AIOTI 2026
March 24, 2026

Robert David

SHACL (R|r)ules Interoperability

Enabling cross-model interoperability with rule-based reasoning

Vladimir Alexiev, Robert David

GRAPHWISE

AI THRIVES ON WHOLE DATA

Applications built on Graphs+GenAI

GRAPHWISE
AI THRIVES ON WHOLE DATA

Bringing Confidence to AI

LLM Grounding with **Multimodal Data**

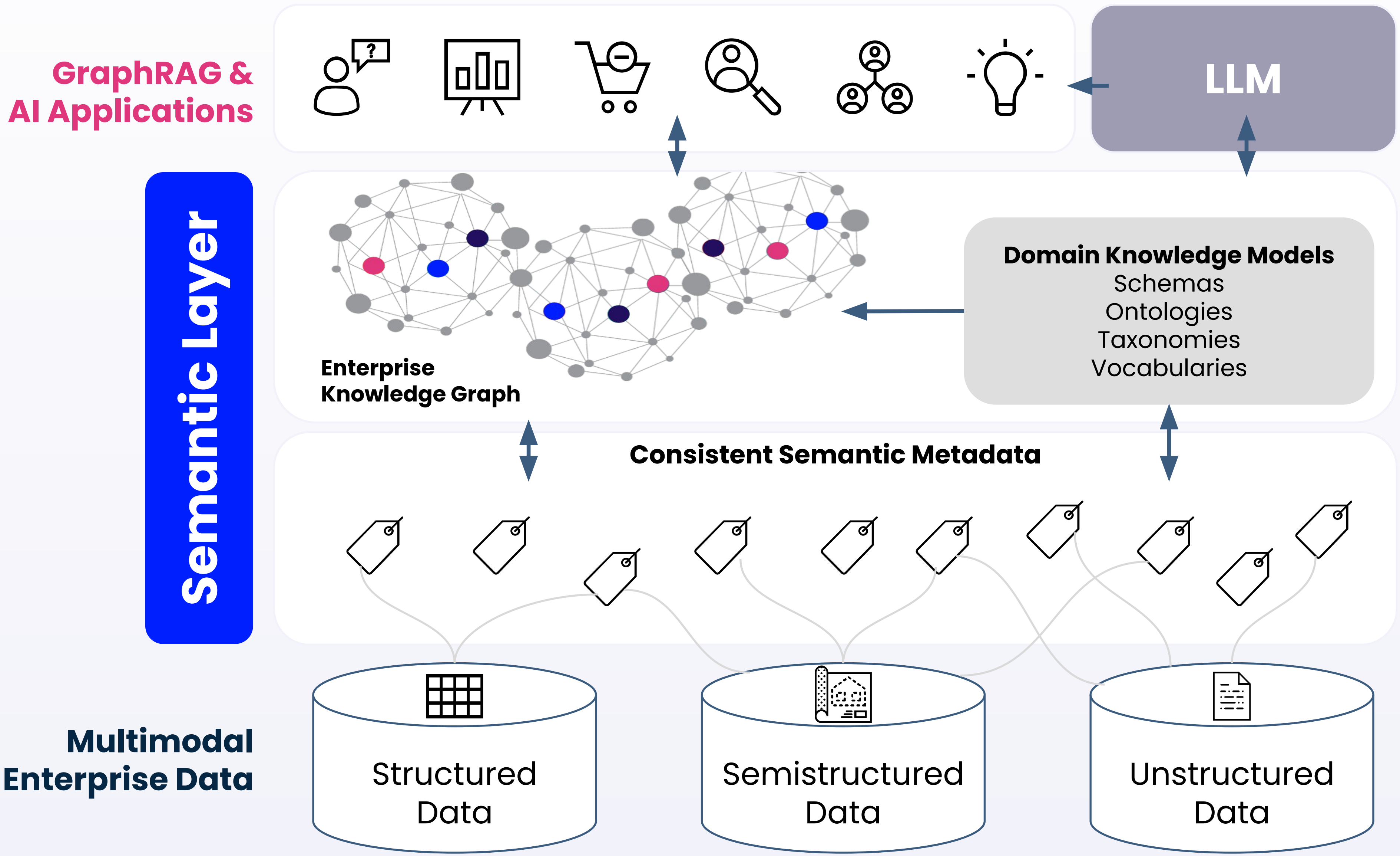
Self-Service Data & Knowledge

Enhance AI performance and optimize costs

Get **Accurate** Search & Recommendation

Semantic Layer for AI Applications

GRAPHWISE
AI THRIVES ON WHOLE DATA



Models for DPP & Initiatives

GRAPHWISE
AI THRIVES ON WHOLE DATA

- Digital Product Passports (DPPs) are a core element of the EU circular economy strategy.
- DPPs are intended as digital records containing essential data on a product’s properties.
- Several data models were developed
  - Asset Administration Shell (AAS)
  - United Nations Transparency Protocol (UNTP)
  - RDF and Ontologies (e.g. CEON)
- They server different purposes and use different modeling schemes.

Models for DPP & Interoperability

GRAPHWISE
AI THRIVES ON WHOLE DATA

- Different models use different underlying (semantic) representations.
  - Asset Administration Shell (AAS)
    - SAMM - RDF representation of AAS models
  - United Nations Transparency Protocol (UNTP)
    - JSON-LD - JSON as a common data format + LD as an RDF representation
  - RDF and Ontologies (e.g. CEON)
    - native RDF + ontologies formalise the semantics of a domain
- All models make sense and there is no unified model to replace them.
- Still, managing DPP EU-wide and beyond requires interoperability of all these models.

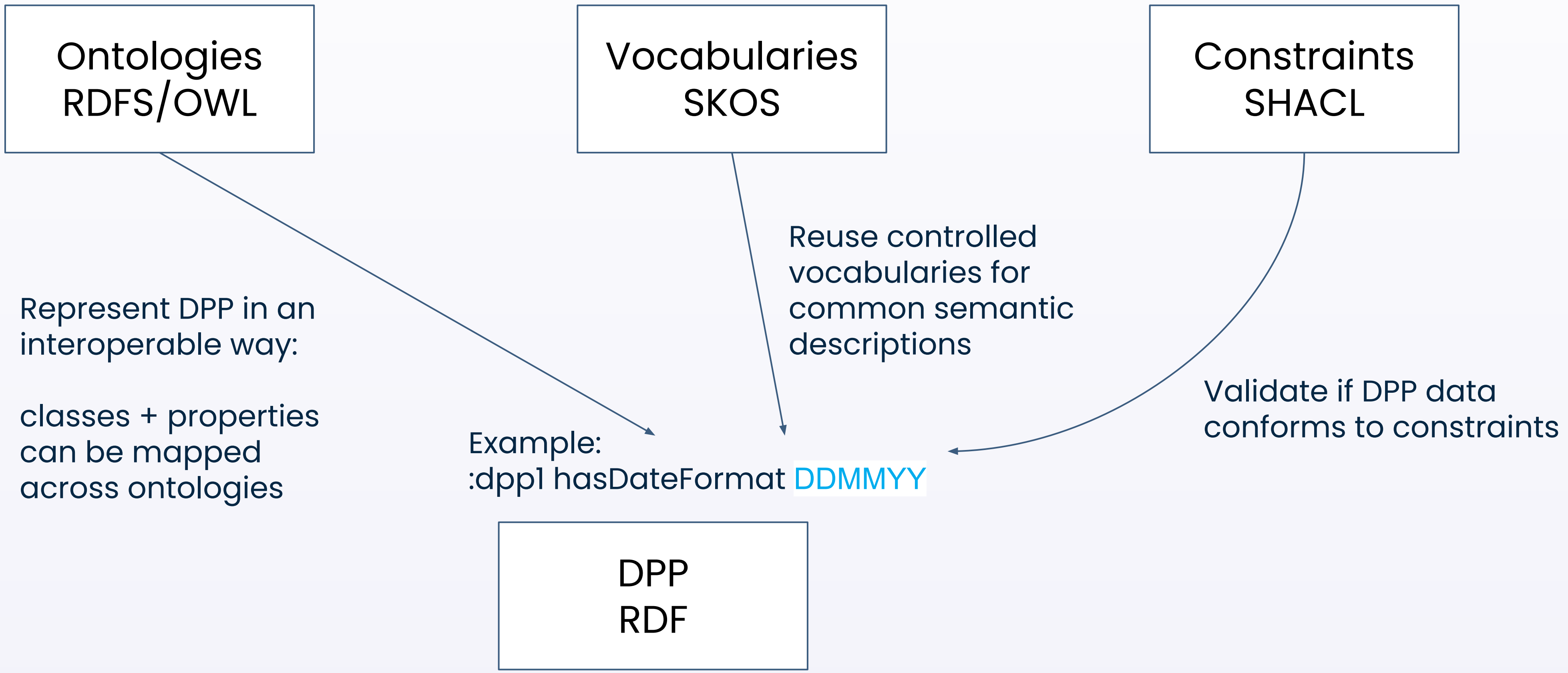
We need interoperability between all models using a unified approach.

Interoperability via Semantic Layer

- Semantic Standards enable Interoperability
  - RDF - basic graph representation of data - semantics via triples
  - RDFS - Schemas for RDF to express classes, properties, …
  - OWL - ontology language to express domain semantics
  - SKOS - vocabulary representation, e.g. for UNCL
  - SHACL - constraint language for RDF graphs
- These standards can be combined to
  - express domain semantics in an **interoperable** way
  - to **infer** knowledge from data
  - to **validate** if the data is correct

Standards for interoperable DPPs

GRAPHWISE
AI THRIVES ON WHOLE DATA



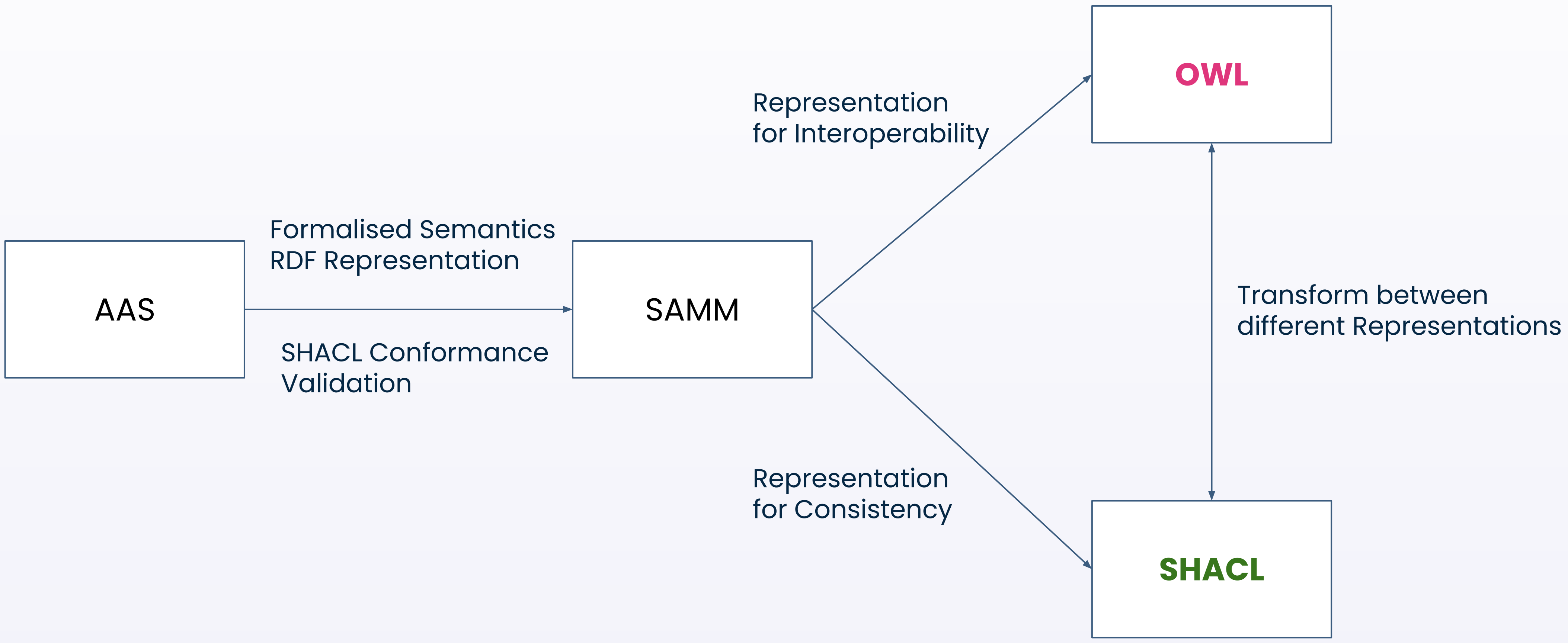
Standards for interoperable DPPs

GRAPHWISE
AI THRIVES ON WHOLE DATA

AAS

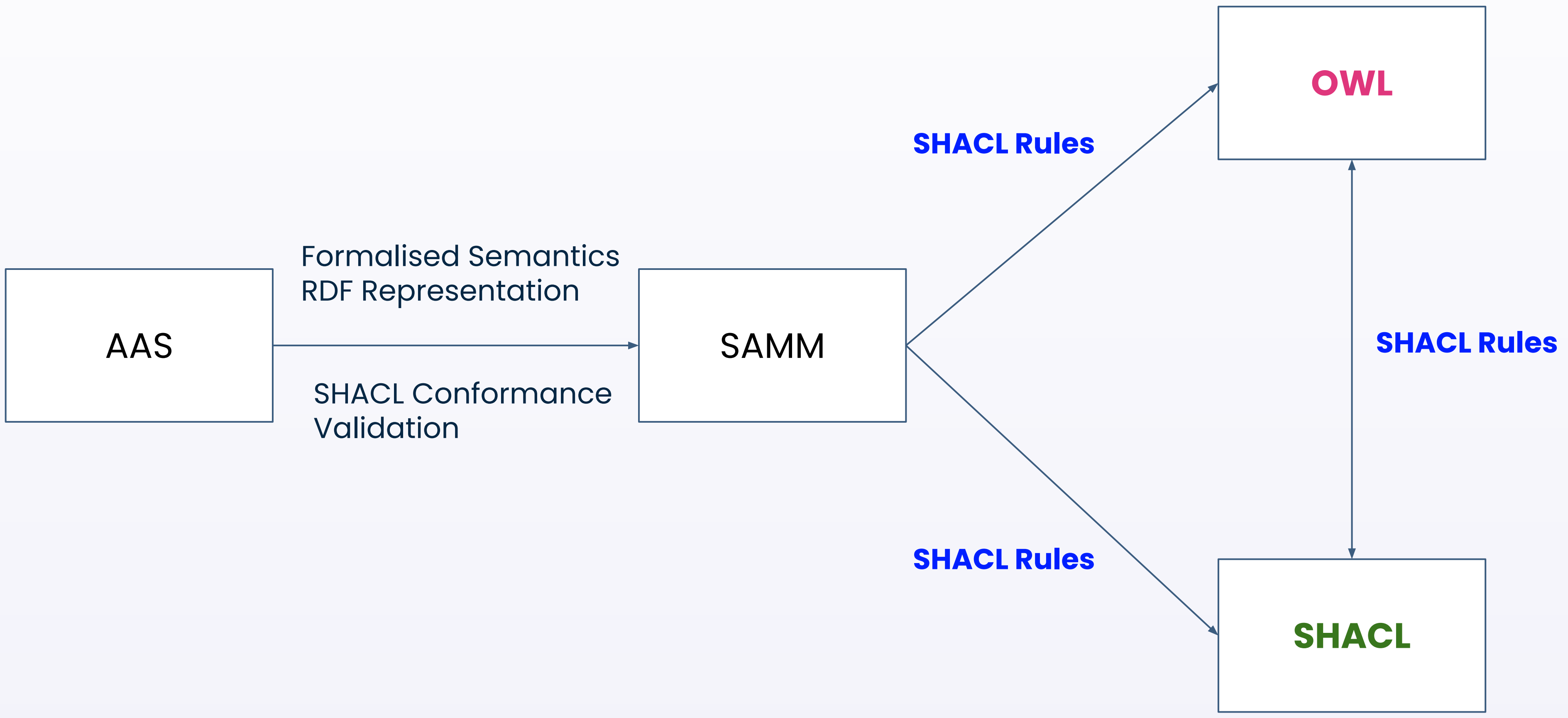
Standards for interoperable DPPs

Standards for interoperable DPPs



Standards for interoperable DPPs

GRAPHWISE
AI THRIVES ON WHOLE DATA



Transformations via SHACL Rules

- SHACL 1.2
  - Upcoming new version of the SHACL W3C recommendation
  - Beyond validation, add new sub-modules for node expressions, UI, Profiles and Rules
  - General toolkit for data validation, inference and transformation
- SHACL 1.2 Rules
  - based on Datalog, RDF adaption of rule language
  - not coupled to validation; can be used standalone
  - supports non-monotonic reasoning with recursion, negation, aggregation

Example: SAMM

GRAPHWISE
AI THRIVES ON WHOLE DATA

```
:MachineOnlineStatus a samm:Aspect ;
    samm:properties ( :onlineStatus ) ;
    samm:operations ( ) .

:onlineStatus a samm:Property ;
    samm:description "The current machine online status."@en ;
    samm:characteristic [
        a samm-c:Enumeration ;
        samm:dataType xsd:string ;
        samm-c:values ( "ON" "OFF" "BOOTING" ) ] .
```

Example: SAMM Triples

GRAPHWISE
AI THRIVES ON WHOLE DATA

:MachineOnlineStatus rdf:type samm:Aspect

:MachineOnlineStatus samm:properties _:N003a204d285f4918bfac75aaf00fee3b

_:N003a204d285f4918bfac75aaf00fee3b rdf:first :onlineStatus

_:N003a204d285f4918bfac75aaf00fee3b rdf:rest rdf:nil

:onlineStatus rdf:type samm:Property

:onlineStatus samm:characteristic _:N15fee727b0b6465da147a6639dc9ec0a

_:N15fee727b0b6465da147a6639dc9ec0a samm:dataType xsd:String

_:N15fee727b0b6465da147a6639dc9ec0a samm:values _:N6bade8ba119d4834b690ae6c16b18aef

_:N6bade8ba119d4834b690ae6c16b18aef rdf:first "ON"

_:N6bade8ba119d4834b690ae6c16b18aef rdf:rest _:Nd679068094854026a7473a43e6281a40

_:Nd679068094854026a7473a43e6281a40 rdf:first "OFF"

.....

Example: OWL

```
:MachineOnlineStatus a owl:Class .

:onlineStatus a owl:DatatypeProperty ;
        rdfs:comment "The current machine online status."@en ;
        rdfs:domain :MachineOnlineStatus ;
        rdfs:range :OnlineStatusValues ;

:OnlineStatusValues a rdfs:Datatype ;
        owl:oneOf ( "ON"^^xsd:string "OFF"^^xsd:string "BOOTING"^^xsd:string ) .
```

Example: SHACL Constraints

```
:MachineOnlineStatusShape a sh:NodeShape ;
    sh:class :MachineOnlineStatus ;
    sh:property [
        sh:path: :onlineStatus;
        sh:description "The current machine online status."@en ;
        sh:datatype xsd:string ;
        sh:in ( "ON"^^xsd:string "OFF"^^xsd:string "BOOTING"^^xsd:string )
    ] .
```

Example: Rule Transformation

- SHACL 1.2 Datalog-style Rules
  https://w3c.github.io/data-shapes/shacl12-rules/
  - 3 basic rule formats
    - DATA { *triple template* }
    - RULE { *head template* } WHERE { *body pattern* }
    - IF { *body pattern* } THEN { *head template* }

Examples:

RULE { ?x rdf:type owl:Class . } WHERE { ?x rdf:type samm:Aspect . }

RULE { ?x rdf:type owl:DatatypeProperty . }

WHERE { ?x rdf:type samm:Property . ?x samm:characteristic ?c . ?c samm:dataType xsd:string . }

Example: Rule Transformation

GRAPHWISE
AI THRIVES ON WHOLE DATA

Input:

:onlineStatus rdf:type samm:Property

:onlineStatus samm:characteristic _:N15fee727b0b6465da147a6639dc9ec0a

_:N15fee727b0b6465da147a6639dc9ec0a samm:dataType xsd:String

Rule:

RULE { ?x rdf:type owl:DatatypeProperty . }

WHERE { ?x rdf:type samm:Property . ?x samm:characteristic ?c . ?c samm:dataType xsd:string . }

Output:

:onlineStatus rdf:type owl:DatatypeProperty

SHACL 1.2 Rules: pros & cons

- Pro:
  - General declarative language for data transformations
  - Works on any RDF graph
  - Supports non-monotonic inference with recursion, negation, aggregation
  - Semantics defined in the W3C spec
- Cons:
  - No disjunction in rules and stratified negation, aggregation; therefore no multiple models as result
  - Difficult to represent error handling in rules; no empty model on non-match

SHACL (R|r)ules Interoperability

- SHACL 1.2 Rules provide a declarative approach to transform between different models and representations.
- SHACL validation helps to avoid quality problems with transformations.
- Rules and constraints combine into a human-in-the-loop approach to develop model interoperability.

THANK YOU