IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

COMPACT REPRESENTATION OF DATA FROM A YANG DATA MODEL.

OPTIMIZING DATA MODELS FOR RESOURCE-CONSTRAINED IOT SYSTEMS

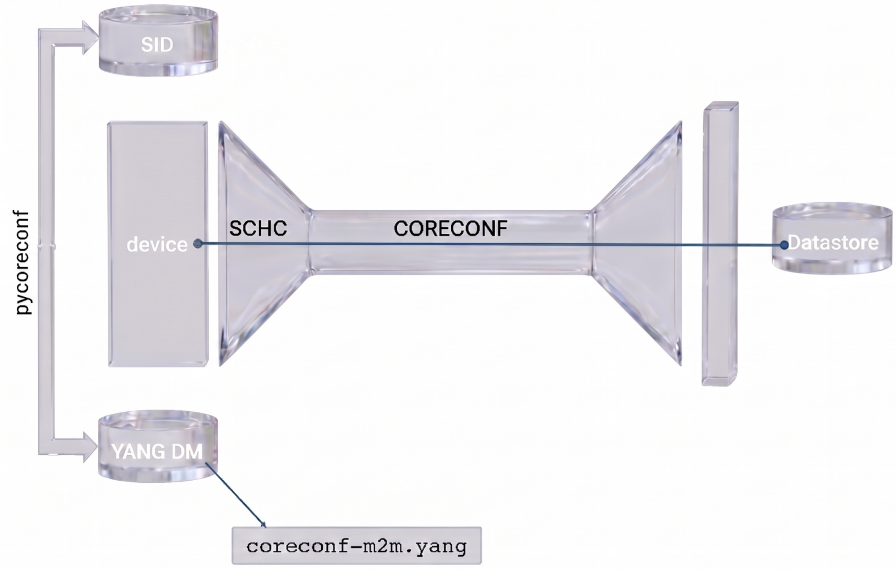
LAURENT TOUTAIN

QUESTIONS

1. HOW MANY BYTES FOR 10 TEMPERATURE MEASUREMENTS?
2. WITH INTEROPERABILITY?

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Global architecture



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

CORECONF

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

FORMULAIRE

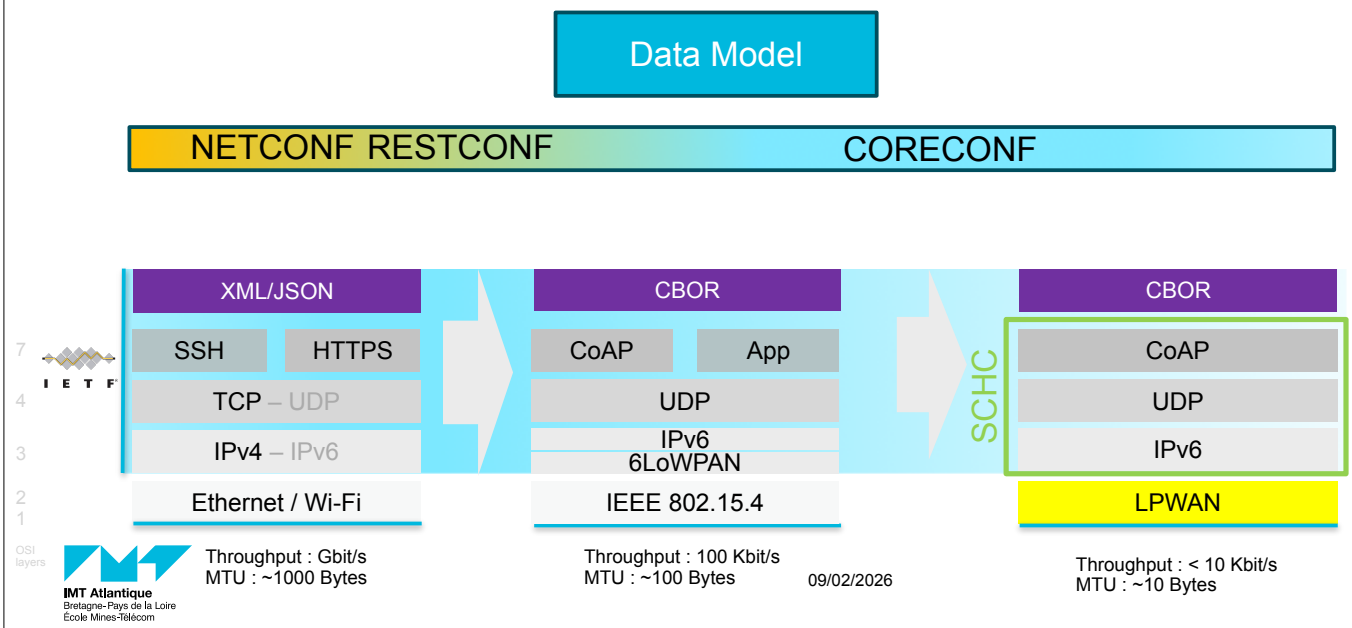
Form identifier

Predefined value

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

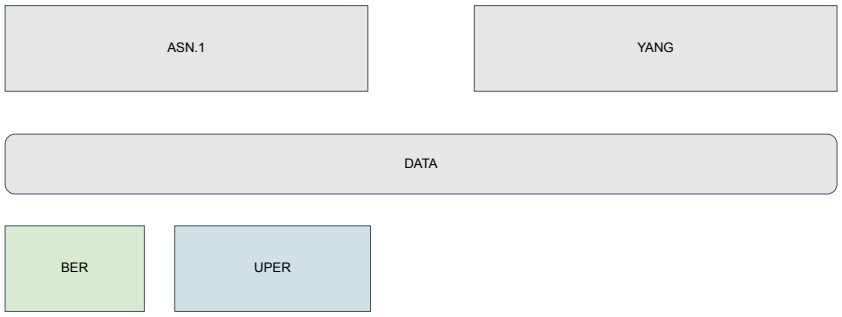
ARCHITECTURE

YANG



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

IMT ATLANTIQUE



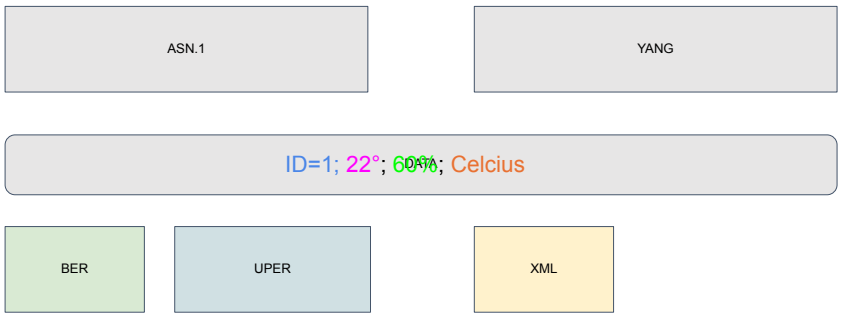

```
module sensor {
  namespace "urn:example:sensor";
  prefix "sen";

  container sensor-data {
    leaf sensor-id {
      type uint16; }
    leaf temperature {
      type int8 {
        range "-40..85"; } }
    leaf humidity {
      type uint8 {
        range "0..100"; } }
    leaf unit {
      type enumeration {
        enum celsius    { value 0; }
        enum fahrenheit { value 1; } } }
  }
}
```

YANG

Data modelling language

RFC 7950. Models config, state, RPCs — and increasingly application data such as sensor telemetry.

IMT ATLANTIQUE

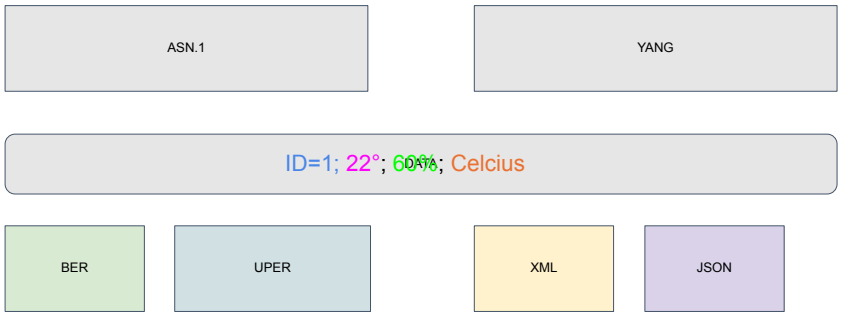



```
module sensor {
  namespace "urn:example:sensor";
  prefix "sen";

  container sensor-data {
    leaf sensor-id {
      type uint16; }
    leaf temperature {
      type int8 {
        range "-40..85"; } }
    leaf humidity {
      type uint8 {
        range "0..100"; } }
    leaf unit {
      type enumeration {
        enum celsius    { value 0; }
        enum fahrenheit { value 1; } } }
  }
}
```


```
<sensor-data xmlns="urn:example:sensor">
  <sensor-id>1</sensor-id>
  <temperature>22</temperature>
  <humidity>60</humidity>
  <unit>celsius</unit>
</sensor-data>
```

164 bytes

IMT ATLANTIQUE




```
module sensor {
  namespace "urn:example:sensor";
  prefix "sen";

  container sensor-data {
    leaf sensor-id {
      type uint16; }
    leaf temperature {
      type int8 {
        range "-40..85"; } }
    leaf humidity {
      type uint8 {
        range "0..100"; } }
    leaf unit {
      type enumeration {
        enum celsius    { value 0; }
        enum fahrenheit { value 1; } } }
  }
}
```

```
<sensor-data xmlns="urn:example:sensor">
  <sensor-id>1</sensor-id>
  <temperature>22</temperature>
  <humidity>60</humidity>
  <unit>celsius</unit>
</sensor-data>
```

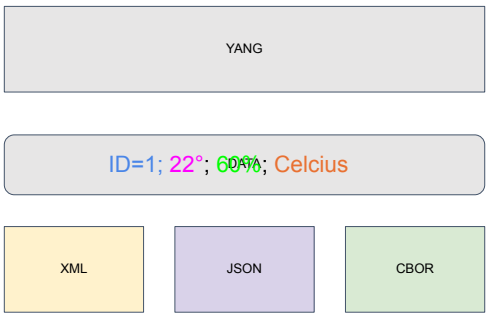
164 bytes

```
{
  "sensor:sensor-data": {
    "sensor-id": 1,
    "temperature": 22,
    "humidity": 60,
    "unit": "celsius"
  }
}
```

119 bytes

IMT ATLANTIQUE

```
3000000    module sensor
3000001    data /sensor:sensor-data
3000002    data /sensor:sensor-data/humidity
3000003    data /sensor:sensor-data/sensor-id
3000004    data /sensor:sensor-data/temperature
3000005    data /sensor:sensor-data/unit
```




```
module sensor {
  namespace "urn:example:sensor"
  prefix "sen";

  container sensor-data {
    leaf sensor-id {
      type uint16; }
    leaf temperature {
      type int8 {
        range "-40..85"; } }
    leaf humidity {
      type uint8 {
        range "0..100"; } }
    leaf unit {
      type enumeration {
        enum celsius    { value
        enum fahrenheit { value
  }
}
```

```
<sensor-data xmlns="urn:example:sensor">
  <sensor-id>1</sensor-id>
  <temperature>22</temperature>
  <humidity>60</humidity>
  <unit>celsius</unit>
</sensor-data>
```

164 bytes

```
{
  "sensor:sensor-data": {
    "sensor-id": 1,
    "temperature": 22,
    "humidity": 60,
    "unit": "celsius"
  }
}
```

119 bytes

```
A1
  1A 002DC6C1
  A4
    01 18 3C
    02 01
    03 16
    04 00
```

16 bytes

types

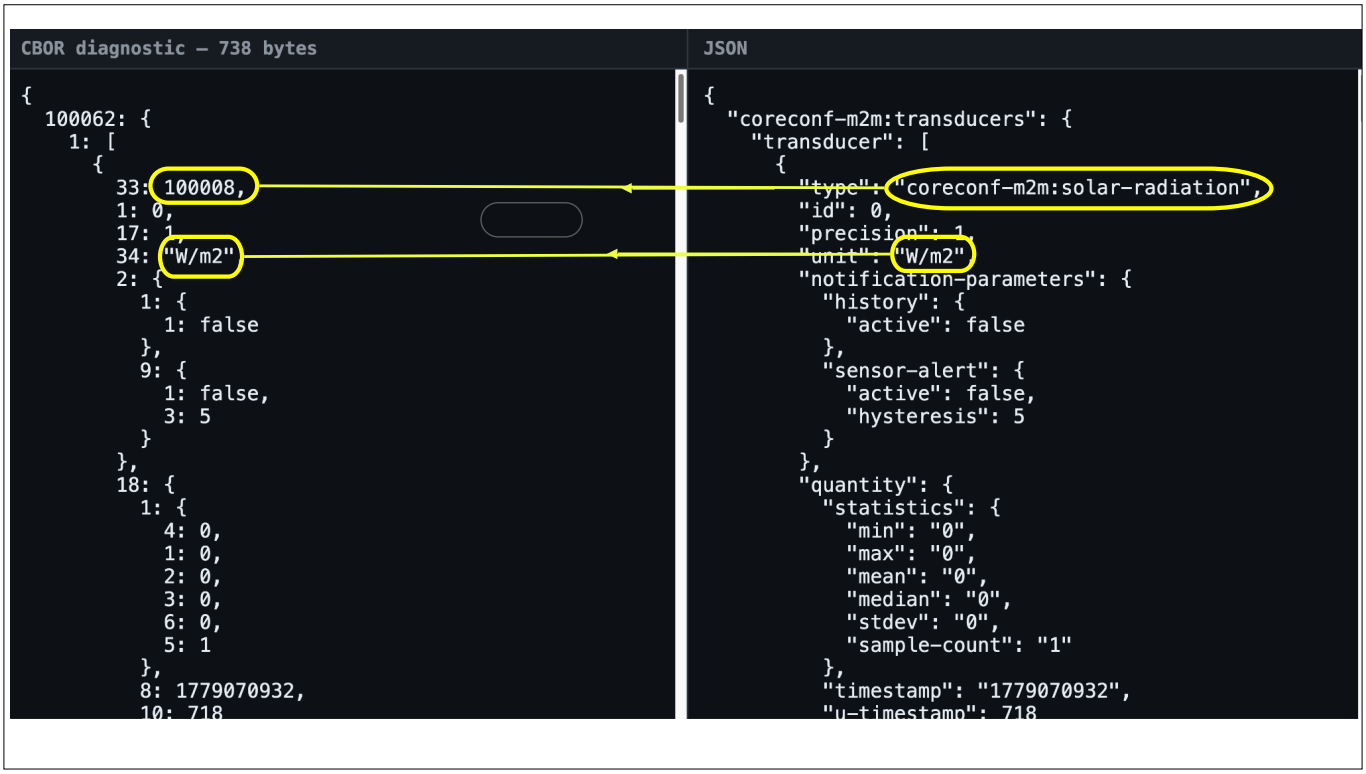
CBOR diagnostic — 738 bytes

```
{
  100062: {
    1: [
      {
        33: 100008,
        1: 0,
        17: 1,
        34: "W/m2",
        2: {
          1: {
            1: false
          },
          9: {
            1: false,
            3: 5
          }
        },
        18: {
          1: {
            4: 0,
            1: 0,
            2: 0,
            3: 0,
            6: 0,
            5: 1
          },
          8: 1779070932,
          10: 718
```

JSON

```
{
  "coreconf-m2m:transducers": {
    "transducer": [
      {
        "type": "coreconf-m2m:solar-radiation",
        "id": 0,
        "precision": 1,
        "unit": "W/m2",
        "notification-parameters": {
          "history": {
            "active": false
          },
          "sensor-alert": {
            "active": false,
            "hysteresis": 5
          }
        },
        "quantity": {
          "statistics": {
            "min": "0",
            "max": "0",
            "mean": "0",
            "median": "0",
            "stdev": "0",
            "sample-count": "1"
          },
          "timestamp": "1779070932",
          "u-timestamp": 718
```


CBOR diagnostic — 738 bytes
JSON
{
100062: {
1: [
{
33: 100008,
1: 0,
17: 1,
34: "W/m2",
2: {
1: {
1: false
},
9: {
1: false,
3: 5
}
},
18: {
1: {
4: 0,
1: 0,
2: 0,
3: 0,
6: 0,
5: 1
},
8: 1779070932,
10: 718
"coreconf-m2m:transducers": {
"transducer": [
{
"type": "coreconf-m2m:solar-radiation",
"id": 0,
"precision": 1,
"unit": "W/m2",
"notification-parameters": {
"history": {
"active": false
},
"sensor-alert": {
"active": false,
"hysteresis": 5
}
},
"quantity": {
"statistics": {
"min": "0",
"max": "0",
"mean": "0",
"median": "0",
"stdev": "0",
"sample-count": "1"
},
"timestamp": "1779070932",
"u-timestamp": 718


CBOR diagnostic — 738 bytes
JSON
{
100062: {
1: [
{
33: 100008,
1: 0,
17: 1,
34: "W/m2"
2: {
1: {
1: false
},
9: {
1: false,
3: 5
}
},
18: {
1: {
4: 0,
1: 0,
2: 0,
3: 0,
6: 0,
5: 1
},
8: 1779070932,
10: 718
"coreconf-m2m:transducers": {
"transducer": [
{
"type": "coreconf-m2m:solar-radiation",
"id": 0,
"precision": 1,
"unit": "W/m2"
"notification-parameters": {
"history": {
"active": false
},
"sensor-alert": {
"active": false,
"hysteresis": 5
}
},
"quantity": {
"statistics": {
"min": "0",
"max": "0",
"mean": "0",
"median": "0",
"stdev": "0",
"sample-count": "1"
},
"timestamp": "1779070932",
"u-timestamp": 718

Type

- CBOR structure = JSON structure
  - key: delta encoded or string following the hierarchy
  - value type is needed for leaf value conversion
- Include type into the sid file
  - convenient since sid ⇔ YANG name is present
- 4 types
  - YANG basic types (uint, int, float, string, ….)
  - identityref
  - enumeration
  - bits
- union

test sid-extension in pyang

```
%uvx --from git+https://github.com/ltn22/pyang@sid-extension
pyang -p ../pyang/modules/ietf/ --sid-generate-file=100000:400
--sid-extension coreconf-m2m@2026-03-29.yang
WARNING: Module 'ietf-geo-location' imported without revision,
using latest revision 2022-02-11
File coreconf-m2m@2026-03-29.sid created
Number of SIDs available : 400
Number of SIDs used : 98
```

coreconf-m2m@2026-03-29.sid


```
{
    "namespace": "data",
    "identifier": "/coreconf-m2m:characteristics",
    "status": "unstable",
    "sid": "100018"
},
```

```
{
   "namespace": "data",
   "identifier": "/coreconf-m2m:history/time-series/id",
   "status": "unstable",
   "sid": "100045",
   "type": "uint8"
},
```

```
{
   "namespace": "data",
   "identifier": "/coreconf-m2m:sensor-alert/target/type",
   "status": "unstable",
   "sid": "100058",
   "type": "identityref"
 },
```

```
{
   "namespace": "data",
   "identifier": "/coreconf-m2m:transducer
   "status": "unstable",
   "sid": "100090",
   "type": {
     "0": "source",
     "1": "receiver"
    }
},
```


Example

```
sid_path = "coreconf-m2m@2026-03-29.sid"
ccm = pycoreconf.CORECONFModel(sid_path)

cbor_data = ccm.encode_json(json_file)
decoded_json = ccm.decode(cbor_data)
```

see:
https://github.com/alex-fddz/pycoreconf/blob/main/samples/datastore/main.py

key_mapping

key_mapping

Used to navigate inside the data structure:

- When reaching a list, the leaf acting as a key must be known.
- `key_mapping` in the sid file adds this information:
  - JSON key indicates which nodes are YANG list.
  - JSON array gives the list of YANG nodes acting as a YANG key.
  - association the value in this YANG key with the values given in the request.

```
"key-mapping": {
    "100063": [
      100096,
      100064
    ],
    "100044": [
      100050,
      100045
    ],
    "100056": [
      100058,
      100057
    ]
  }
```

pycoreconf datastore, direct code manipulation:

```
ds = ccm.create_datastore_from_cbor(cbor_data)
transducer_keys = ds["/transducers/transducer"]
xpath_entry =
 "/transducers/transducer[type='coreconf-m2m:solar-radiation'][id='0']"
entry = ds[xpath_entry]
ds[xpath_entry+"quantity/statitics/sample_count”] += 1

new_xpath =
 "/transducers/transducer[type='coreconf-m2m:solar-radiation'][id='1']"
ds[new_xpath] = {'precision': 3}

for pred in ds.predicates("/transducers/transducer"):
   xpath_sc = f"/transducers/transducer{pred}/quantity/…/sample-count"
   ds[xpath_sc] += 1
```

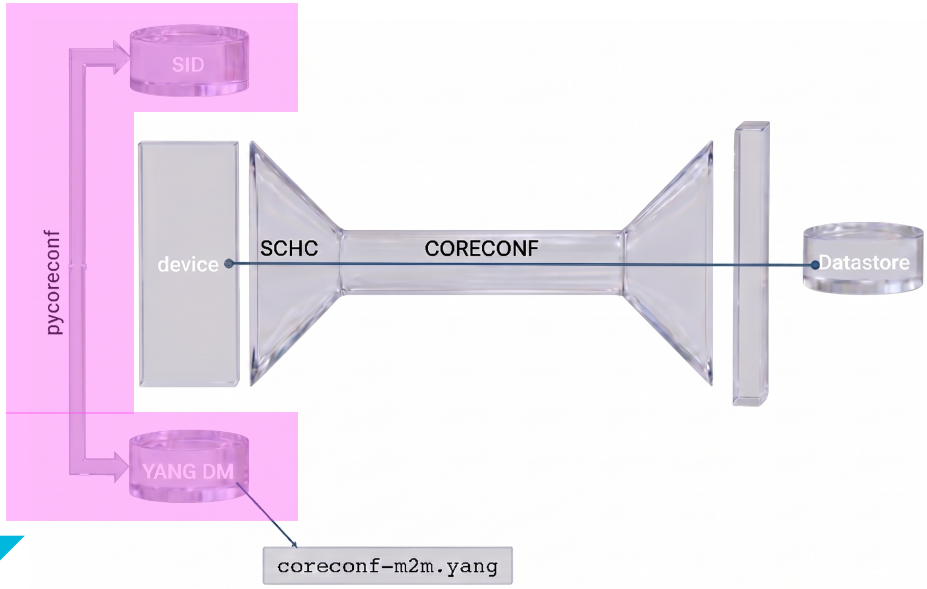
Interaction with CoAP

```
async def render_fetch(self, request):

   if not request.payload:
      return Message(payload=self.db.to_coreconf_cbor(), content_format=142
   item = request.payload

   if type(item) is int:
      leaf_sid = item
      keys = []
   else:
      leaf_sid, *keys = item

   raw = cbor.loads(self.db.to_coreconf_cbor())
   result = self.db.model.findSID(raw, sid=leaf_sid, keys=keys, depth=depth)
   if result is not None:
      return Message(payload=result.getvalue(), content_format=142)
```

Interaction with CoAP

```
async def render_ipatch(self, request):

    for key, value in cbor.loads(request.payload).items():
        xpath = f"/{MODULE}:" + self.db.datastore._create_xpath(key[0], key[1:]).lstrip("/")
        incoming = next(iter(json.loads(self.db.model.toJSON(cbor.dumps(value))).values()))
        existing = self.db.datastore[xpath]
        merged = {**existing, **incoming}

        self.db.datastore[xpath] = merged

    return Message(code=numbers.codes.Code.CHANGED)
```

Global architecture



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

CORECONF-M2M

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

CORECONF-M2M

Existing approaches don't fully fit constrained, peer-to-peer M2M exchanges over low-power links

THE GAP

- **SenML**

  Flexible serialization, but no strong typing, schema validation, or remote-operation support

- **LwM2M**

  Adds device management on CoAP/SenML, but periodic registration adds overhead on LPWANs

- **CORECONF today**

  Strong YANG typing + CBOR/SID compactness — but designed operator-to-device, not peer-to-peer M2M

THIS DRAFT'S USE CASES

Resource discovery

Transducers (sensors/actuators) advertise type, unit and precision

Simple query

Each transducer individually queried or set via FETCH / iPATCH

Statistical computation

Device computes mean, variance, min, max — resettable on demand

Alert notification

Threshold crossing (min/max) triggers an immediate notification

Time series

Samples collected and pushed once a count, size or duration limit is reached

CORECONF-M2M


```
module: coreconf-m2m
  +--ro state
  |  +--ro uptime?   uint64
  +--rw characteristics
  |  +--rw name?         string
  |  +--rw version?      string
  |  +--rw identifier?   string
  +--rw transducers
     +--rw transducer* [type id]
        +--rw type                    identityref
        +--rw id                      uint8
        +--rw unit?                   string
        +--rw precision?              uint8
        +--ro quantity
        |  +--ro value?               int64
        |  +--ro timestamp?           uint64
        |  +--ro u-timestamp?         uint32
        |  +--ro timestamp-source?    enumeration
        |  +--ro statistics
        |     +--ro min?              int64
        |     +--ro max?              int64
        |     +--ro mean?             int64
        |     +--ro median?           int64
        |     +--ro stdev?            uint64
        |     +--ro sample-count?     uint64
        +--rw notification-parameters
        |  +--rw history
        |  |  +--ro active?           boolean
        |  |  +--rw step?             uint32
        |  |  +--rw precision?        uint8
        |  |  +--rw max-samples?      uint32
        |  |  +--rw time-period?      uint32
        |  |  +--rw encoding?         encoding-type
        |  |  +--rw max-payload?      uint32
        |  |  +--rw check-interval?   uint16
        |  +--rw sensor-alert
        |     +--ro active?           boolean
        |     +--rw t-min?            int32
        |     +--rw t-max?            int32
        |     +--rw hysteresis?       uint8
        |     +--rw dampening?        uint32
        |     +--rw check-interval?   uint16
        +---x reset-stats
notifications:
  +---n history
  |  +--ro last?          boolean
  |  +--ro time-series* [type id]
  |     +--ro type        identityref
  |     +--ro id          uint8
  |     +--ro values*     int64
  |     +--ro internal
  |        +--ro last-update?     uint64
  |        +--ro start-time?      uint64
  |        +--ro messages-sent?   uint64
  +---n sensor-alert
     +--ro target* [type id]
        +--ro type     identityref
        +--ro id       uint8
        +--ro value?   int64
```

On the wire: compact CoAP traffic

Two CoAP methods, two content-formats, and SID-based CBOR keep every exchange SCHC-friendly

FETCH *replaces GET*

Body carries the exact SIDs to retrieve — precise, bandwidth-efficient reads, also used with Observe to subscribe to notifications.

iPATCH *replaces PUT / POST*

Partial updates of quantities and notification parameters; an empty value clears a node, so DELETE is unnecessary.

141 application/yang-fetch+cbor — FETCH request bodies **142** application/yang-data+cbor;id=sid — all responses & iPATCH payloads

Non-Confirmable

FETCH / iPATCH sent as CoAP NON — fewer fixed header patterns means tighter SCHC compression on constrained links.

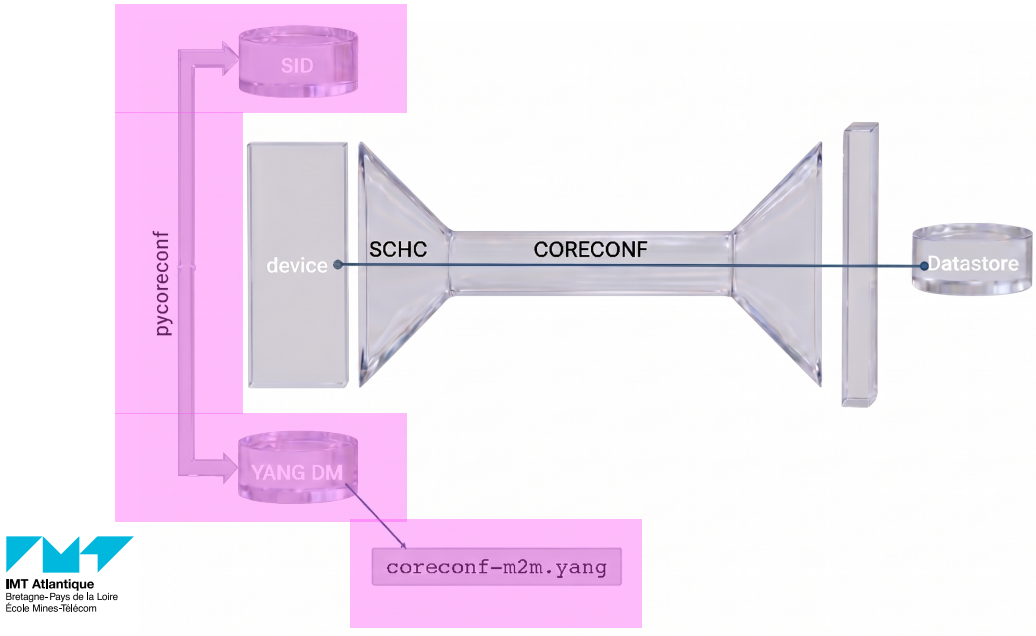
DTLS / OSCORE

Security MUST be enforced at the CoAP layer; trust matters even more without a central manager in pure M2M setups.

SID range 100000–399

Example allocation for this draft's module; official IANA values to be assigned before publication.

Global architecture

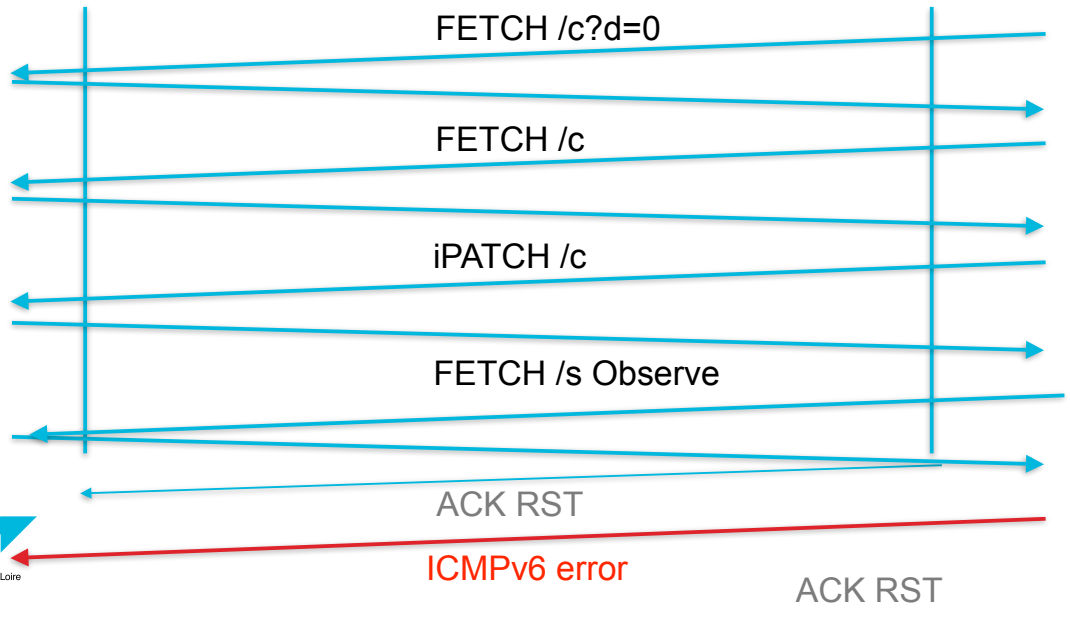


IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

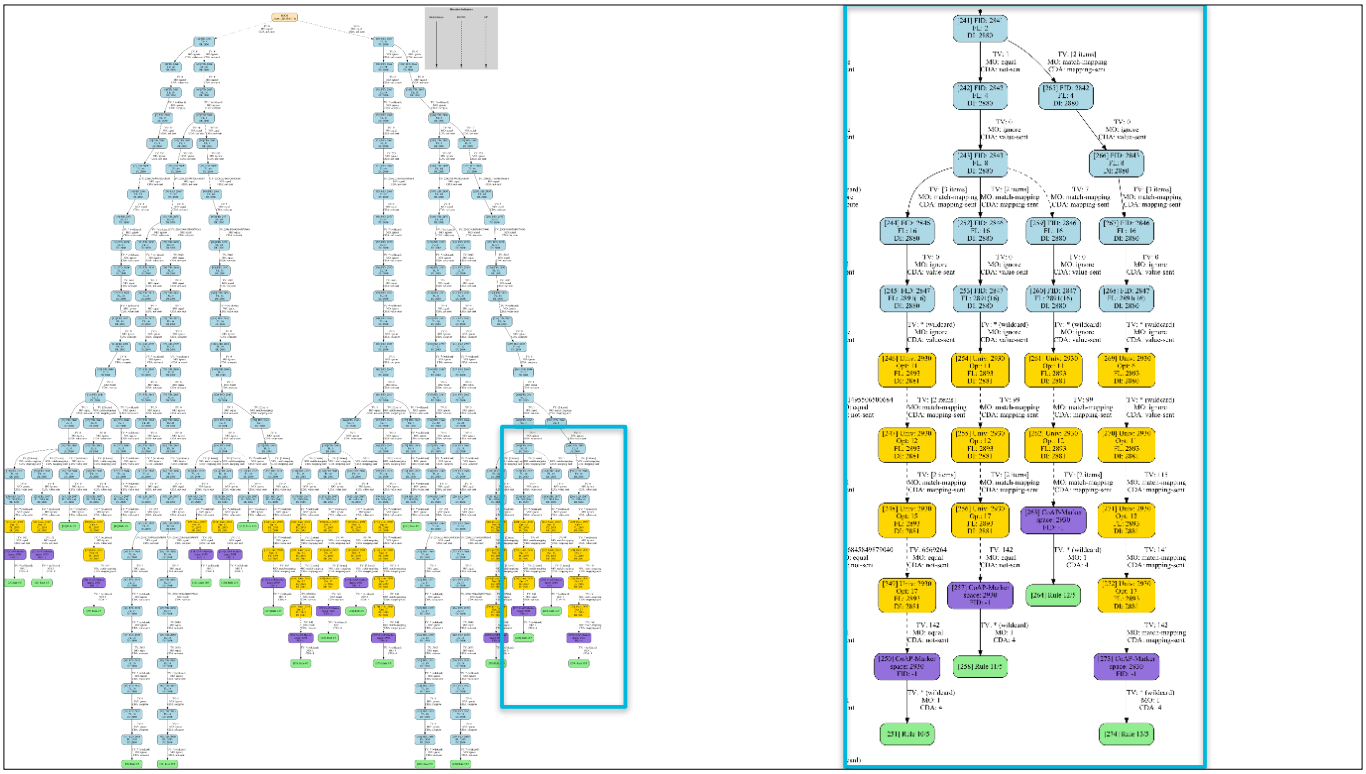
SCHC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

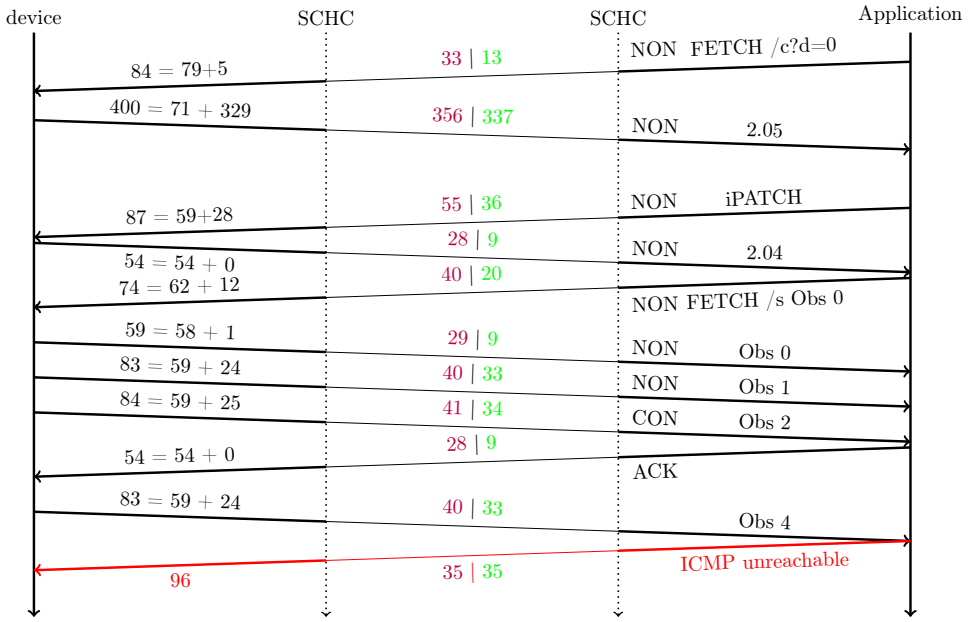
SCHC



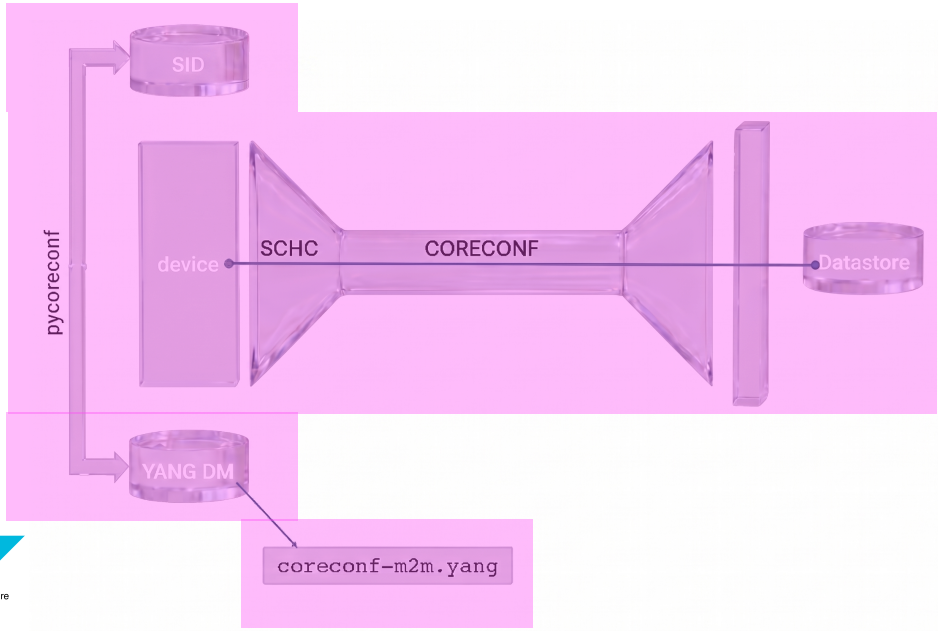
IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom



COMPRESSION RESULTS

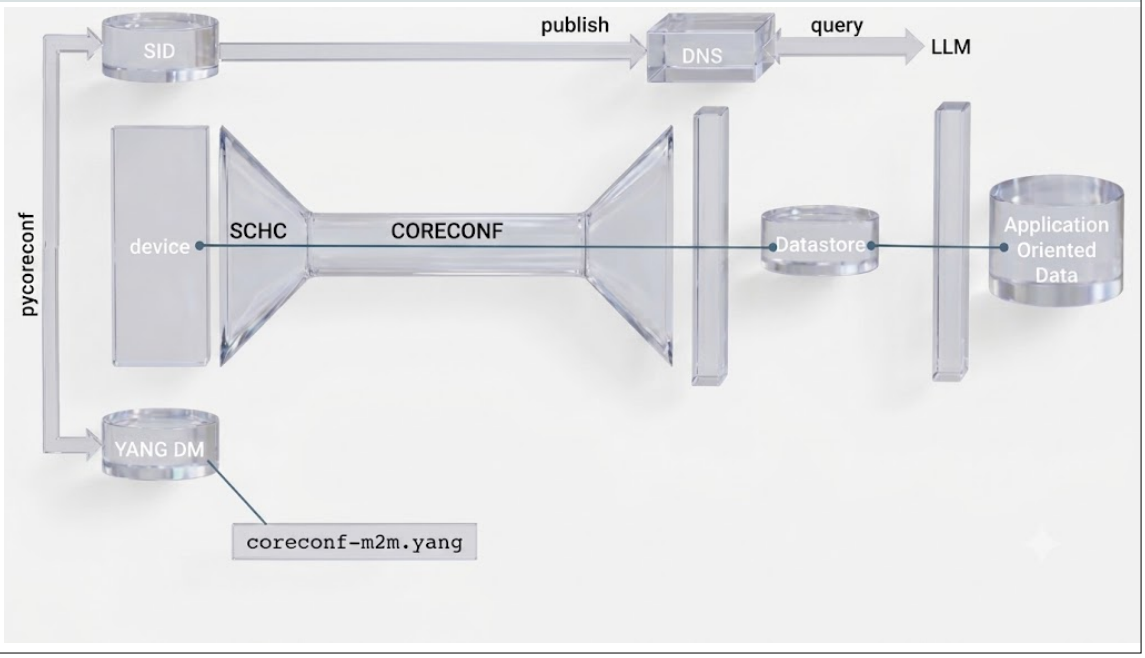


Global architecture



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Global architecture



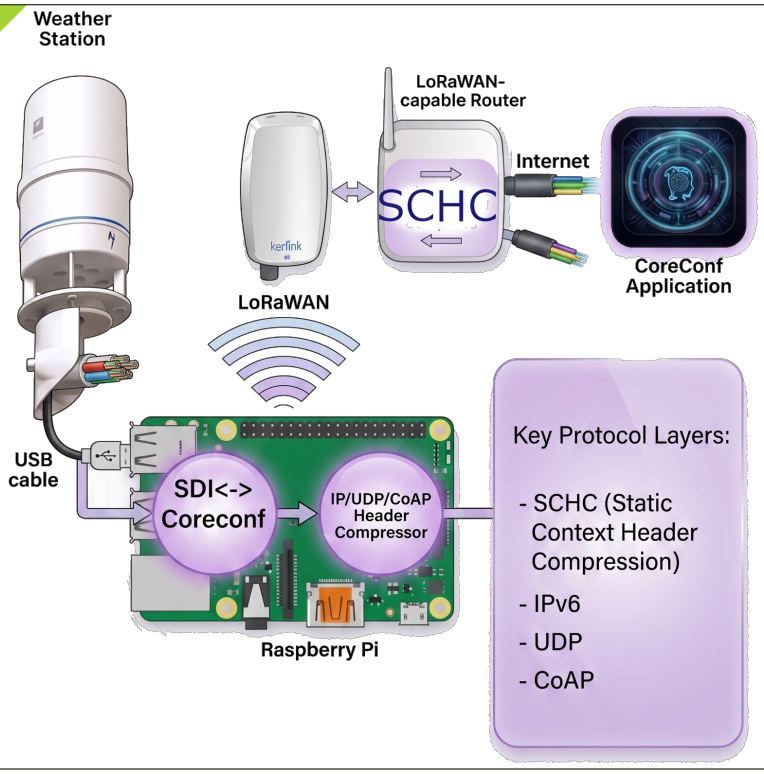
IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

NEXT STEPS

1. CORECONF HAS NEVER BEEN SO EASY.
2. ADD DYNAMIC RULE MANAGEMENT
3. REAL LORAWAN NETWORK
4. SATELLITE COMMUNICATIONS
5. LINK TO ONTOLOGIES

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

QUESTION?



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom