    Cargo.lock
    Cargo.toml
    run1.sh
    run2.sh
    run3.sh
  push-button
    src
      device.rs
      main.rs
      service.rs
      switch.rs
    target
    Cargo.lock
    Cargo.toml
    run1.sh
```

Scenario simulation

CLICK event = Switch ON/OFF



HOLD event = Dim ±20%


Light switch
HOLD
Press
Light Fixture
Light Fixture <2>
Light Fixture <3>