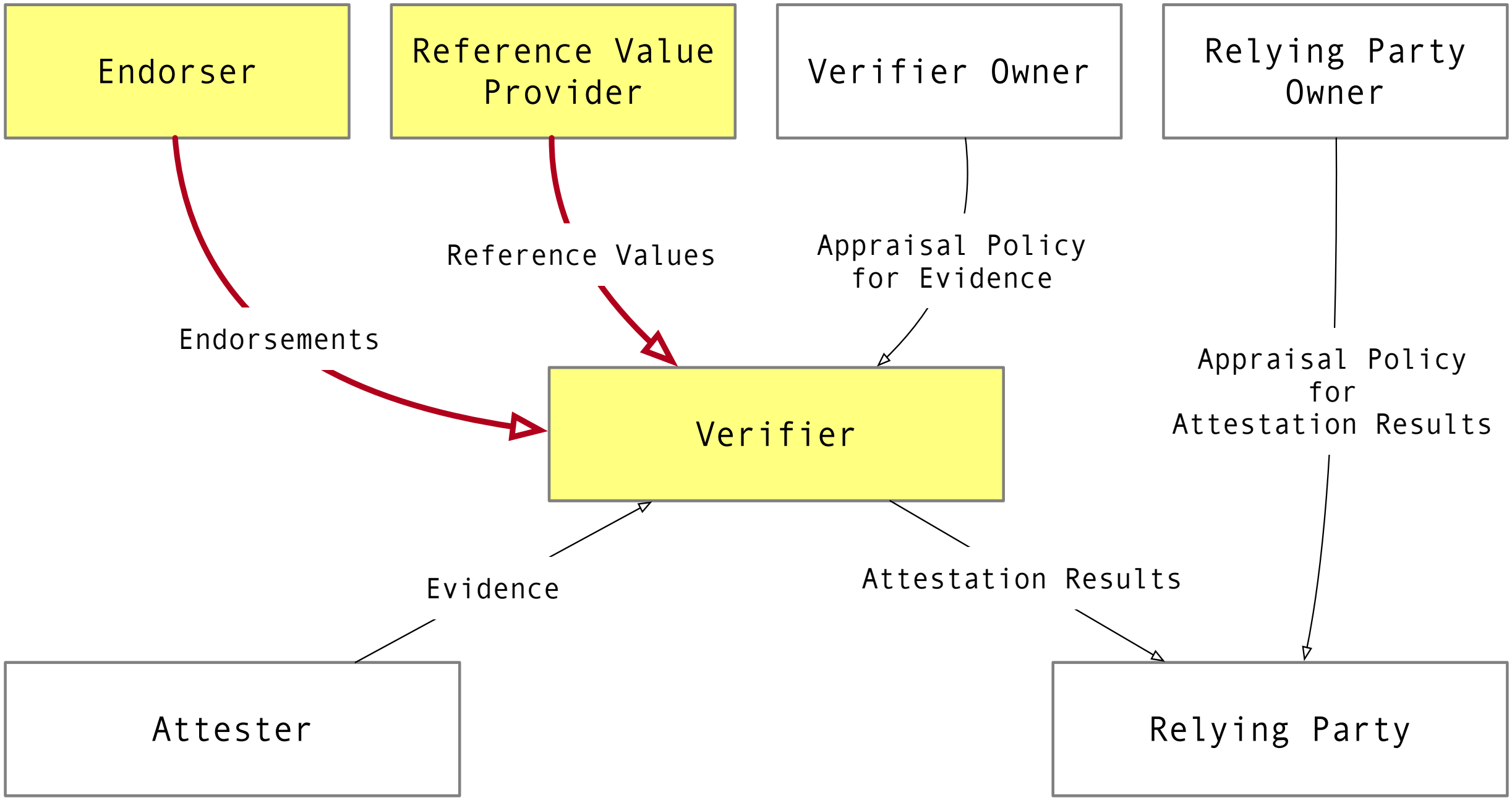
CoRIM, quick recap

Problem Statement

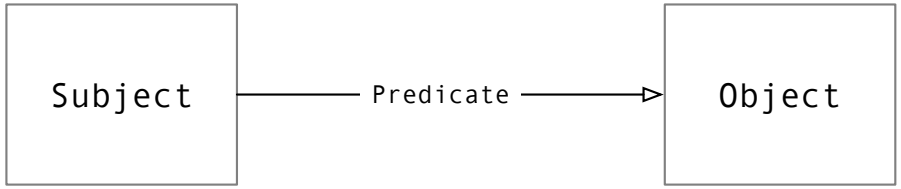
One or more authorized supply chain actors (OEM, ISVs, SiPs, etc.) need to come together and "describe" an Attester to a Verifier. So, when Evidence from that Attester is passed on to the Verifier, it can use the attributes that apply to the Attester to evaluate Evidence against the Appraisal Policy

Standard IM/DM is likely to lead to standard tooling – reduce fragmentation, lower barrier to entry for the supply chain actors



RATS-ARCH, Conceptual Data Flow

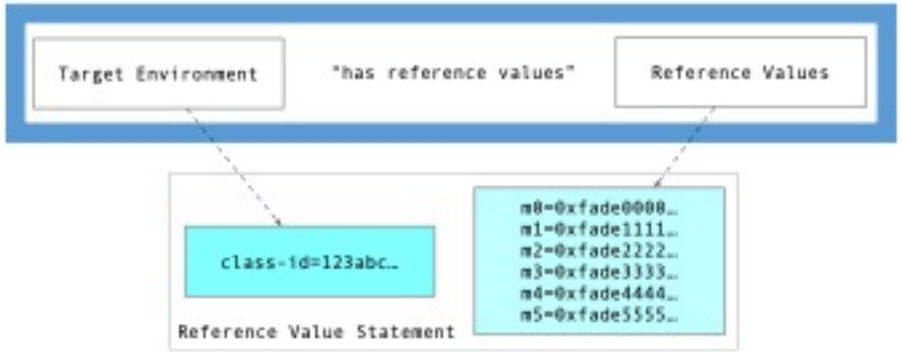
High level design

- Graph-based data model (RDF-like) with its own specialized vocabulary and data types
- The “triple” 

is the core construct
- Used to define a “Device / Attester” ontology
- Tracking triples provenance via explicit cryptographic means
- **Co**mpact representation (**Co**RIM, **Co**MID)

Triples

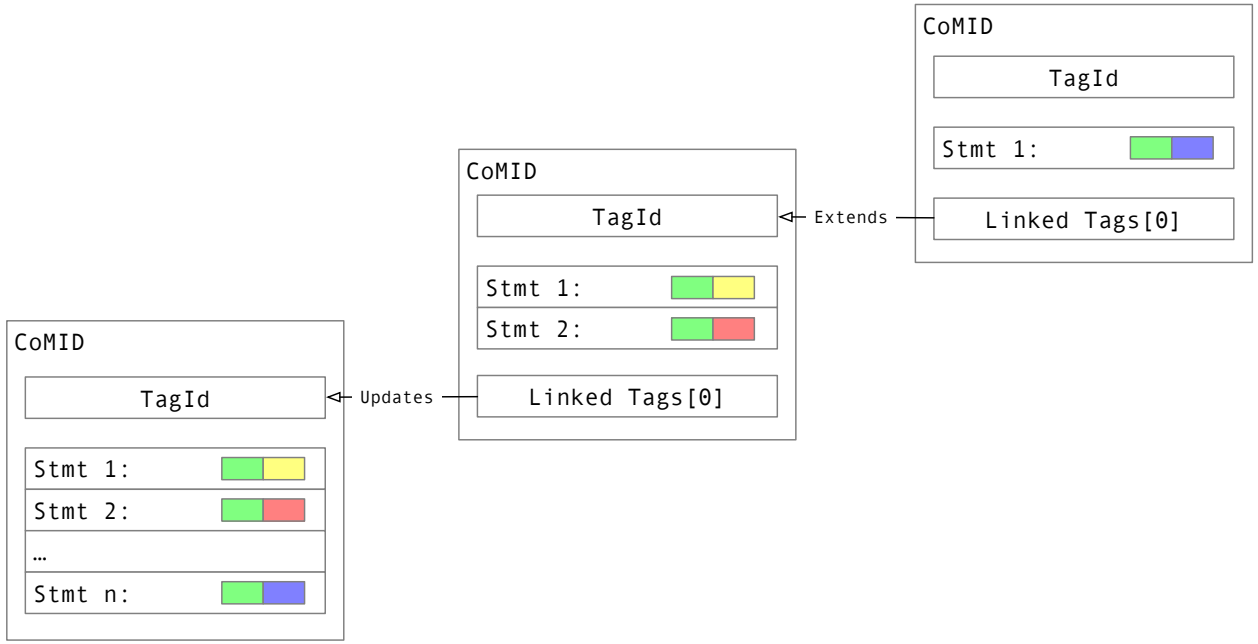
Reference Value Statement



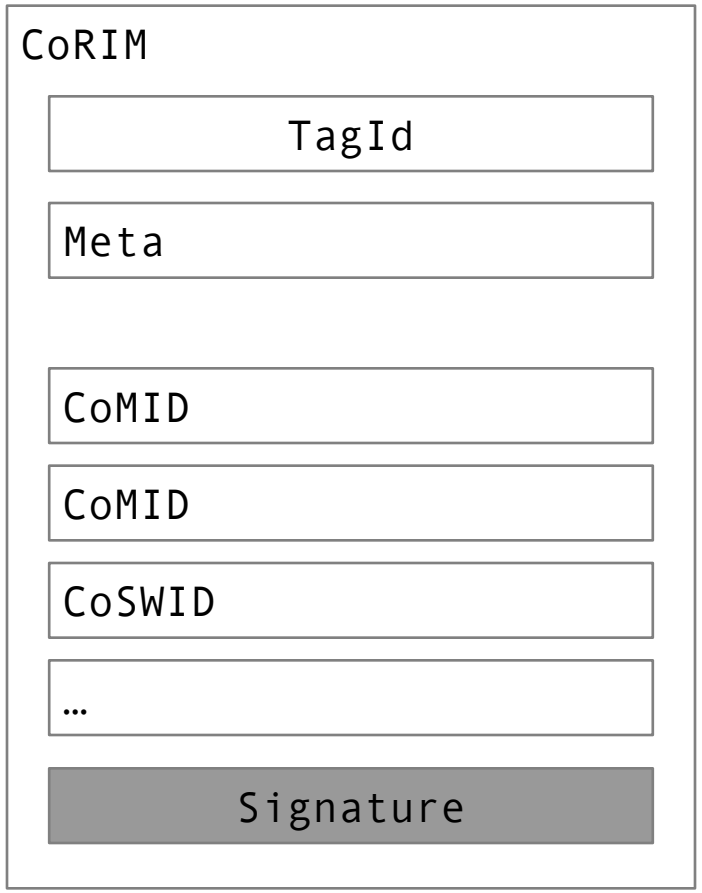
Endorsed Value Statement

Cryptographic Identity Statement

CoMIDs


CoMID
TagId
Stmt 1:
Stmt 2:
…
Stmt n:
Updates
CoMID
TagId
Stmt 1:
Stmt 2:
Linked Tags[0]
Extends
CoMID
TagId
Stmt 1:
Linked Tags[0]

CoRIMs



CoRIM, update

Specs

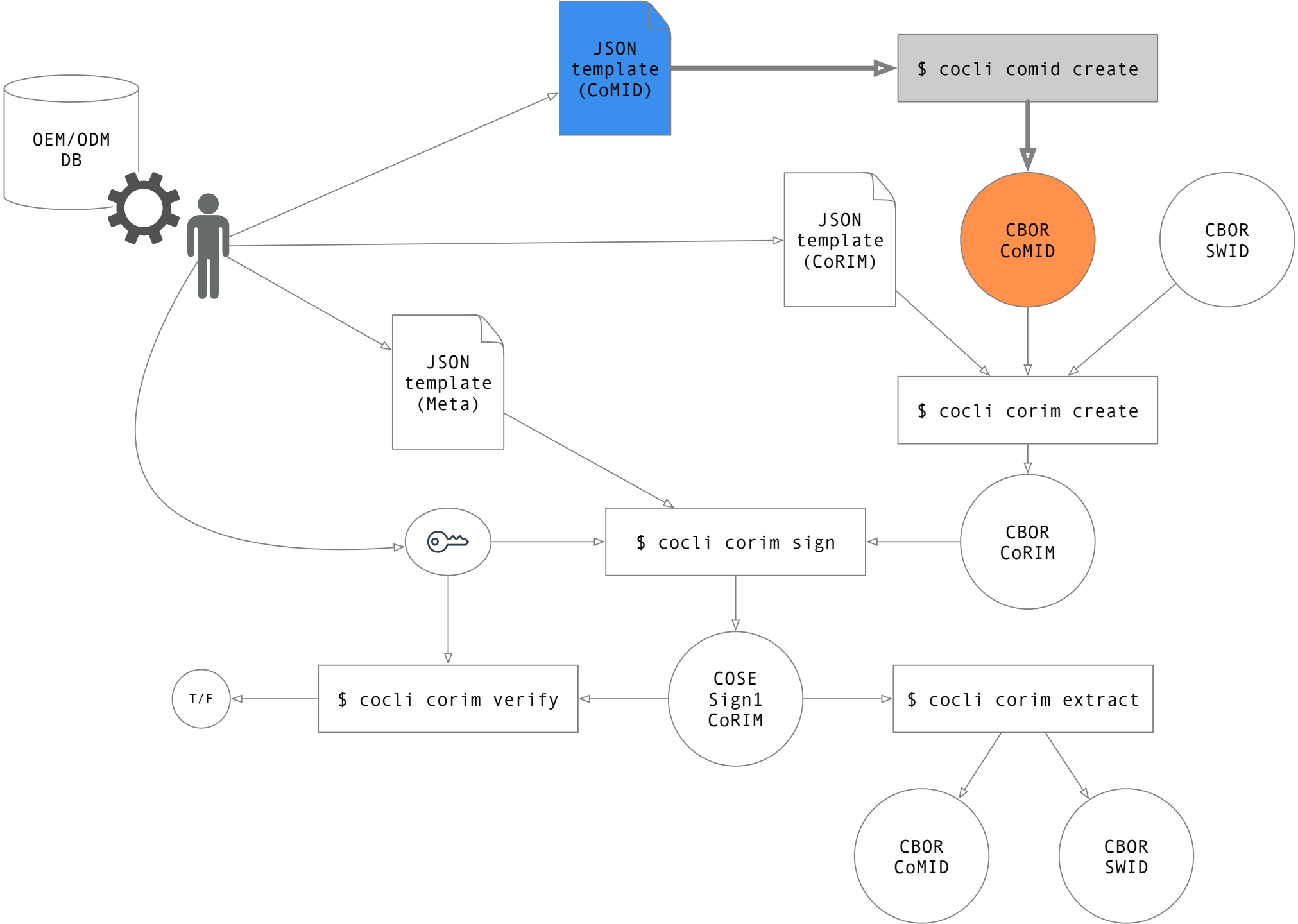
- Information model
  - TCG, DICE WG: “Endorsements Architecture”
    - Not public yet -- internal review stage
- Data model
  - https://github.com/ietf-rats/ietf-corim-cddl (bleeding edge)
    - CDDL, examples in CBOR diagnostic notation, assembly & test infra
  - IETF I-D https://datatracker.ietf.org/doc/html/draft-birkholz-rats-corim

Running Code

- CoRIM (https://github.com/veraison/corim)
- Go packages:
  - `github.com/veraison/corim/corim`
    - Low-level CoRIM manipulation – CBOR, JSON (bespoke) codecs
  - `github.com/veraison/corim/comid`
    - Low-level CoMID manipulation – CBOR, JSON (bespoke) codecs
  - `github.com/veraison/corim/cocli`
    - Command Line Interface to deal CoRIMs and CoMIDs (and to a smaller extent, CoSWIDs) for the end user

- SWID (https://github.com/veraison/swid)
- Go package:
  - `github.com/veraison/swid`
    - XML (SWID, ISO/IEC 19770-2:2015, NISTIR-8060),
    - CBOR (CoSWID, draft-ietf-sacm-coswid) and JSON (bespoke)

A tour of `cocli`


JSON
template
(CoMID)
$ cocli comid create
OEM/ODM
DB
JSON
template
(CoRIM)
CBOR
CoMID
CBOR
SWID
JSON
template
(Meta)
$ cocli corim create
$ cocli corim sign
CBOR
CoRIM
T/F
$ cocli corim verify
COSE
Sign1
CoRIM
$ cocli corim extract
CBOR
CoMID
CBOR
SWID

```
$ cocli comid create --template t1.json --output-dir /tmp
>> created "/tmp/t1.cbor" from "t1.json"
```

Template file

Templates folder

```
$ tree templates/
templates/
├── t1.json
├── t2.json
...
└── tn.json
```

```
$ cocli comid create --template-dir templates
>> created "t1.cbor" from "templates/t1.json"
>> created "t2.cbor" from "templates/t2.json"
...
>> created "tn.cbor" from "templates/tn.json"
```

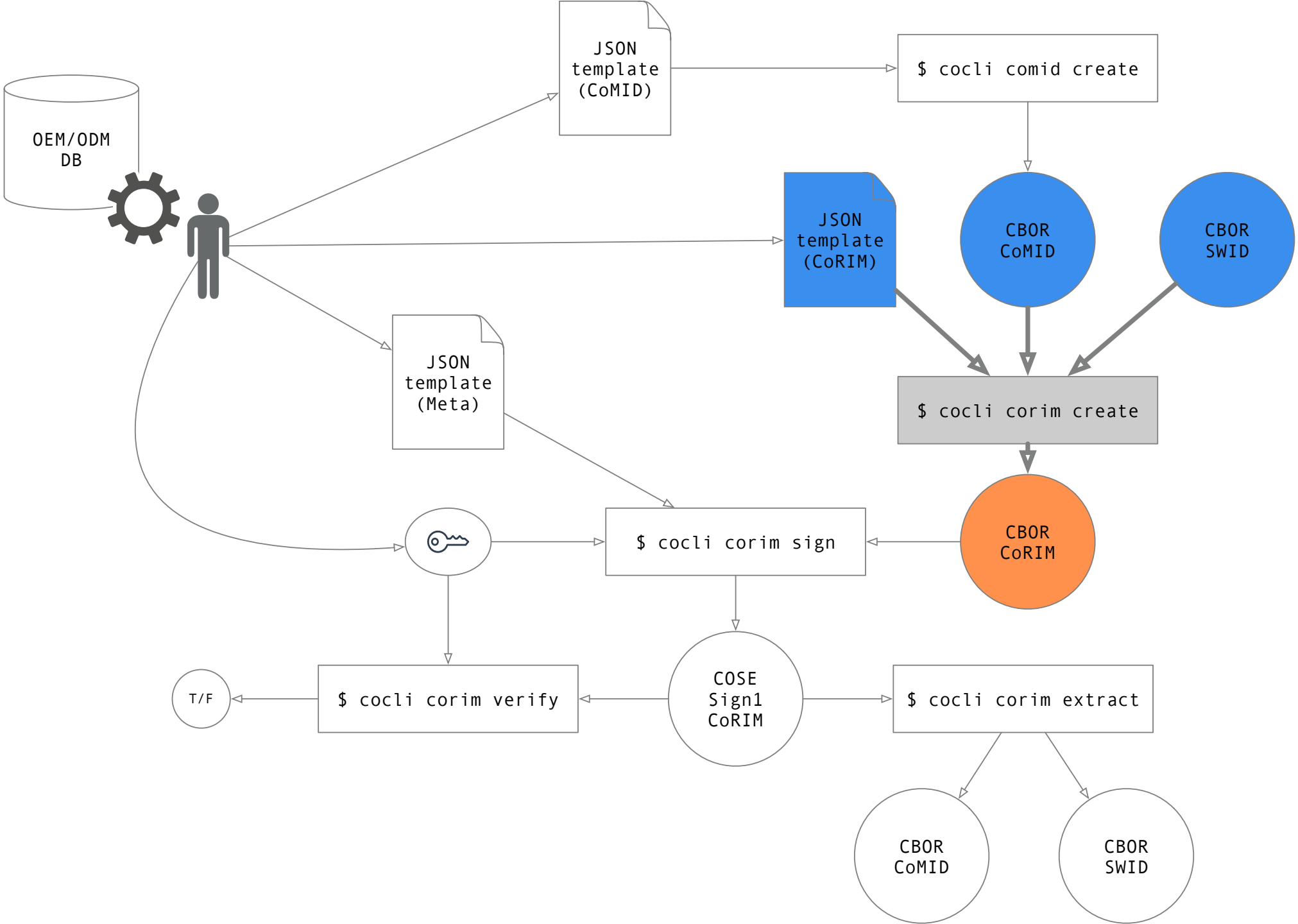
```
$ cocli comid create -T comid-templates/ \
                     -T comid-templates-aux/ \
                     -t extra-comid.json \
                     -t yet-another-comid.json \
                     -o /var/spool/comid
```

Template files and folders

CoMID template example


```
{
  "lang": "en-GB",
  "tag-identity": {
    "id": "366D0A0A-5988-45ED-8488-2F2A544F6242",
    "version": 0
  },
  "entities": [
    {
      "name": "ACME Ltd.",
      "regid": "https://acme.example",
      "roles": [
        "tagCreator",
        "creator",
        "maintainer"
      ]
    }
  ],
  "triples": {
    "attester-verification-keys": [
      {
        "environment": {
          "class": {
            "id": {
              "type": "psa.impl-id",
              "value": "YWNtZS1pbXBsZW1lbnRhdGlvbi1pZC0wMDAwMDAwMDE="
            },
            "vendor": "ACME",
            "model": "RoadRunner"
          },
          "instance": {
            "type": "ueid",
            "value": "Ac7rrnuJJ6MiflMDz14PH3s0u1Qq1yUKwD+83jbsLxUI"
          }
        },
        "verification-keys": [
```

```
          {
            "key": "MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEFn0taoAwR3PmrKkYLtAsD9o05KSM6
          }
        ]
      },
      {
        "environment": {
          "class": {
            "id": {
              "type": "psa.impl-id",
              "value": "YWNtZS1pbXBsZW1lbnRhdGlvbi1pZC0wMDAwMDAwMDE="
            },
            "vendor": "ACME",
            "model": "RoadRunner"
          },
          "instance": {
            "type": "ueid",
            "value": "AUyj5PUL8kjDl4cCDWj/0FyIdndRvyZFypI/V6mL7NKW"
          }
        },
        "verification-keys": [
          {
            "key": "MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE6Vwqe7hy3O8Ypa+BUETLUjBNU3rEX
          }
        ]
      }
    ]
  }
}
```


OEM/ODM
DB
JSON
template
(CoMID)
$ cocli comid create
JSON
template
(CoRIM)
CBOR
CoMID
CBOR
SWID
JSON
template
(Meta)
$ cocli corim create
$ cocli corim sign
CBOR
CoRIM
T/F
$ cocli corim verify
COSE
Sign1
CoRIM
$ cocli corim extract
CBOR
CoMID
CBOR
SWID

CoMID and CoSWID files

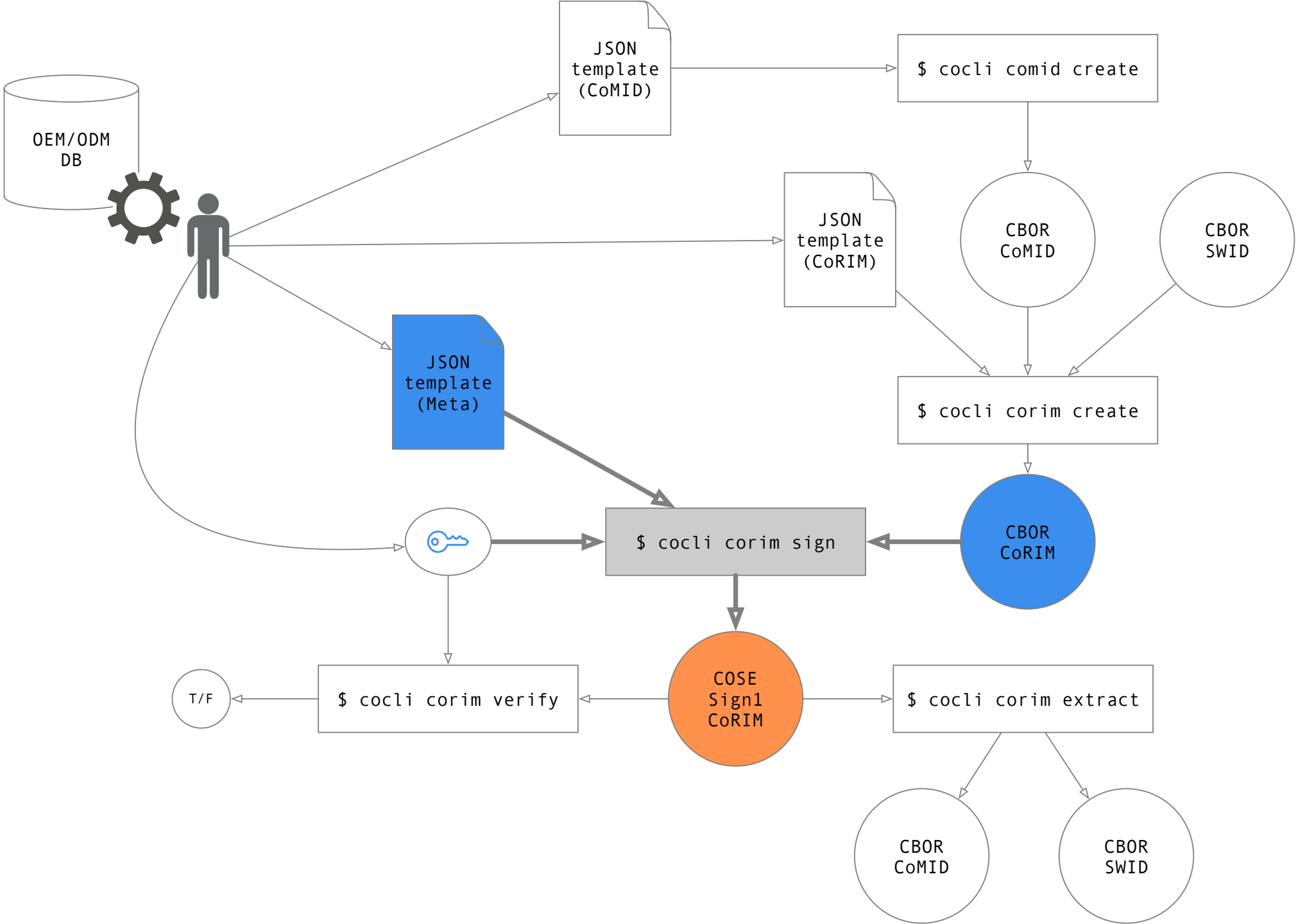
```
$ cocli corim create -t c1.json -m m1.cbor -s s1.cbor -o my.cbor
>> created "my.cbor" from "c1.json"
```

CoMIDs from folder

```
$ cocli corim create --template c1.json --comid-dir comids.d/
```

CoRIM template example


```
{
  "corim-id": "5c57e8f4-46cd-421b-91c9-08cf93e13cfc",
  "dependent-rims": [
    {
      "href": "https://parent.example/rims/ccb3aa85-61b4-40f1-848e-02ad6e8a254b",
      "thumbprint": "sha-256:5Fty9cDAtXLbTY06t+l/No/3TmI0eoJN7LZ6hOUiTXU="
    }
  ],
  "profiles": [
    "1.3.6.1.4.1.4128.100",
    "http://arm.com/iot/profile/1"
  ],
  "validity": {
    "not-before": "2021-12-31T00:00:00Z",
    "not-after": "2025-12-31T00:00:00Z"
  },
  "entities": [
    {
      "name": "ACME Ltd.",
      "regid": "acme.example",
      "roles": [
        "manifestCreator"
      ]
    }
  ]
}
```


OEM/ODM
DB
JSON
template
(CoMID)
$ cocli comid create
JSON
template
(CoRIM)
CBOR
CoMID
CBOR
SWID
JSON
template
(Meta)
$ cocli corim create
$ cocli corim sign
CBOR
CoRIM
T/F
$ cocli corim verify
COSE
Sign1
CoRIM
$ cocli corim extract
CBOR
CoMID
CBOR
SWID

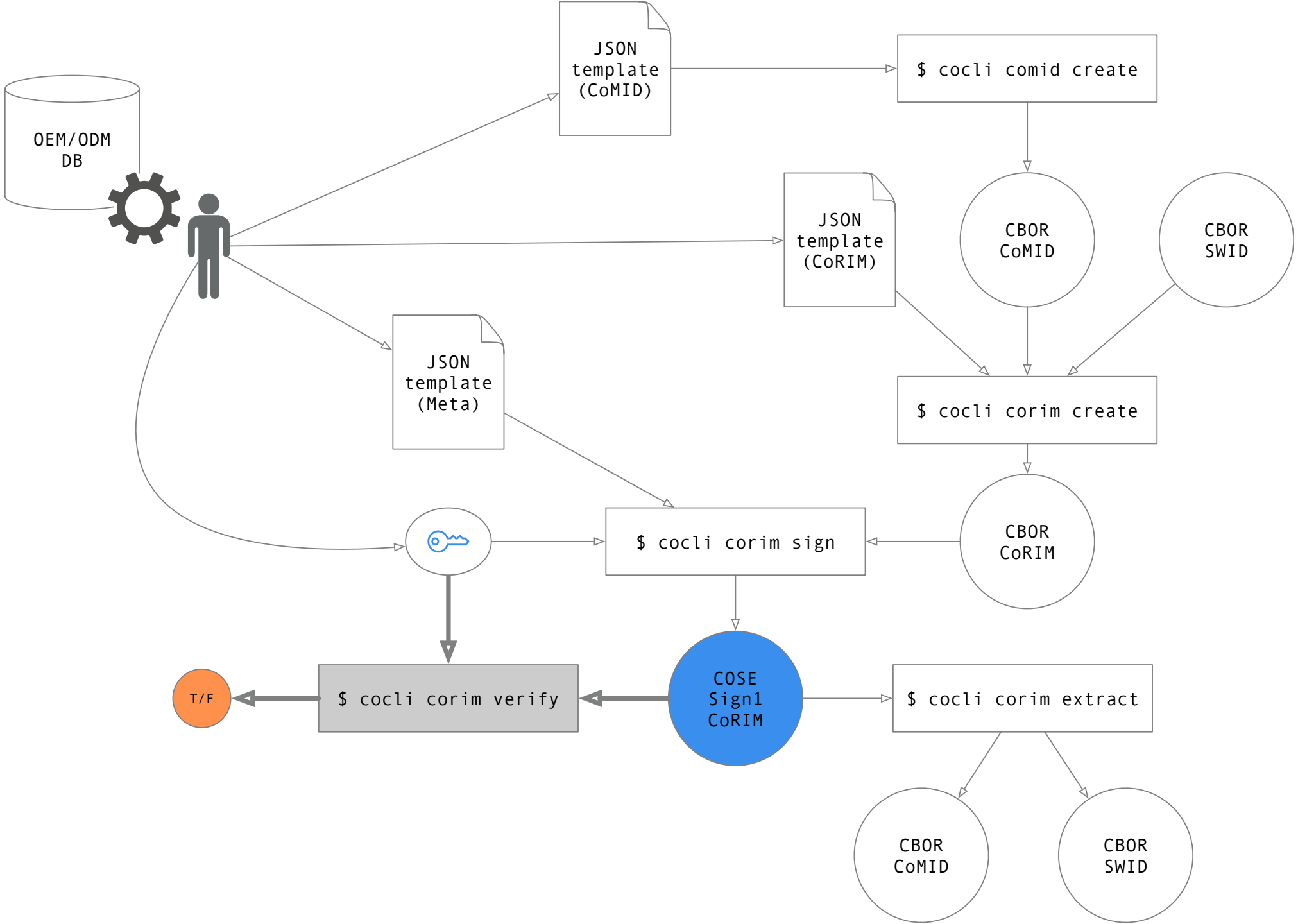
```
$ cocli corim sign --file corim.cbor \
                   --key ec-p256.jwk \
                   --meta meta.json \
                   --output /var/spool/signed-corim.cbor
>> "corim.cbor" signed and saved to "/var/spool/signed-corim.cbor"
```

```
{
  "kty": "EC",
  "crv": "P-256",
  "x": "MKBCTNIcKUSDii11ySs3526iDZ8AiTo7Tu6KPAqv7D4",
  "y": "4Etl6SRW2YiLUrN5vfvVHuhp7x8PxltmWWlbbM4IFyM",
  "d": "870MB6gfuTJ4HtUnUvYMyJpr5eUZNP4Bk43bVdj3eAE",
  "use": "enc",
  "kid": "1"
}
```

Signing key

```
{
  "signer": {
    "name": "ACME Ltd signing key",
    "uri": "https://acme.example"
  },
  "validity": {
    "not-before": "2021-12-31T00:00:00Z",
    "not-after": "2025-12-31T00:00:00Z"
  }
}
```

CoRIM Meta template


OEM/ODM
DB
JSON
template
(CoMID)
$ cocli comid create
JSON
template
(CoRIM)
CBOR
CoMID
CBOR
SWID
JSON
template
(Meta)
$ cocli corim create
$ cocli corim sign
CBOR
CoRIM
T/F
$ cocli corim verify
COSE
Sign1
CoRIM
$ cocli corim extract
CBOR
CoMID
CBOR
SWID

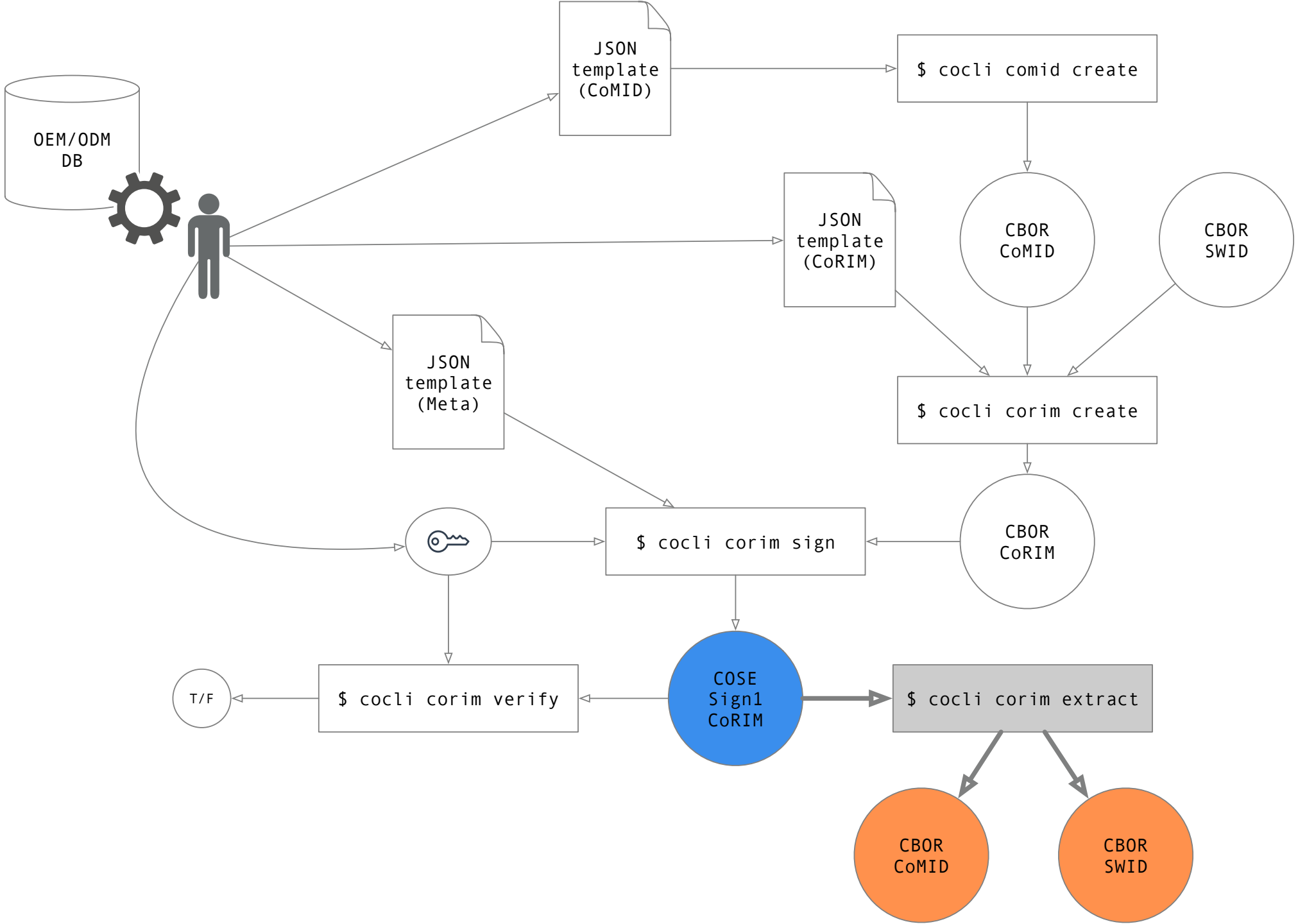
OK

```
$ cocli corim verify --file signed-corim.cbor --key ec-p256.jwk
>> "corim.cbor" verified
```

failure

```
$ cocli corim verify --file signed-corim-bad-signature.cbor --key ec-p256.jwk

Error: error verifying signed-corim-bad-signature.cbor with key ec-p256.jwk: verification failed ecdsa.Verify
```


OEM/ODM
DB
JSON
template
(CoMID)
$ cocli comid create
JSON
template
(CoRIM)
CBOR
CoMID
CBOR
SWID
JSON
template
(Meta)
$ cocli corim create
$ cocli corim sign
CBOR
CoRIM
T/F
$ cocli corim verify
COSE
Sign1
CoRIM
$ cocli corim extract
CBOR
CoMID
CBOR
SWID

```
$ cocli corim extract --file signed-corim.cbor --output-dir output.d/
$ tree output.d/
output.d/
├── 000000-comid.cbor
├── 000001-comid.cbor
└── 000002-coswid.cbor
```

And more…

- Display & validate CoMIDs
- Display & validate CoRIMs (Meta, embedded CoMIDs and CoSWIDs)

See https://github.com/veraison/corim/tree/main/cocli/README.md

Future

- Constantly keep up-to-date with DM
- Built-in CoSWID support
- JSON-schema validators for the templates
- More documentation