January 2025

Semantic Data Validation & Verification

Presenter: Marcel Wagner, Intel

Overview

- Motivation: IndustryFusion Foundation
- The Three Pillars of Semantic Validation
- Example: OPC UA Validation
- From Static to Runtime Validation: Challenges & Solution Space

Motivation: IndustryFusion Foundation

Why Semantic Data Matters

Context: IndustryFusion* Foundation (IFF) – Self Empowerment of Small and Medium Machine Builders and Manufacturers

- Founded by 60 manufacturers and stakeholders in the metal processing industry in Germany
- Focus: Metal sheet processing, cutting, welding, de-burring
- Aligned with Platform Industry 4.0
- Ownership of base platform
- Semantics based use-case and machine definition
- Open source (Apache 2.0)
- Cloud Native from shopfloor to Cloud

- https://www.industry-fusion.com/
- https://github.com/IndustryFusion

Image Source: Provided by IndustryFusion Foundation

Reference Use Case: „Smartness“ for OpEx

Legacy: Cutting System + Air Filter hardwired

- Safety regulations demands air filtering
- Hard-wire is reliable and safe, yet non-economical!

Opportunity: Knowledge-based “smart” filtering

- Strength of filter COULD be dependent on dimensions of workpiece
  - Use knowledge about the workpiece: material, thickness…
  - Control the strength of filtering – save operation energy
  - Retain the filtered air – save heating energy
  - Reduce hazard level of filter cartridges – save disposal cost

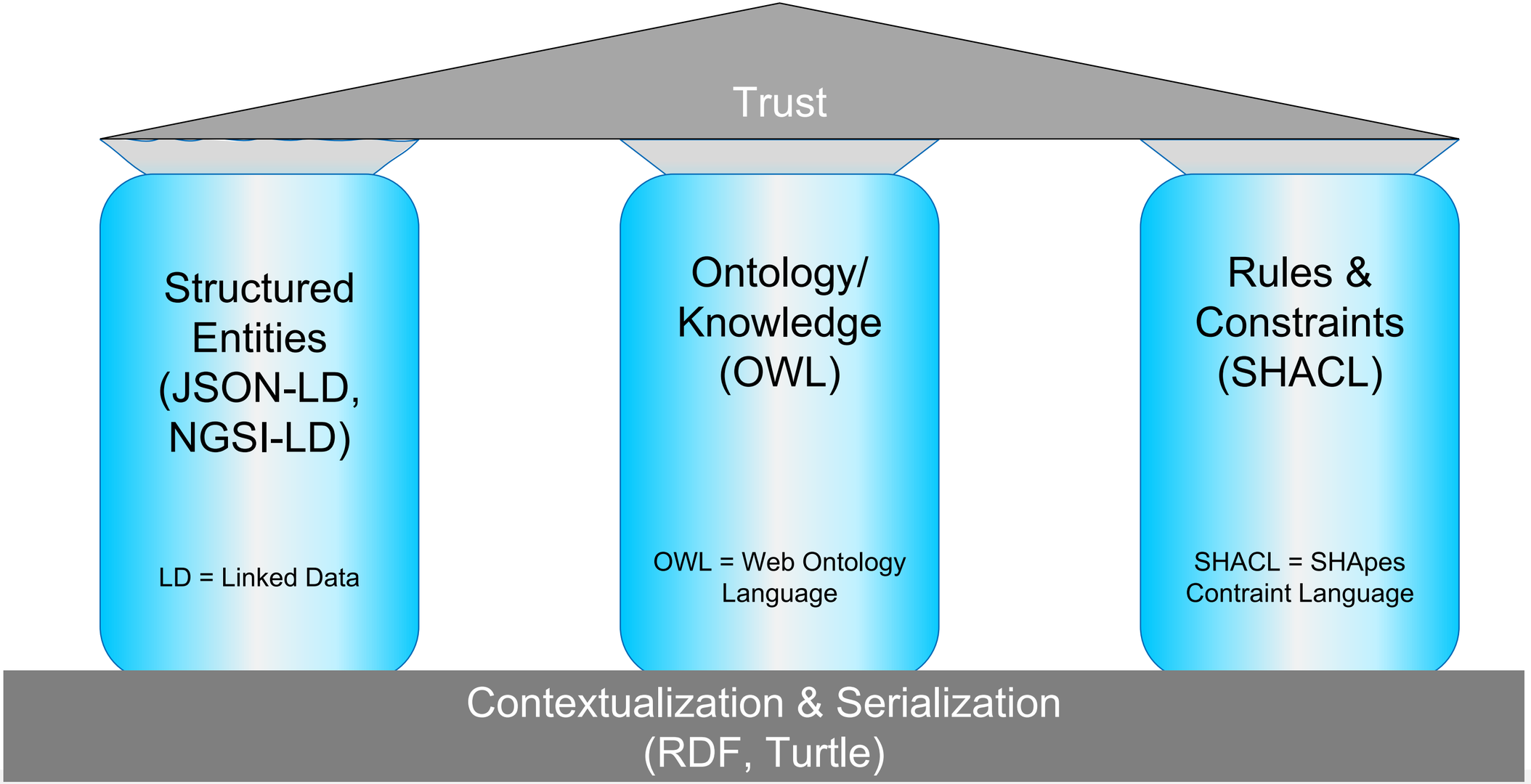
Image Source: Provided by IndustryFusion Foundation

The Three Pillars of Semantic Validation

Ensuring trust through structure, meaning and constraints

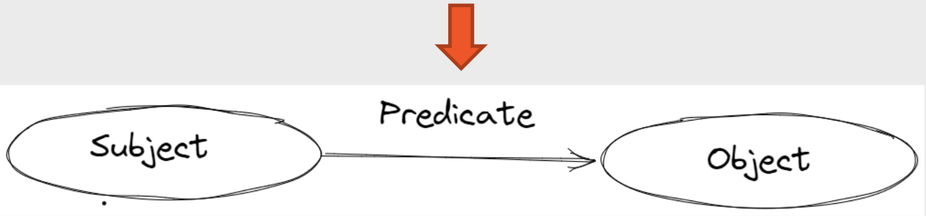
The Three Pillars of Semantic Validation

Ensuring trust through structure, meaning and constraints



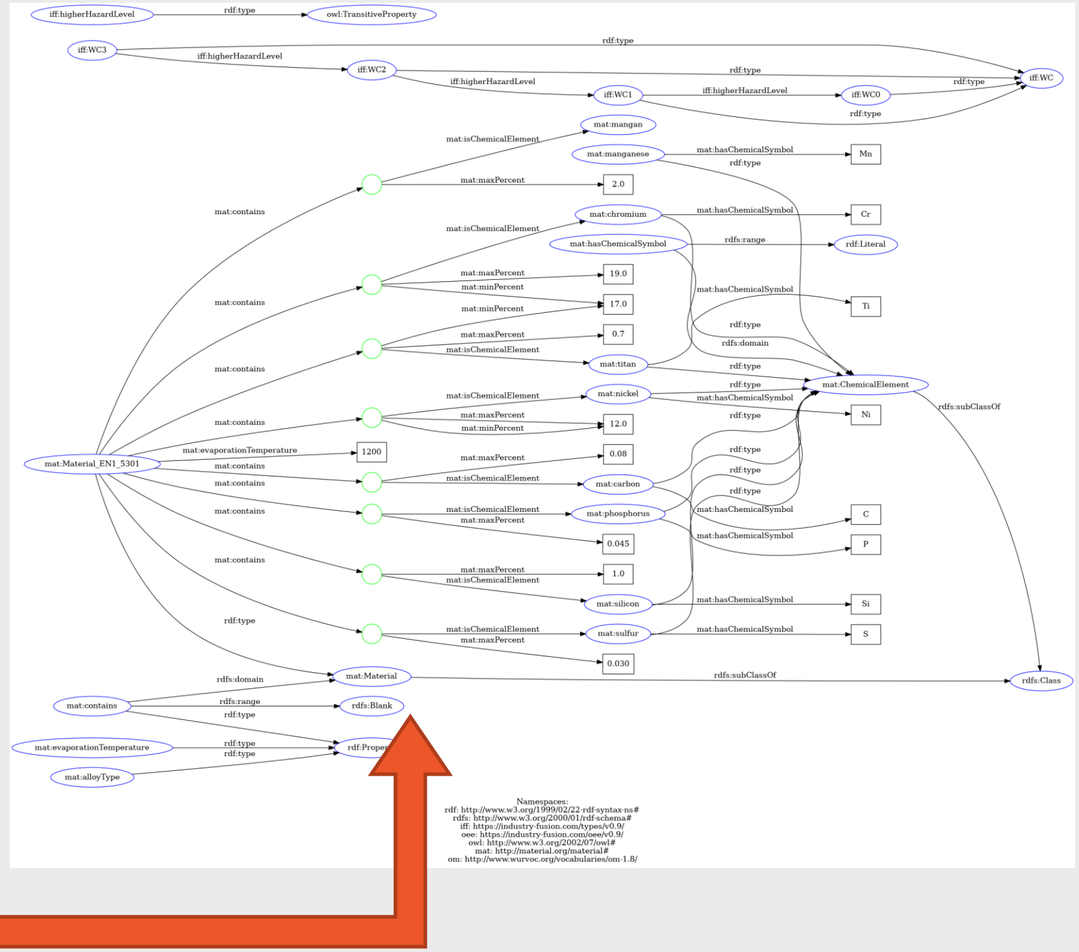
Ontology/Knowledge: OWL/RDF to Create (static) Knowledge Graphs

```
Subject Predicate Object .
```

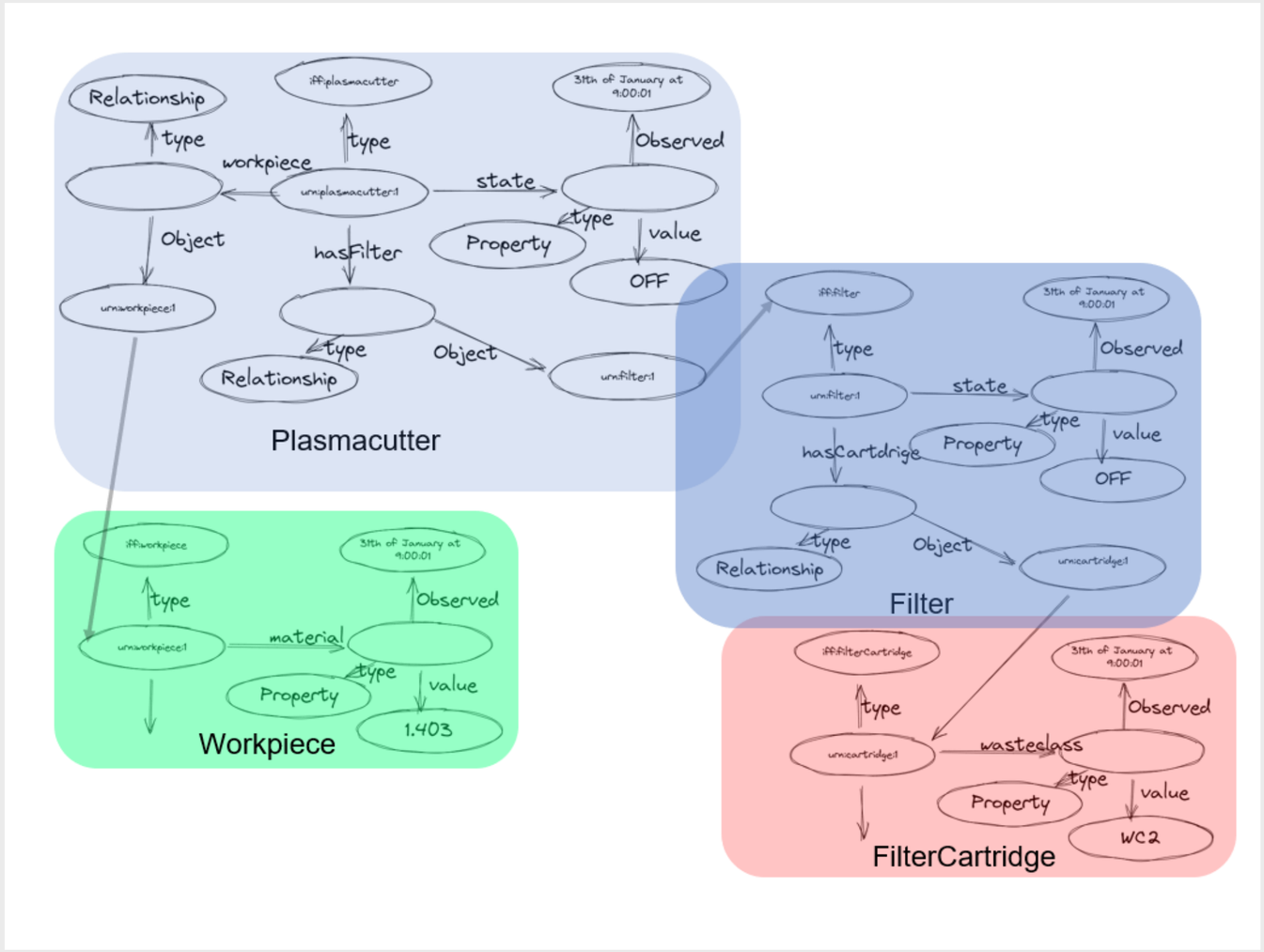



```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix iff: <https://industry-fusion.com/types/v0.9/> .
@prefix schema: <http://schema.org> .

iff:cutter rdf:type rdfs:Class .
iff:plasmacutter rdfs:subClassOf iff:cutter .
<urn:iff:microstep:cutter:1> rdf:type iff:plasmacutter .
<urn:iff:microstep:cutter:1> schema:serialNumber "12345" .
<urn:iff:microstep:cutter:1> iff:locatedIn <urn:iff:factory1> .
<https://www.microstep.eu> schema:owns <urn:iff:microstep:cutter:1> .
```

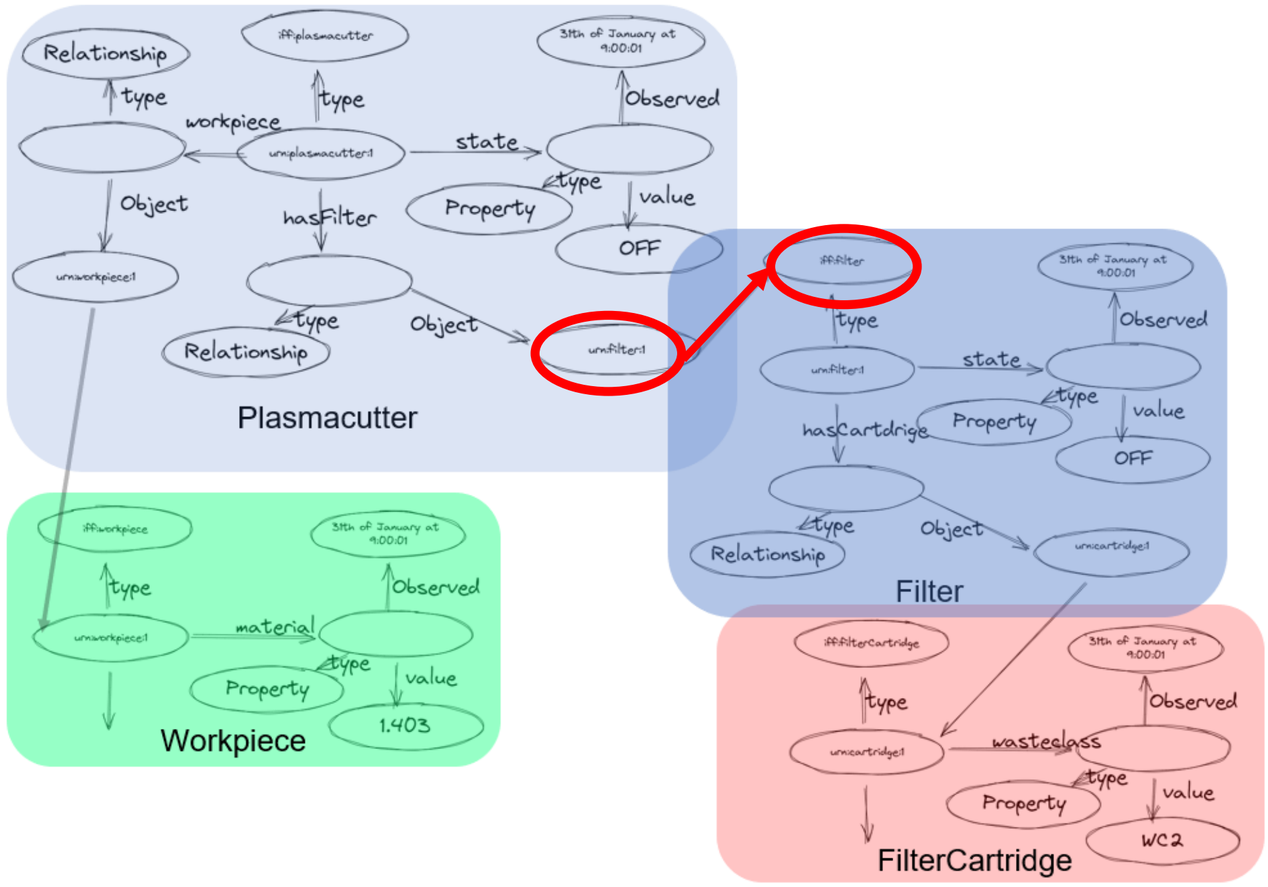


Structured Entities: JSON-LD (NGSI-LD)




```
{
    "@context": "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld",
    "id": "urn:plasmacutter:1",
    "type": "https://industry-fusion.com/types/v0.9/plasmacutter",
    "https://industry-fusion.com/types/v0.9/state": {
      "type": "Property",
      "value": {
        "@id": "https://industry-fusion.com/types/v0.9/state_ON"
      }
    },
    "https://industry-fusion.com/types/v0.9/hasWorkpiece": {
      "type": "Relationship",
      "object": "urn:workpiece:1"
    },
    "https://industry-fusion.com/types/v0.9/hasFilter": {
      "type": "Relationship",
      "object": "urn:filter:1"
    }
},
{
    "@context": ["https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld",
                 { "xsd": "http://www.w3.org/2001/XMLSchema#"}
    ],
    "id": "urn:filter:1",
    "type": "https://industry-fusion.com/types/v0.9/filter",
    "https://industry-fusion.com/types/v0.9/state": {
    "type": "Property",
    "value": {
        "@id": "https://industry-fusion.com/types/v0.9/state_ON"
    }
    },
    "https://industry-fusion.com/types/v0.9/strength": {
    "type": "Property",
    "value": {
      "@value": "1",
      "@type": "xsd:integer"
    }
    },
    "https://industry-fusion.com/types/v0.9/hasCartridge": {
    "type": "Relationship",
    "object": "urn:filterCartridge:1"
    }
},
```

Constraints & Rules: Shapes Constraint Language (SHACL)



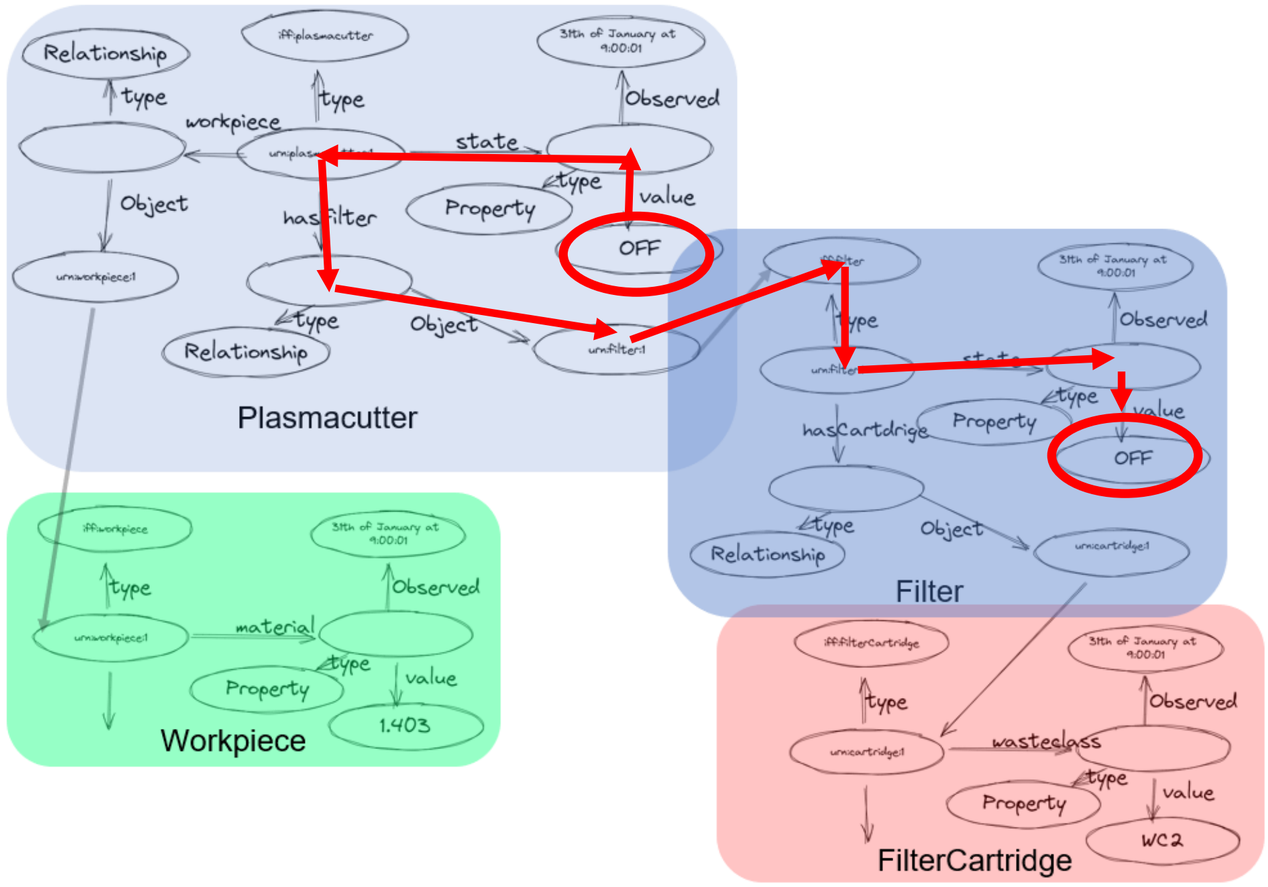

```
iff:CutterShape
    a sh:NodeShape ;
    sh:targetClass iff:cutter ;
    ...
    sh:property [
        sh:path <https://industry-fusion.com/types/v0.9/hasFilter> ;
        sh:minCount 1 ;
        sh:maxCount 1 ;
        sh:severity iff:severityCritical ;

        sh:property [
            sh:path <https://uri.etsi.org/ngsi-ld/hasObject> ;
            sh:class iff:filter ;
        ] ;
    ] ;
    ...
```

Validates

- Every entity of type ‘cutter’
- Whether there are exactly 1 ‘filter’ relationship from cutter
- Whether the linked filter is of type ‘filter’

Constraints & Rules: Shapes Constraint Language (SHACL)



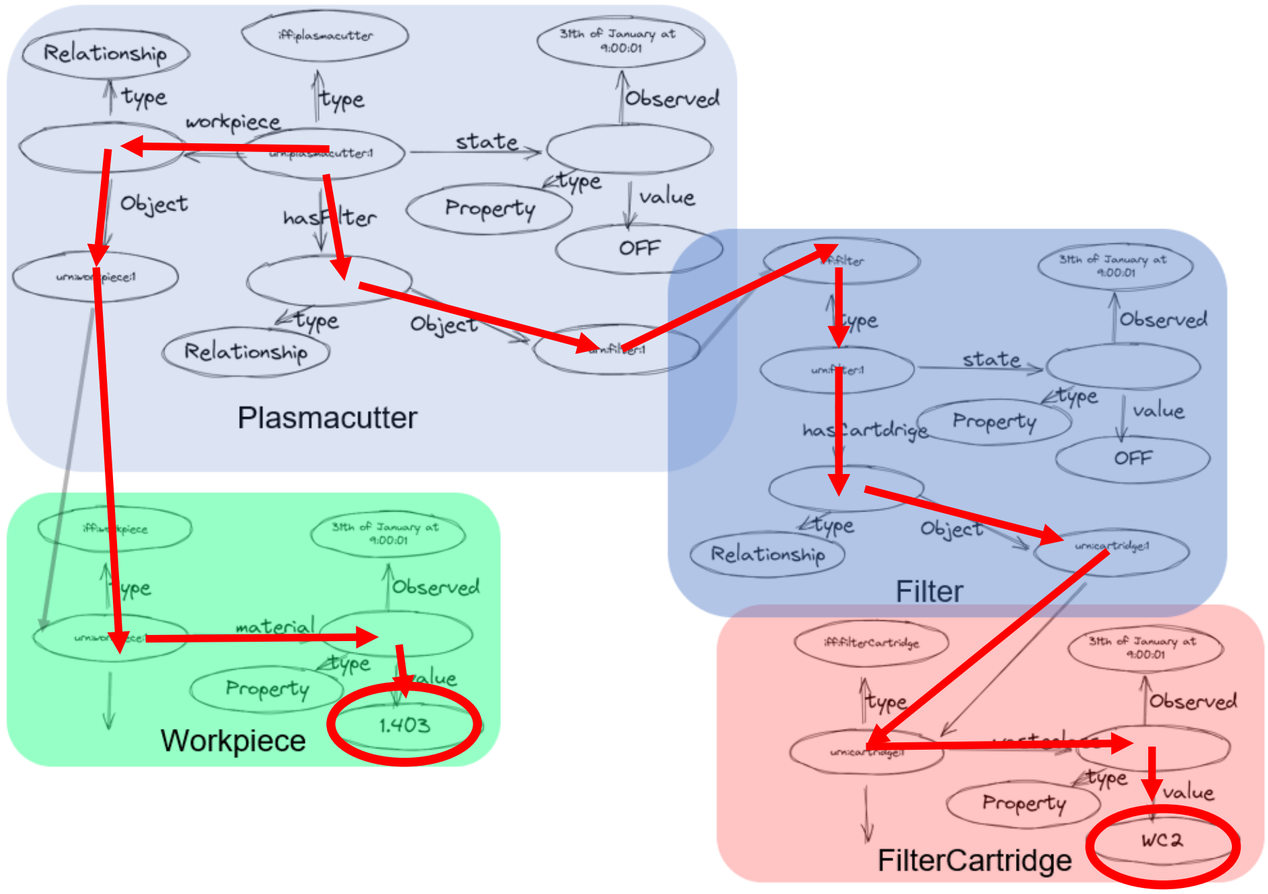

```
iff:StateOnCutter
    a sh:NodeShape ;
    sh:targetClass iff:cutter ;
    sh:sparql [
        a sh:SPARQLConstraints;
        sh:severity iff:severityCritical ;
        sh:message "Cutter running without running filter" ;
        sh:select """
PREFIX iff: <https://industry-fusion.com/types/v0.9/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT $this ?v1 ?f ?v2
    where {
        $this a iff:cutter .
        $this  iff:state [<https://uri.etsi.org/ngsi-ld/hasValue> ?v1 ] .
        $this  iff:hasFilter [ <https://uri.etsi.org/ngsi-ld/hasObject> ?f ] .
        ?f  iff:state [ <https://uri.etsi.org/ngsi-ld/hasValue> ?v2 ] .
        FILTER(?v1 = <https://industry-fusion.com/types/v0.9/state_ON> &&
            ?v2 != <https://industry-fusion.com/types/v0.9/state_ON>)

    }
        """ ;
    ] .
```

Validates

- Every Entity of type ‘cutter’
- If cutter is running, whether linked filter runs, too

Constraints & Rules: Shapes Constraint Language (SHACL)

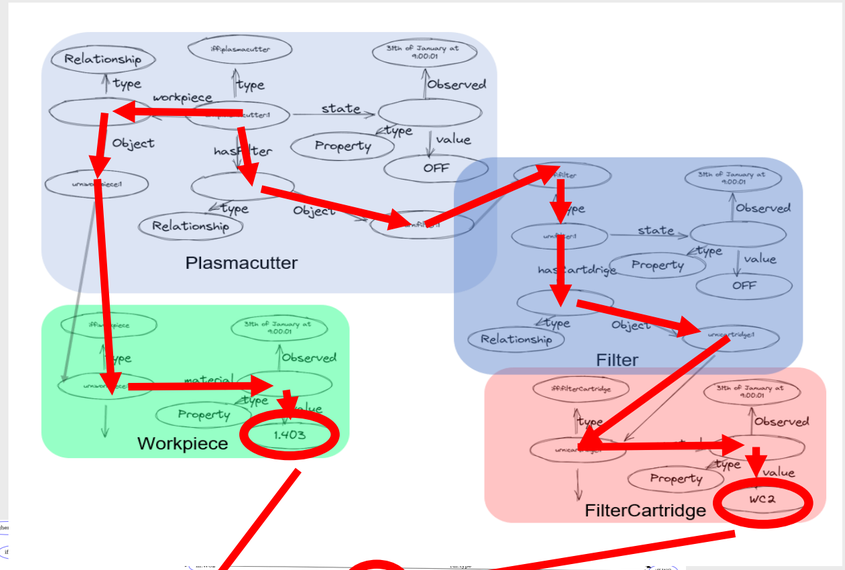


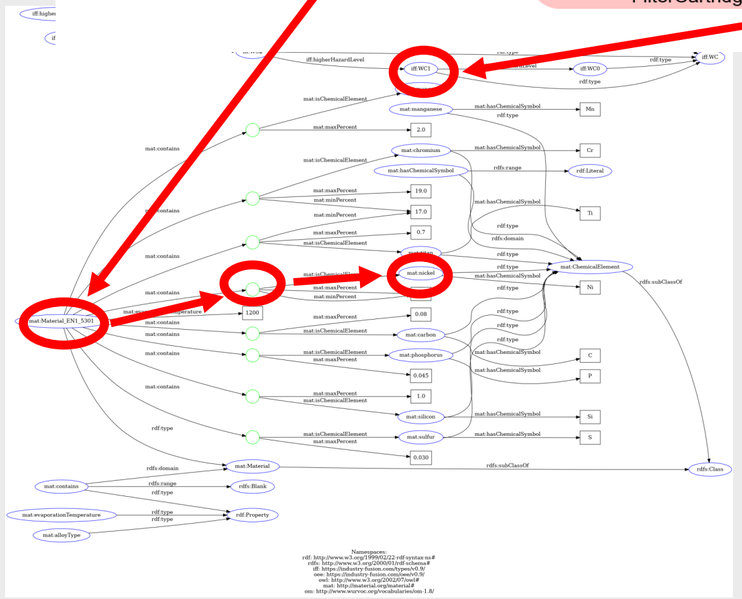

```
iff:WarnIfCartridgeClassChanges
    a sh:NodeShape ;
    sh:targetClass iff:cutter ;
    sh:sparql [
        a sh:SPARQLConstraints;
        sh:severity iff:severityCritical ;
        sh:message "Cutter running without running filter" ;
        sh:select """
PREFIX iff: <https://industry-fusion.com/types/v0.9/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT $this ?v1 ?f ?v2
    where {
        $this a iff:cutter .
        $this  iff:hasFilter [ <https://uri.etsi.org/ngsi-ld/hasObject> ?f ] .
        $this  iff:hasWorkpiece [ <https://uri.etsi.org/ngsi-ld/hasObject> ?w ] .
        ?f  iff:hasCartridge [ <https://uri.etsi.org/ngsi-ld/hasObject> ?c ] .
        ...
    }
        """ ;
    ] .
```

Validates

- If material type of workpiece is changing the hazard level in the cartridge wasteclass
- Needs to link information from Entities AND Knowledge Graph

Put it all together: Structured Entities, Knowledge & Constraints/Rules






```
iff:WarnIfCartridgeClassChanges
    a sh:NodeShape ;
    sh:targetClass iff:cutter ;
    sh:sparql [
        a sh:SPARQLConstraints;
        sh:severity iff:severityCritical ;
        sh:message "Cutter running without running filter" ;
        sh:select """
PREFIX iff: <https://industry-fusion.com/types/v0.9/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT $this ?v1 ?f ?v2
    where {
        $this a iff:cutter .
        $this  iff:hasFilter [ <https://uri.etsi.org/ngsi-ld/hasObject> ?f ] .
        $this  iff:hasWorkpiece [ <https://uri.etsi.org/ngsi-ld/hasObject> ?w ] .
        ?f  iff:hasCartridge [ <https://uri.etsi.org/ngsi-ld/hasObject> ?c ] .
        ...
    }
        """ ;
    ] .
```

Validates

- If material type of workpiece is changing the hazard level in the cartridge wasteclass
- Needs to link information from Entities AND Knowledge Graph

Summary: Three Pillars of Semantic Validation

- **Structured Entities**
  - JSON-LD enables the representation of entities and their relationships through linked, structured data.
- **Ontology and Knowledge Representation**
  - OWL (Web Ontology Language) forms the foundation of knowledge graphs, modeling complex relationships and domain knowledge.
- **Rules and Validation**
  SHACL (Shapes Constraint Language) defines and enforces constraints and rules to ensure semantic data quality and consistency.

Example: OPC UA Validation

Translate OPC UA to a semantic description and validate it

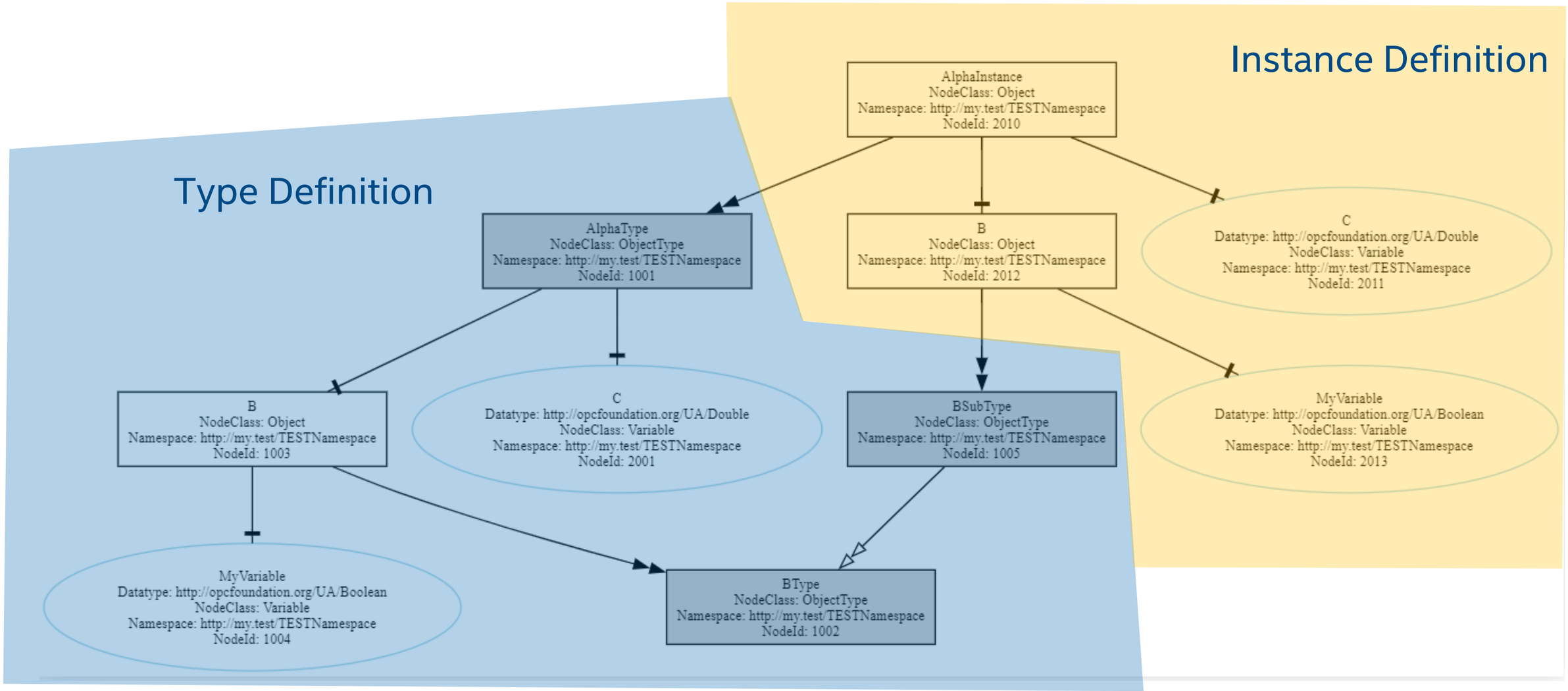
Problem Statement

OPC UA Evolution into a System Layer

- **Increasing Complexity**: OPC UA is transitioning from isolated machine communication to a comprehensive system layer by integrating multiple Companion Specifications (CS).
- **Lack of Standardization**: Data is streamed to the cloud without a standardized semantic or serialization layer, leading to potential inconsistencies.
- **Proprietary Specifications**: Modern OPC UA systems incorporate many Companion Specifications, with several being proprietary, complicating interoperability.
- **Informal Rules**: Many rules and constraints in CSs are described verbally rather than formally, making them difficult to enforce and validate.
- **Validation Challenges**:
  - **Compliance and Consistency**: Who ensures the compliance and consistency of instances?
  - **Complexity and Decidability**: Who addresses the unintended complexity and decidability issues arising from the combination of specifications?

Solution Needed: Translation to an expressive semantic layer to automate validation, assess complexity pitfalls and ensure system integrity.

Simple OPC UA Example



Translation to Semantic Web

Type Definition:
SHACL(Constraints, Rules) & OWL(Ontology)

```
shacl:AlphaTypeShape a sh:NodeShape ;
    sh:property [ a base:SubComponentRelationship ;
            sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasB> ;
            sh:property [ sh:class test:BType ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:IRI ;
                    sh:path ngsi-ld:hasObject ] ],
        [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasC> ;
            sh:property [ sh:datatype xsd:double ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:AlphaType .

shacl:BTypeShape a sh:NodeShape ;
    sh:property [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasMyVariable> ;
            sh:property [ sh:datatype xsd:boolean ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:BType .
```

```
:BSubType a owl:Class ;
rdfs:subClassOf test:BType .

tity: a owl:Ontology ;
owl:versionInfo 1e-01 .

tity:hasB a owl:NamedIndividual,
    owl:ObjectProperty ;
rdfs:domain test:AlphaType .

tity:hasC a owl:NamedIndividual,
    owl:ObjectProperty ;
rdfs:domain test:AlphaType .

tity:hasMyVariable a owl:NamedIndividual,
    owl:ObjectProperty ;d:\entities.ttl
rdfs:domain test:BType .

:BType a owl:Class ;
rdfs:subClassOf opcua:BaseObjectType .
```

Instance Definition:
JSON-LD (NGSI-LD)

```
[
    {
        "type": "http://my.test/BSubType",
        "id": "urn:test:AlphaInstance:sub:i2012",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasMyVariable": {
            "type": "Property",
            "value": false
        }
    },
    {
        "type": "http://my.test/AlphaType",
        "id": "urn:test:AlphaInstance",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasC": {
            "type": "Property",
            "value": 0.0
        },
        "uaentity:hasB": [
            {
                "type": "Relationship",
                "object": "urn:test:AlphaInstance:sub:i2012"
            }
        ]
    }
]
```

Validation – FAIL without Ontology (There is no relationship between BSubType and Btype)

SHACL(Constraints, Rules) & OWL(Ontology)

```
shacl:AlphaTypeShape a sh:NodeShape ;
    sh:property [ a base:SubComponentRelationship ;
            sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasB> ;
            sh:property [ sh:class test:BType ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:IRI ;
                    sh:path ngsi-ld:hasObject ] ],
        [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasC> ;
            sh:property [ sh:datatype xsd:double ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:AlphaType .

shacl:BTypeShape a sh:NodeShape ;
    sh:property [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasMyVariable> ;
            sh:property [ sh:datatype xsd:boolean ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:BType .
```

```
test:BSubType a owl:Class ;
    rdfs:subClassOf test:BType .

uaentity: a owl:Ontology ;
    owl:versionInfo 1e-01 .

uaentity:hasB a owl:NamedIndividual,
        owl:ObjectProperty ;
    rdfs:domain test:AlphaType .

uaentity:hasC a owl:NamedIndividual,
        owl:ObjectProperty ;
    rdfs:domain test:AlphaType .

uaentity:hasMyVariable a owl:NamedIndividual,
        owl:ObjectProperty ;d:\entities.ttl
    rdfs:domain test:BType .

test:BType a owl:Class ;
    rdfs:subClassOf opcua:BaseObjectType .
```

JSON-LD (NGSI-LD)

```
[
    {
        "type": "http://my.test/BSubType",
        "id": "urn:test:AlphaInstance:sub:i2012",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasMyVariable": {
            "type": "Property",
            "value": false
        }
    },
    {
        "type": "http://my.test/AlphaType",
        "id": "urn:test:AlphaInstance",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasC": {
            "type": "Property",
            "value": 0.0
        },
        "uaentity:hasB": [
            {
                "type": "Relationship",
                "object": "urn:test:AlphaInstance:sub:i2012"
            }
        ]
    }
]
```

Validation – Success with Ontology (Understand that BSubType is subclass of BType)

SHACL(Constraints, Rules) & OWL(Ontology)

```
shacl:AlphaTypeShape a sh:NodeShape ;
    sh:property [ a base:SubComponentRelationship ;
            sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasB> ;
            sh:property [ sh:class test:BType ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:IRI ;
                    sh:path ngsi-ld:hasObject ] ],
        [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasC> ;
            sh:property [ sh:datatype xsd:double ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:AlphaType .

shacl:BTypeShape a sh:NodeShape ;
    sh:property [ sh:maxCount 1 ;
            sh:minCount 0 ;
            sh:nodeKind sh:BlankNode ;
            sh:path <http://my.test/entity/hasMyVariable> ;
            sh:property [ sh:datatype xsd:boolean ;
                    sh:maxCount 1 ;
                    sh:minCount 1 ;
                    sh:nodeKind sh:Literal ;
                    sh:path ngsi-ld:hasValue ] ] ;
    sh:targetClass test:BType .
```

```
test:BSubType a owl:Class ;
    rdfs:subClassOf test:BType .

uaentity: a owl:Ontology ;
    owl:versionInfo 1e-01 .

uaentity:hasB a owl:NamedIndividual,
        owl:ObjectProperty ;
    rdfs:domain test:AlphaType .

uaentity:hasC a owl:NamedIndividual,
        owl:ObjectProperty ;
    rdfs:domain test:AlphaType .

uaentity:hasMyVariable a owl:NamedIndividual,
        owl:ObjectProperty ;d:\entities.ttl
    rdfs:domain test:BType .

test:BType a owl:Class ;
    rdfs:subClassOf opcua:BaseObjectType .
```

JSON-LD (NGSI-LD)

```
[
    {
        "type": "http://my.test/BSubType",
        "id": "urn:test:AlphaInstance:sub:i2012",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasMyVariable": {
            "type": "Property",
            "value": false
        }
    },
    {
        "type": "http://my.test/AlphaType",
        "id": "urn:test:AlphaInstance",
        "@context": [
            "http://context.source/context.jsonld"
        ],
        "uaentity:hasC": {
            "type": "Property",
            "value": 0.0
        },
        "uaentity:hasB": [
            {
                "type": "Relationship",
                "object": "urn:test:AlphaInstance:sub:i2012"
            }
        ]
    }
]
```

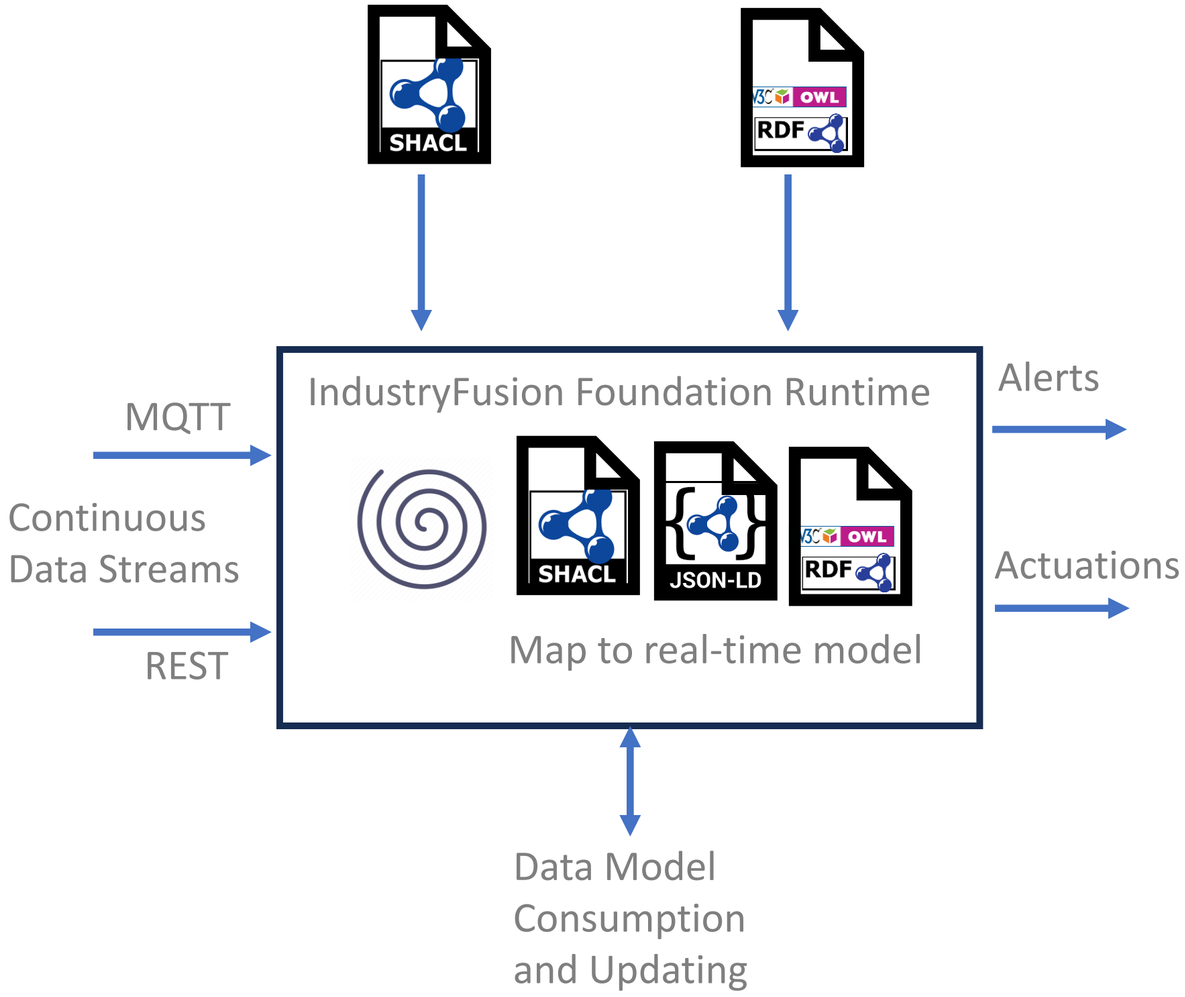
From Static to Runtime validation

Challenges & Solution Space

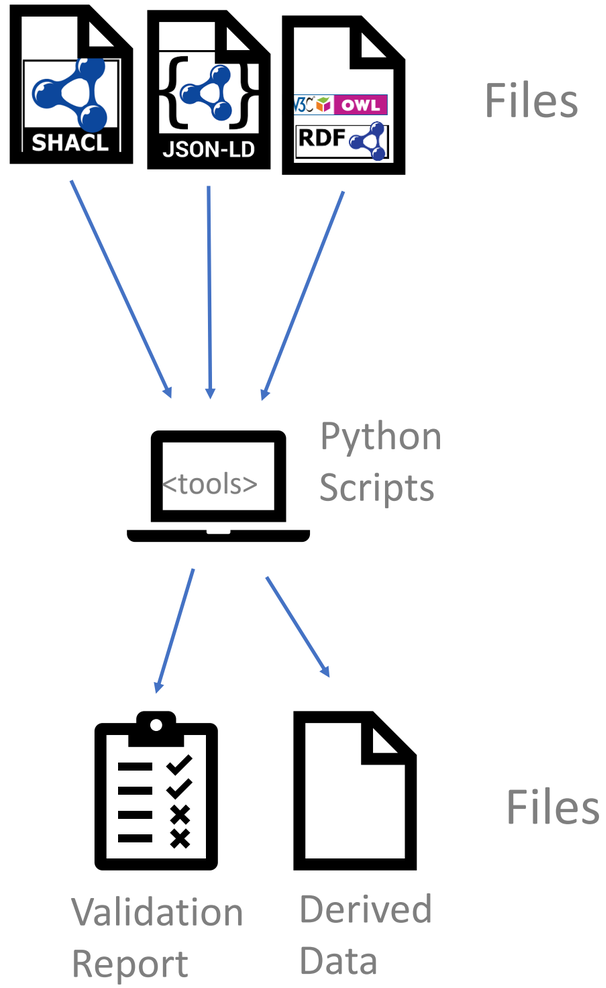
IndustryFusion FOUNDATION

Static and Continous Validation Both Possible With Same Foundation

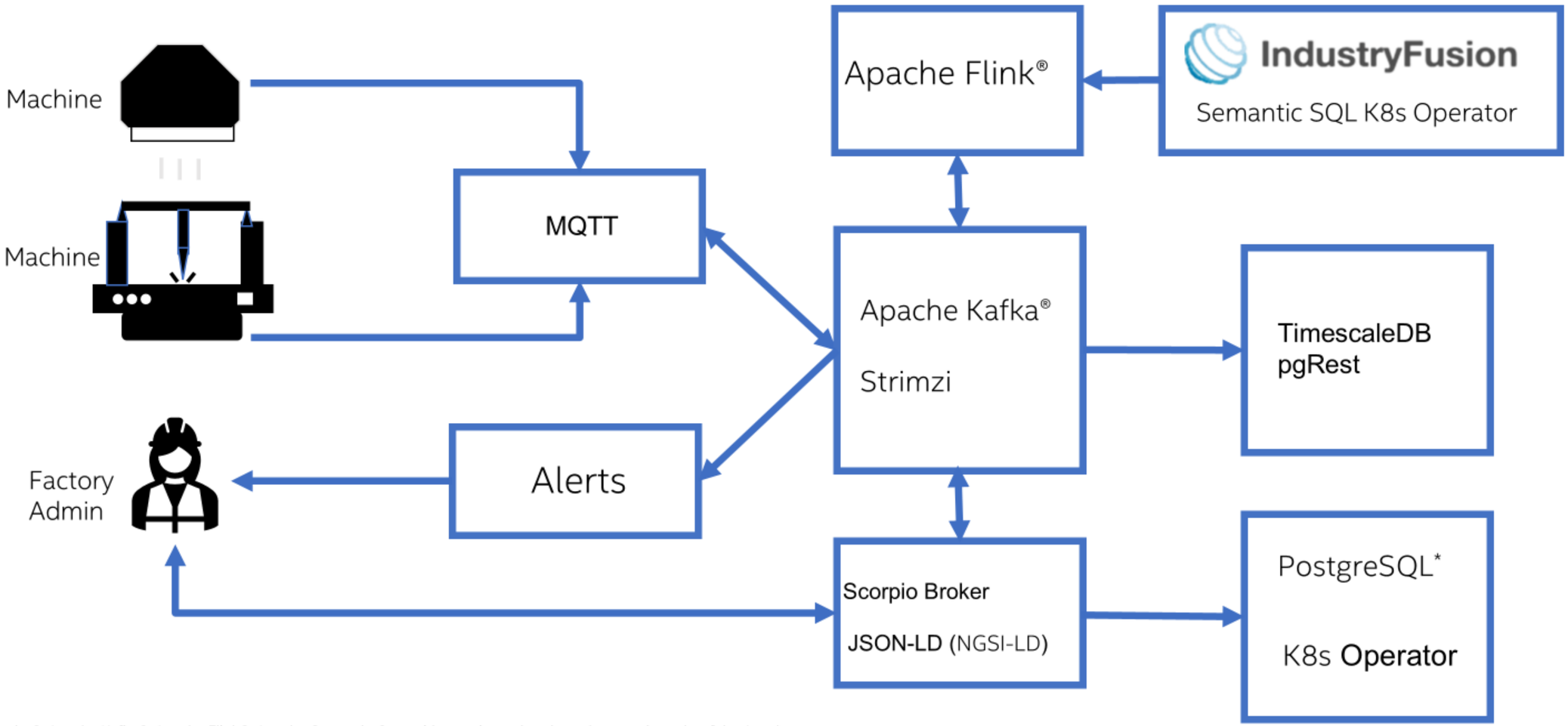
Continuous Validation



Static Validation



IndustryFusion Architectural Overview



Conclusion

- **Trust Through Validation**: Building trust in systems requires rigorous data validation processes.
- **Define Semantics Clearly**: Validation needs a clear defined semantics
- **Powerful Semantic Tools**: JSON-LD, OWL, and SHACL offer the semantic expressiveness needed to build trust and reliability.
- **Robust Modeling and Validation**: systems can be accurately modeled and validated both offline and in real-time using semantic data.
- **Unified Data Integration**: A Semantic Data Layer can seamlessly integrate diverse data sources, serving as a robust and scalable data layer beyond individual machines.
- **Standardized and Open-Source:** Built upon standards, open source and utilizing best practices in real-time data analytics as part of the IndustryFusion project. Visit: https://github.com/IndustryFusion/DigitalTwin

Backup

AI in Operation is only possible with Semantic Data


White Paper

Manufacturing
Smart Manufacturing
Digital Manufacturing

Guideline for operating resilient and flexible production facilities using runtime risk management

Proposal of a new safety paradigm: operational safety intelligence - optimizing operations by runtime risk management and predictive modeling.

1 Abstract 02

Highlights:

- Semantic Data forms the Backbone of operational Safety Intelligence
- AI based decisions and recommendations must be safeguarded by data understanding
- Semantic Data allows formal validation and verification

Legal Notices and Disclaimers