ASDF @IETF 110 Hackathon and WoT Plugfest

Extension features for SDF .next

Design Patterns and Examples

Michael Koster

March 1, 2021

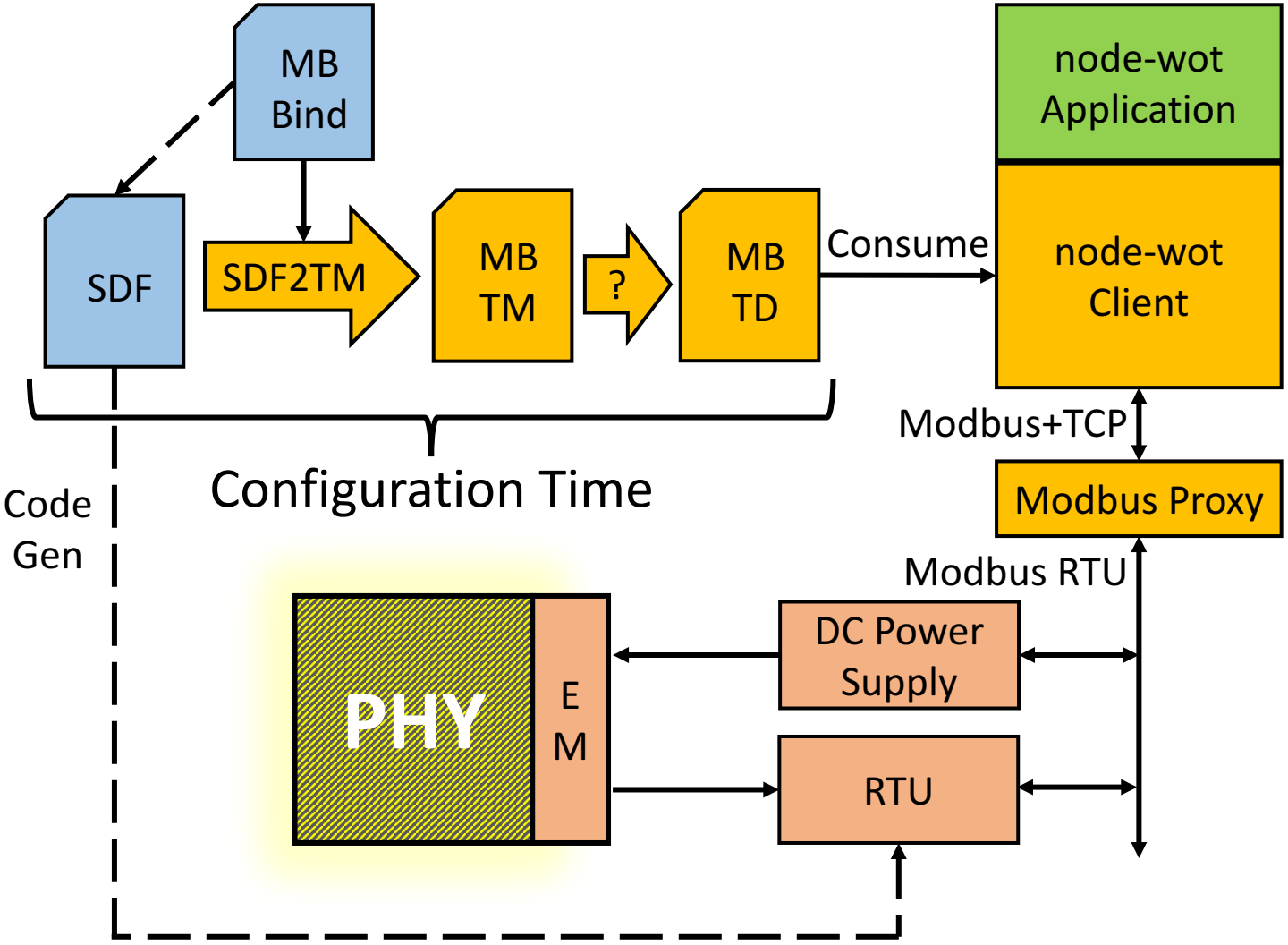
1.next

- ASDF Language and Processing Features
- Mapping File formats
- SDF Language Extension schemas
- Versioning of models

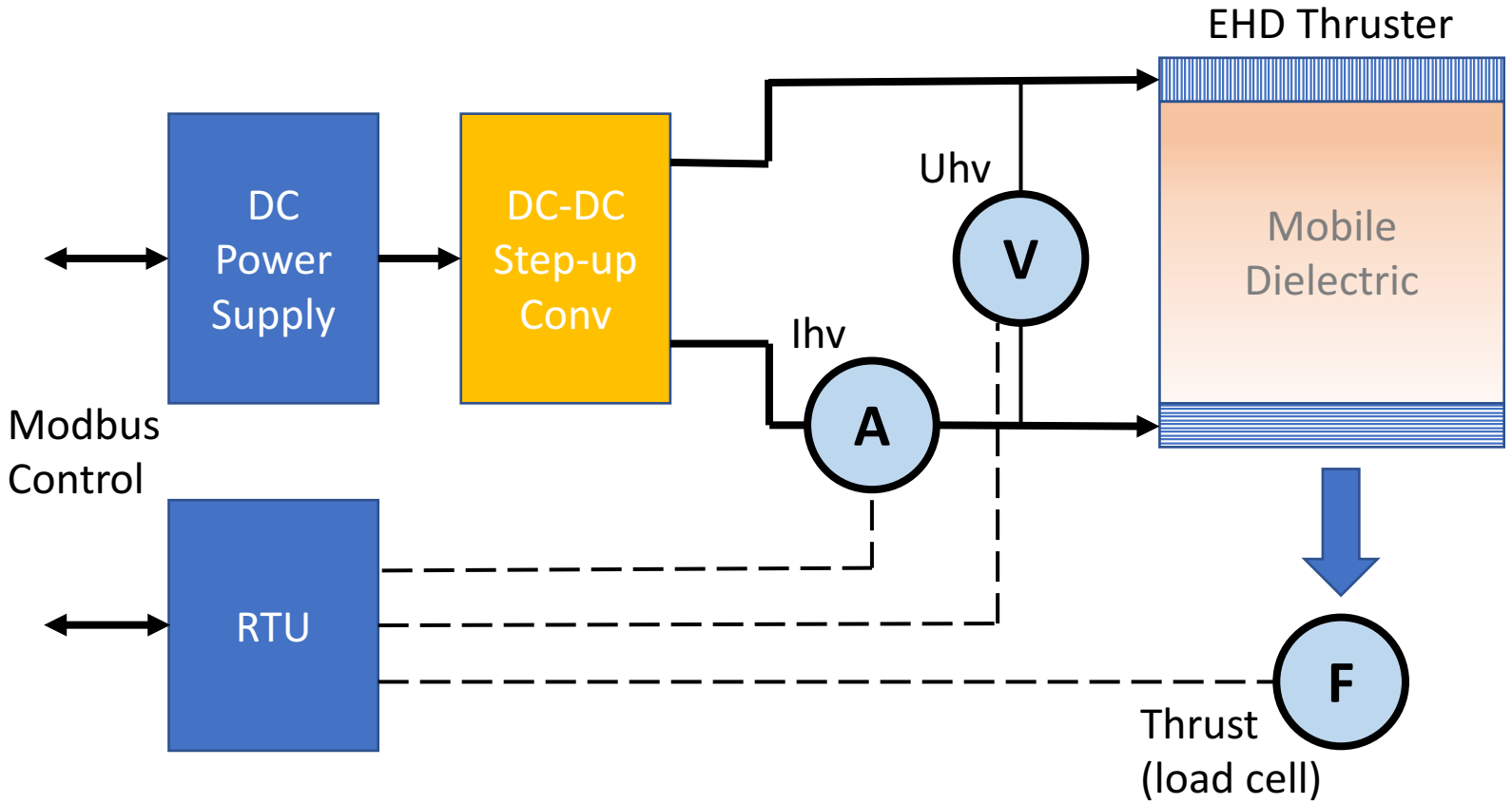
W3C WoT and SDF/OneDM

- Conversion workflow: SDF => Thing Model => Thing Description
  - Mapping files, protocol bindings, TD Forms
- SDF Processing for external references
  - TD Validation of annotations
- Semantic Proxy using node-wot
  - Multiple proof points

Modbus Integration Test Case



EHD Propulsion Experiment "Ionic Wind Tunnel"



Ref: https://www.researchgate.net/publication/235189622_A_Model_of_an_Ideal_Electrohydrodynamic_Thruster

Mapping Files, Bindings, and Extension Points

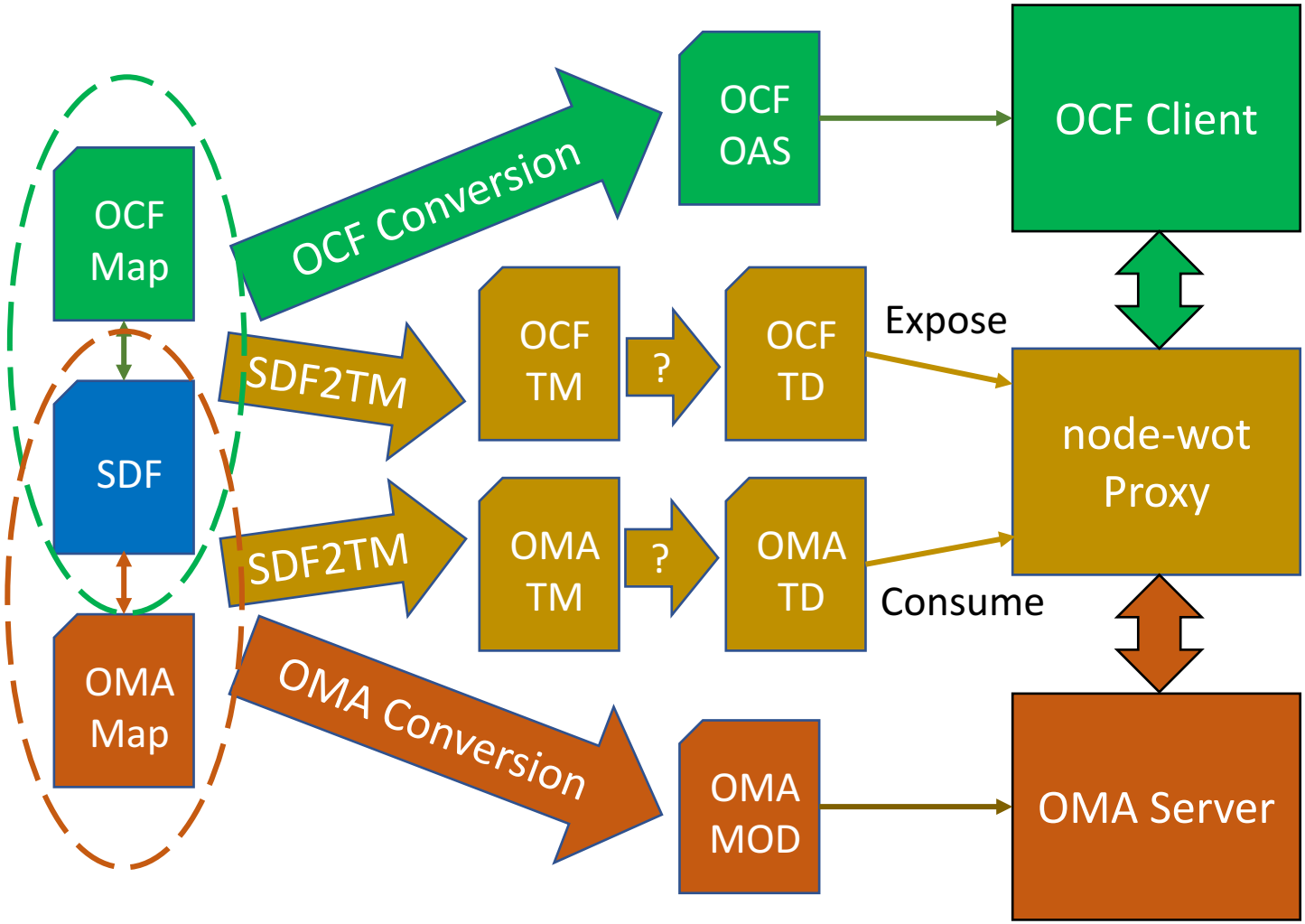
- Examples of extension points based on auxiliary schemas
  - DataSchema, ProtocolBinding, Hyperlink
- Declaration Format - "extensionPoint" in info block
- How to identify points in the base schema that the extension point is valid for

Extension Declaration Format

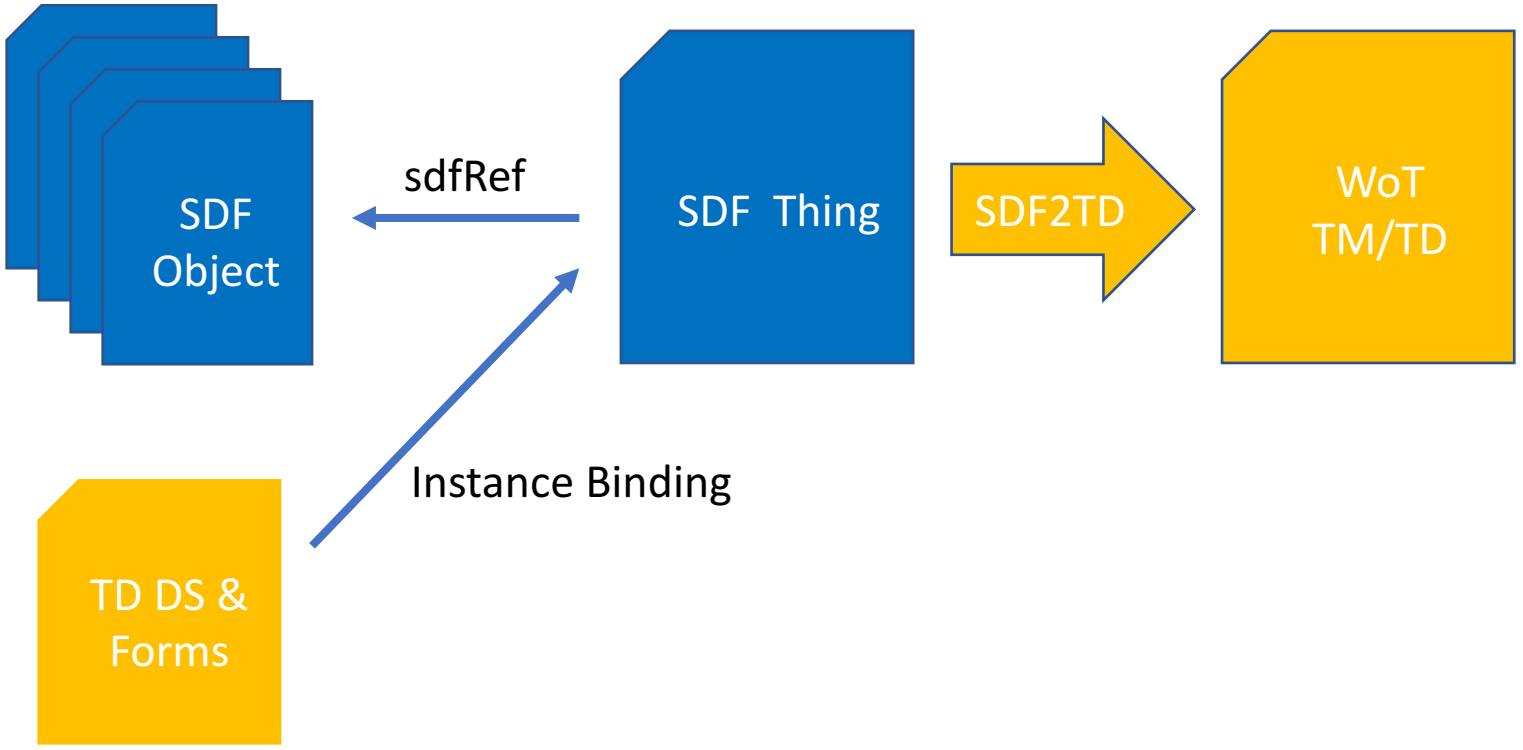
```
"extensionPoint": {
  "DataSchema" : {
    "description": "Additional data schema constraints and mappings",
    "sdfRef": "#/sdfExtensionSchema/DataSchema"
  },
  "ProtocolBinding": {
    "description": "Protocol Binding Form compatible with WoT TD Form",
    "sdfRef": "#/sdfExtensionSchema/ProtocolBinding"
  }

"sdfAction": {
  "SetAmperage": {
    "sdfInputData":{
      "sdfRef": "#/sdfThing/DCPowerSupply_LW3010E/sdfData/AmperageData",
      "DataSchema": {
        "WidthInBits": 16
      }
    },
    "ProtocolBinding": {
      "href": "tcp+modbus://192.168.1.1:502/",
      "modv:unitID": 1,
      "modv:regID": 1001,
      "modv:function": "WriteHoldingRegister"
    }
```

Semantic Proxy



Model Construction Workflow



References

Back up

Versioning scheme

(modification to YANG's new versioning scheme for OneDM)

```
<model version> +
<authority> +
<OPTIONAL dev version>
```

Pros:

- Can handle multiple dev paths
- Rich info in one tag

Cons:

- Complex
- Overkill?

YANG's mechanism can't be copied over without modification due to different stages in process. Several "error cases" need to be handled

```
1.2.3                       model version in ecoX
0.1.0-ecoX-00               initial contribution to PG
0.1.0-ecoX-01
0.2.0-ecoX-02
1.0.0-ecoX-03               Promotor proposes adoption
1.0.0                       Adopted version
< New features needed, two alternatives >
1.1.0-ecoX-01               1.1.0-ecoY-01
1.1.0-ecoX-02               1.1.0-ecoY-02
< Merge alternatives >
1.1.0-ecoX-03               Merged proposal
1.1.0-ecoX                  Promotor
```

OLD Model versioning

https://github.com/one-data-model/processes/blob/master/versioning.md

Principles

- Use Semantic versioning and tags
- Don't do properties/feature detection

- **`version`** (1.0.0, 1.1.0) – semantic versioning
  - Major X.y.z => breaking backwards compatibility
  - Minor x.Y.z => add new features only
  - Patch x.y.Z => patch = change to non-critical text description only
- **`version-tags`** (odm, playground,dev, ema, ocf) – version metadata for develop
  - od
  - An to in
  - In