gaia-x

Gaia-X SUMMIT 2024

EMPOWERING GLOBAL DATA SPACES

SHAPING TOMORROW'S CLOUD INFRASTRUCTURE

Helsinki, Finland | 14 & 15 November

In partnership with gaia-x

🇫🇮 Hub Finland

Gaia-X Digital Clearing Houses

Christophe Strnadl - Ewann Gavard

Chief Technology Officer - Technical Lead

Gaia-X Association for Data and Cloud AISBL

What is a Gaia-X Digital Clearing House (GXDCH) ?

- They represent the operationalisation of the Gaia-X TrustFramework
- The AISBL does not want to centralize validation, verification using the software developed by the lab by being the only host.

- The AISBL relies on partners who are committed to host the components and provide an access for free to the whole community.

What is a Gaia-X Digital Clearing House (GXDCH) ?

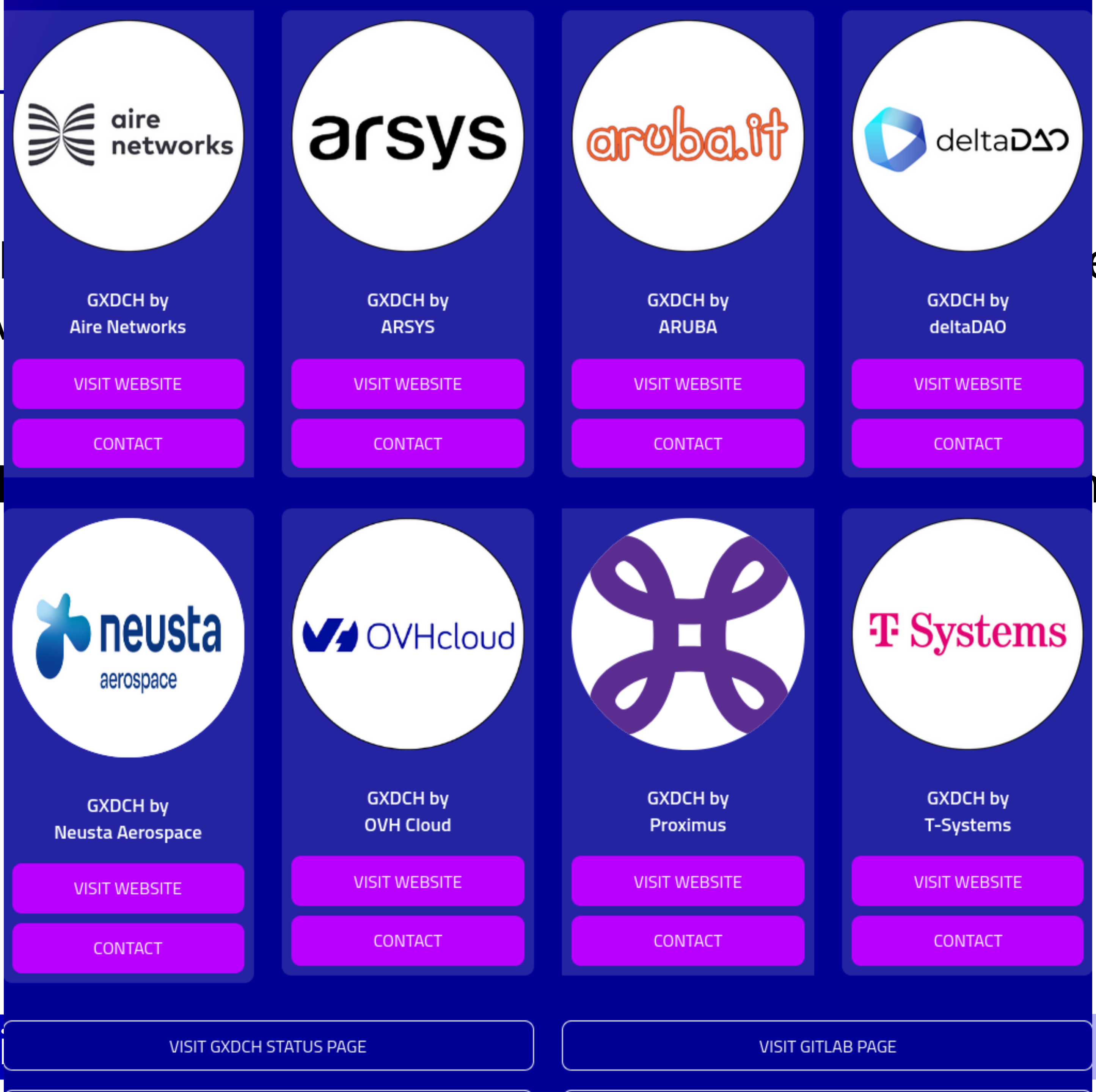
- These trusted hosts are called **G**aia-**X** **D**igital **C**learing **H**ouse Providers or GXDCH Providers

- 8 Clearing House running now and more to come

What is a Gaia-
(GXDCH) ?

- The trusted I   e Providers or GXDCH Prov

- 8 Clearing H   me



GXDCH by Aire Networks

GXDCH by ARSYS

GXDCH by ARUBA

GXDCH by deltaDAO

GXDCH by Neusta Aerospace

GXDCH by OVH Cloud

GXDCH by Proximus

GXDCH by T-Systems

How to become a clearing house ?

- Acquire a bit of knowledge regarding what it means to be a clearing house (software deployment, SLA) and about Gaia-X
- Get in touch with Mr Przemek Halub for the contractual part
- Deploy the components (Helm charts provided by the Lab)
- When the contract is signed, that Lab validates the deployment

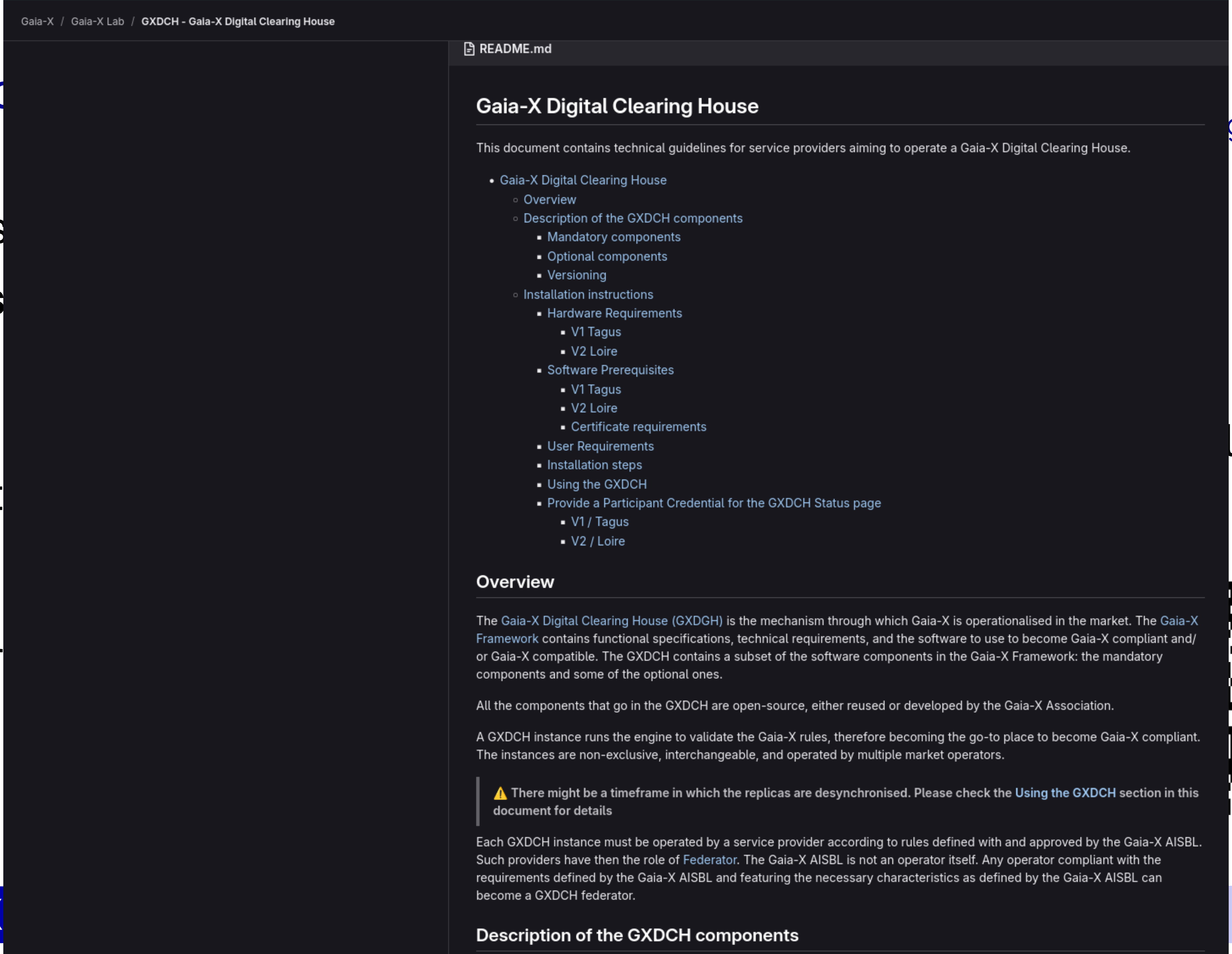
- Tada, you’re a GXDCH 🎉

Components of a GXDCH

- 2 versions to run in parallel, Tagus (v1) and Loire (v2)
- Tagus composed of Registry, Notary and Compliance
- Loire composed of Registry, Notary, Compliance and Credentials Events Service
- Installation instructions on gitlab.com/gaia-x/lab/gxdch

Compo

- 2 vers
- Tagus
- Loire ls
  Event
- Instal



Gaia-X / Gaia-X Lab / GXDCH - Gaia-X Digital Clearing House

README.md

Gaia-X Digital Clearing House

This document contains technical guidelines for service providers aiming to operate a Gaia-X Digital Clearing House.

- Gaia-X Digital Clearing House
  - Overview
  - Description of the GXDCH components
    - Mandatory components
    - Optional components
    - Versioning
  - Installation instructions
    - Hardware Requirements
      - V1 Tagus
      - V2 Loire
    - Software Prerequisites
      - V1 Tagus
      - V2 Loire
      - Certificate requirements
    - User Requirements
    - Installation steps
    - Using the GXDCH
    - Provide a Participant Credential for the GXDCH Status page
      - V1 / Tagus
      - V2 / Loire

Overview

The Gaia-X Digital Clearing House (GXDGH) is the mechanism through which Gaia-X is operationalised in the market. The Gaia-X Framework contains functional specifications, technical requirements, and the software to use to become Gaia-X compliant and/or Gaia-X compatible. The GXDCH contains a subset of the software components in the Gaia-X Framework: the mandatory components and some of the optional ones.

All the components that go in the GXDCH are open-source, either reused or developed by the Gaia-X Association.

A GXDCH instance runs the engine to validate the Gaia-X rules, therefore becoming the go-to place to become Gaia-X compliant. The instances are non-exclusive, interchangeable, and operated by multiple market operators.

> ⚠ There might be a timeframe in which the replicas are desynchronised. Please check the Using the GXDCH section in this document for details

Each GXDCH instance must be operated by a service provider according to rules defined with and approved by the Gaia-X AISBL. Such providers have then the role of Federator. The Gaia-X AISBL is not an operator itself. Any operator compliant with the requirements defined by the Gaia-X AISBL and featuring the necessary characteristics as defined by the Gaia-X AISBL can become a GXDCH federator.

Description of the GXDCH components

Components of GXDCH

- C4 Model Architecture diagrams available on Gitlab
- Model allowing to start from global picture and go into very deep technical details

Components of GXDCH

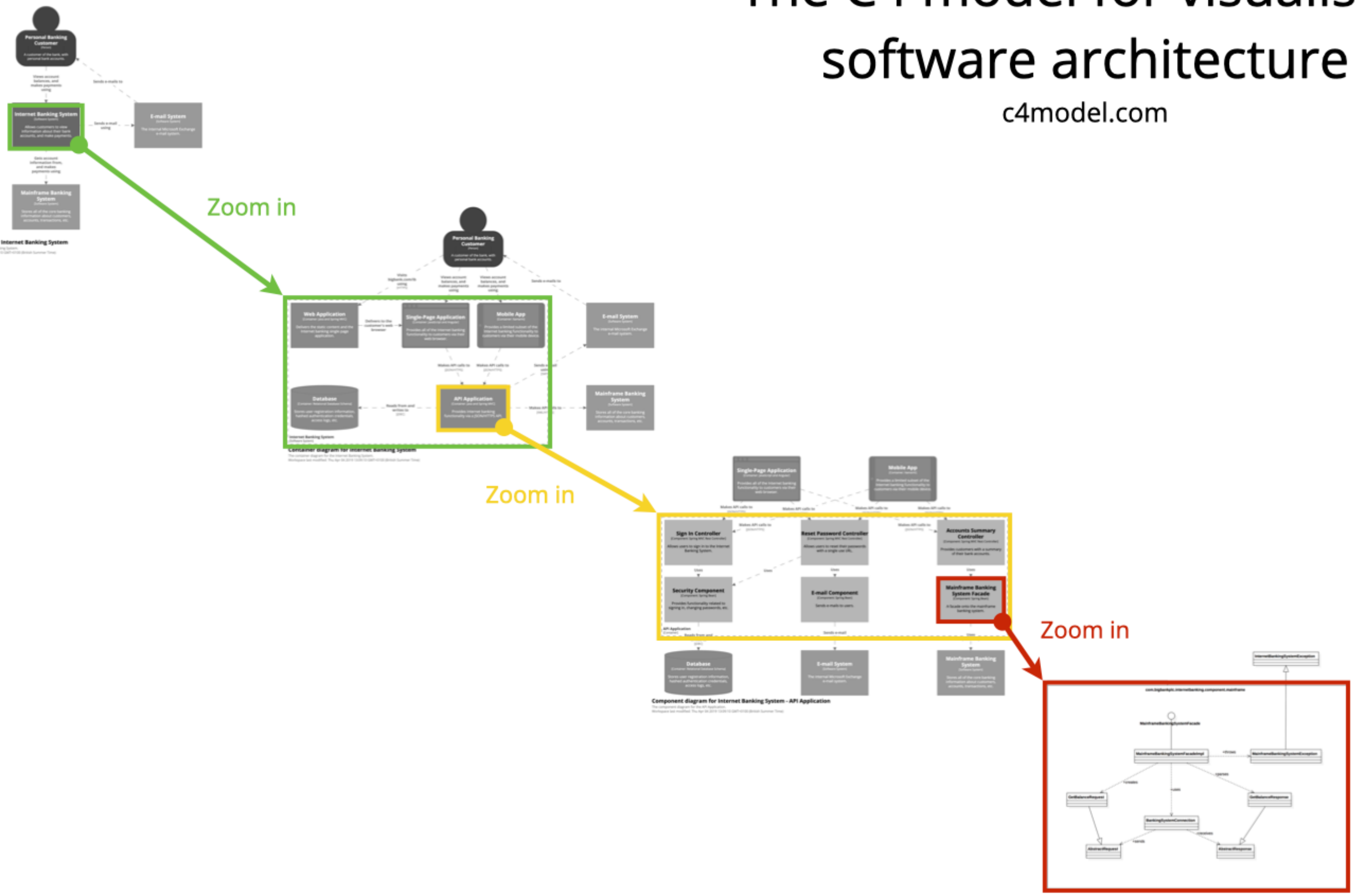
- C4 Model Architecture diagrams available on Gitlab
- Model allowing to start from global picture and go into very deep technical details

Com

- C4 I
- Moc tech

The C4 model for visualising software architecture

c4model.com

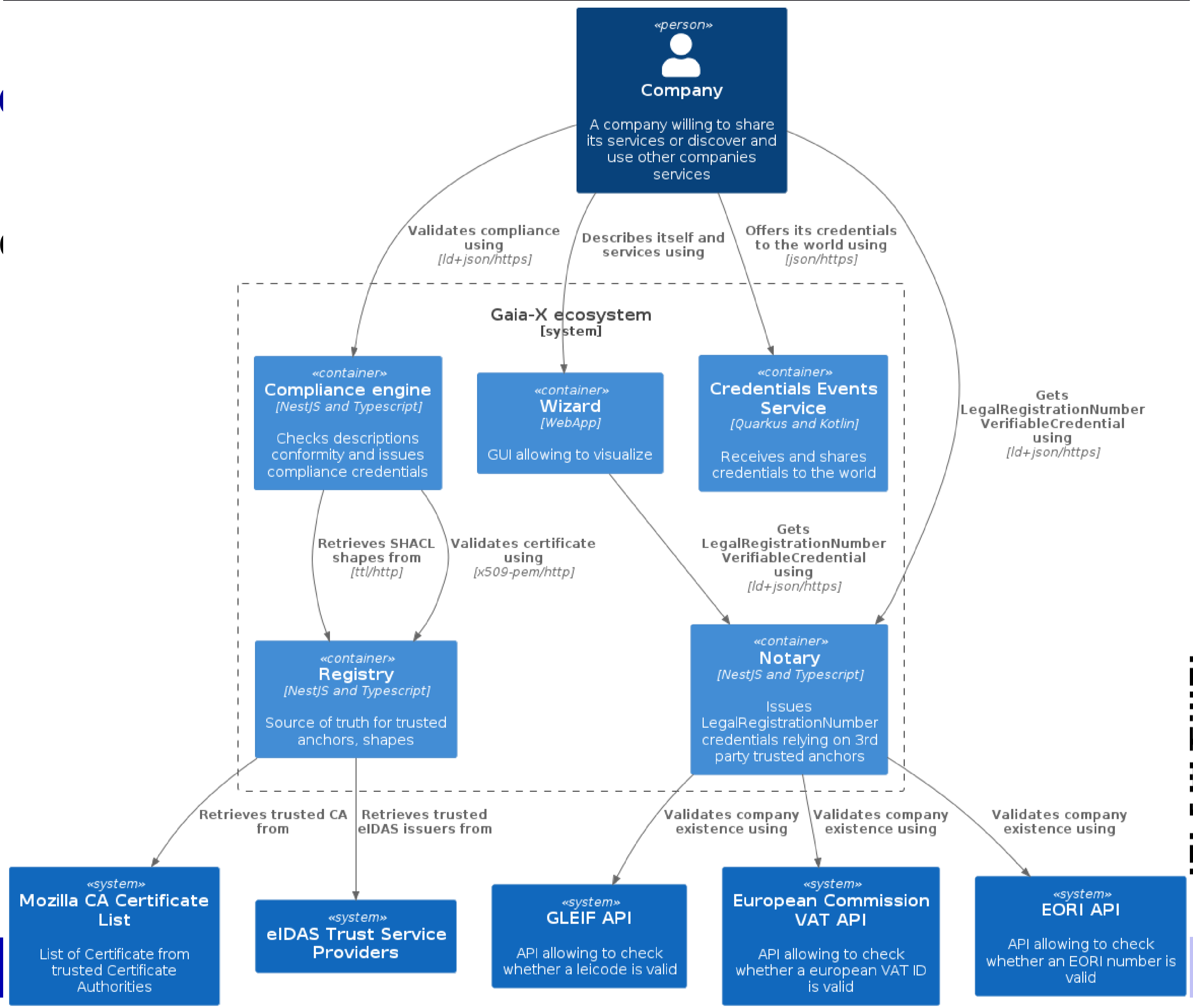


Level 1	Level 2	Level 3	Level 4
Context	Containers	Components	Code

#Gaia

Compon

- C4 Mod

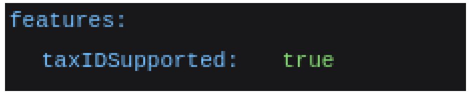


«person»
Company
A company willing to share its services or discover and use other companies services
Validates compliance using
[ld+json/https]
Describes itself and services using
Offers its credentials to the world using
[json/https]
Gets LegalRegistrationNumber VerifiableCredential using
[ld+json/https]
Gaia-X ecosystem
[system]
«container»
Compliance engine
[NestJS and Typescript]
Checks descriptions conformity and issues compliance credentials
«container»
Wizard
[WebApp]
GUI allowing to visualize
«container»
Credentials Events Service
[Quarkus and Kotlin]
Receives and shares credentials to the world
Retrieves SHACL shapes from
[ttl/http]
Validates certificate using
[x509-pem/http]
Gets LegalRegistrationNumber VerifiableCredential using
[ld+json/https]
«container»
Registry
[NestJS and Typescript]
Source of truth for trusted anchors, shapes
«container»
Notary
[NestJS and Typescript]
Issues LegalRegistrationNumber credentials relying on 3rd party trusted anchors
Retrieves trusted CA from
Retrieves trusted eIDAS issuers from
Validates company existence using
Validates company existence using
Validates company existence using
«system»
Mozilla CA Certificate List
List of Certificate from trusted Certificate Authorities
«system»
eIDAS Trust Service Providers
«system»
GLEIF API
API allowing to check whether a leicode is valid
«system»
European Commission VAT API
API allowing to check whether a european VAT ID is valid
«system»
EORI API
API allowing to check whether an EORI number is valid

#GaiaXSu

Notary

- Issue registration number credentials from trusted sources
  - EU Commission VAT API
  - EORI
  - GLEIF
  - OpenCorporate (on notary V2 with the flag enabled)



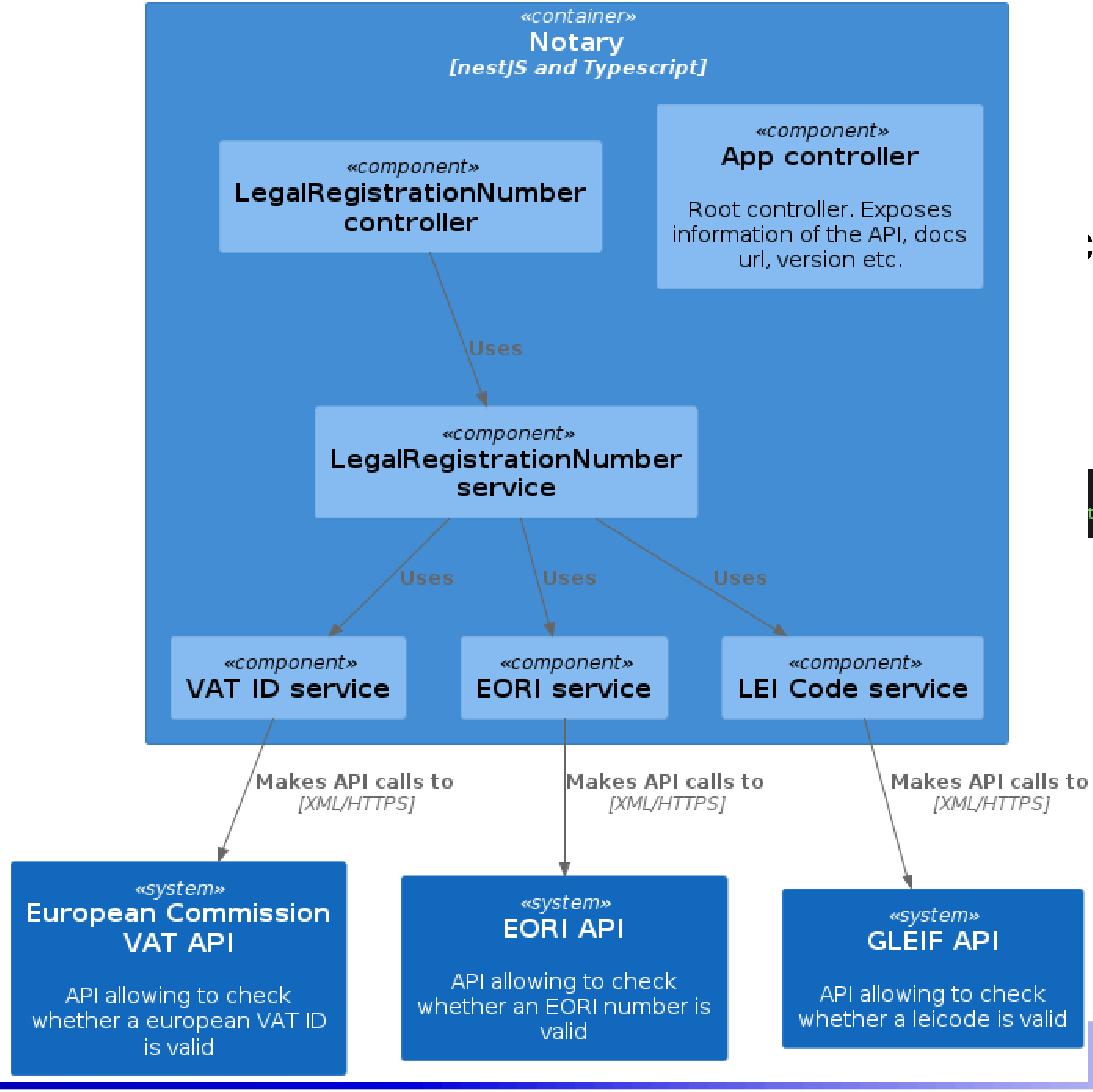
```
features:
  taxIDSupported:   true
```

- Proves existence of a LegalPerson

Notary

- Issue registra :es
  - EU Commis
  - EORI
  - GLEIF
  - OpenCorpc

- Proves existe


«container»
Notary
[nestJS and Typescript]

«component»
LegalRegistrationNumber controller

«component»
App controller

Root controller. Exposes information of the API, docs url, version etc.

«component»
LegalRegistrationNumber service

«component»
VAT ID service

«component»
EORI service

«component»
LEI Code service

«system»
European Commission VAT API

API allowing to check whether a european VAT ID is valid

«system»
EORI API

API allowing to check whether an EORI number is valid

«system»
GLEIF API

API allowing to check whether a leicode is valid

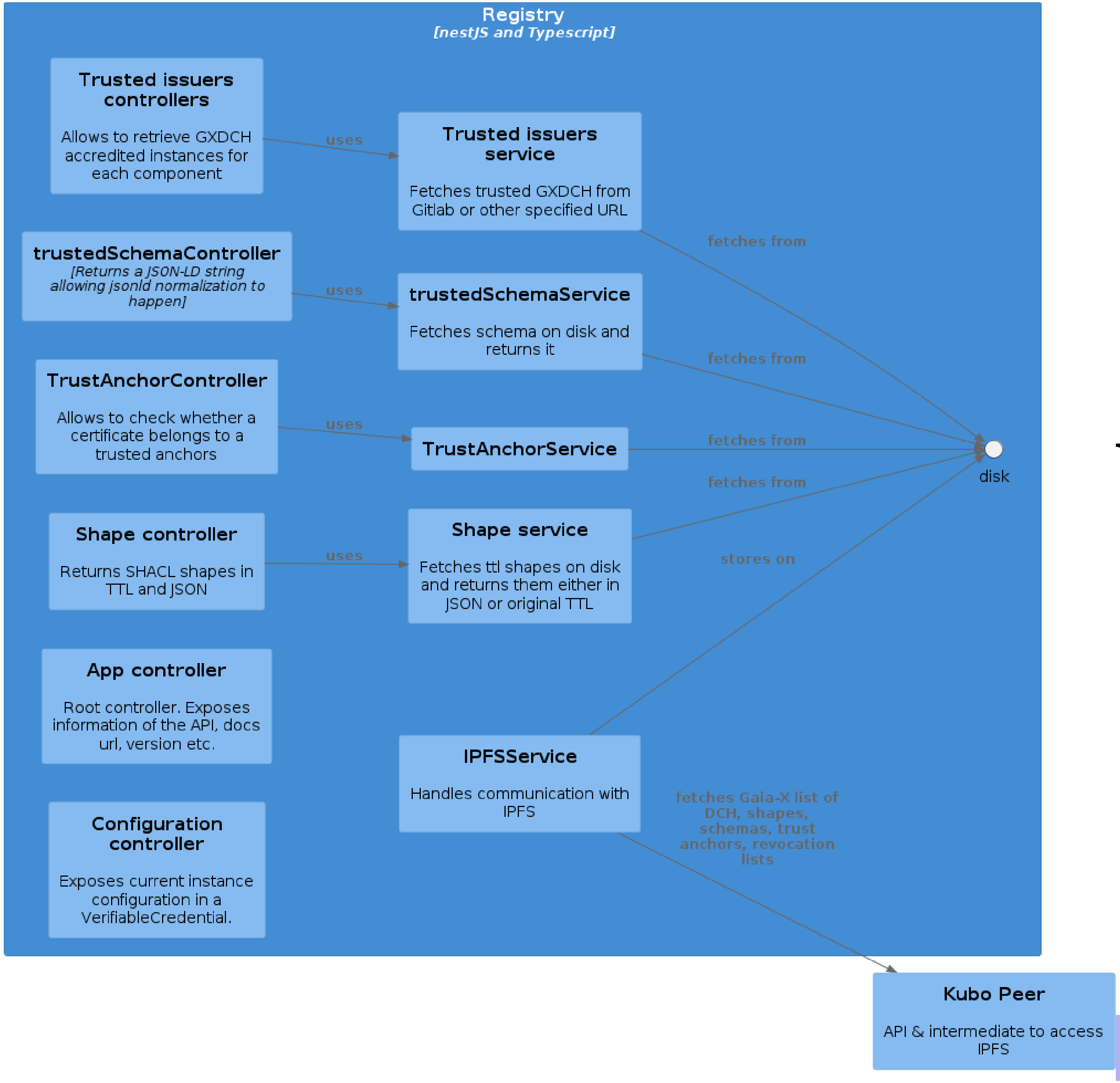
```
true
```

Registry

- Gaia-X's source of truth
  - Validates trust anchors
  - Provides SHACL shapes, schemas
  - Contains the revoked issuers list (malicious actors removed from the ecosystem)
  - Provides the list of accredited clearing houses

Registry

- Gaia-X's sou
  - Validates
  - Provides S
  - Contains t n the ecosysten
  - Provides t



Registry
[nestJS and Typescript]

Trusted issuers controllers
Allows to retrieve GXDCH accredited instances for each component

Trusted issuers service
Fetches trusted GXDCH from Gitlab or other specified URL

trustedSchemaController
[Returns a JSON-LD string allowing jsonld normalization to happen]

trustedSchemaService
Fetches schema on disk and returns it

TrustAnchorController
Allows to check whether a certificate belongs to a trusted anchors

TrustAnchorService

Shape controller
Returns SHACL shapes in TTL and JSON

Shape service
Fetches ttl shapes on disk and returns them either in JSON or original TTL

App controller
Root controller. Exposes information of the API, docs url, version etc.

IPFSService
Handles communication with IPFS

Configuration controller
Exposes current instance configuration in a VerifiableCredential.

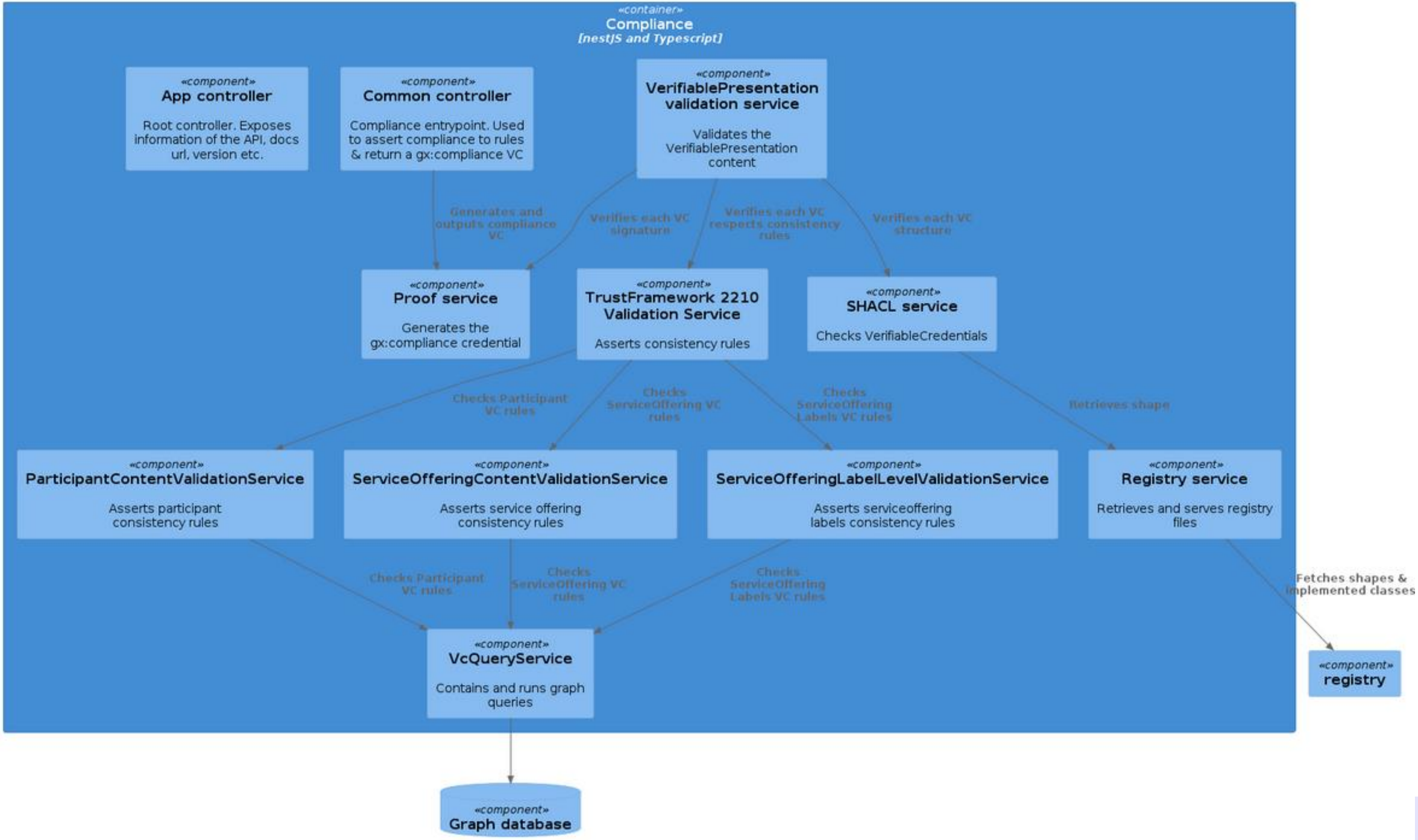
Kubo Peer
API & intermediate to access IPFS

Registry

- Major change in Loire is the backend for storing source files
- We used to use Gitlab repository as storage, and we now rely on IPFS (Interplanetary File System) to distribute them

Compliance

- Engine verifying format and content
  - Signature validation of Verifiable Credentials
  - SHACL Validation of JSON-LD
  - Validation that LegalRegistrationNumber comes from a trusted Notary
  - Few business validations in Tagus (Issuer has signed Terms and Conditions etc.)
  - Dozens of criteria validated in Loire



«container»
Compliance
[nestJS and Typescript]

«component»
App controller
Root controller. Exposes information of the API, docs url, version etc.

«component»
Common controller
Compliance entrypoint. Used to assert compliance to rules & return a gx:compliance VC

«component»
VerifiablePresentation validation service
Validates the VerifiablePresentation content

Generates and outputs compliance VC

Verifies each VC signature

Verifies each VC respects consistency rules

Verifies each VC structure

«component»
Proof service
Generates the gx:compliance credential

«component»
TrustFramework 2210 Validation Service
Asserts consistency rules

«component»
SHACL service
Checks VerifiableCredentials

Checks Participant VC rules

Checks ServiceOffering VC rules

Checks ServiceOffering Labels VC rules

Retrieves shape

«component»
ParticipantContentValidationService
Asserts participant consistency rules

«component»
ServiceOfferingContentValidationService
Asserts service offering consistency rules

«component»
ServiceOfferingLabelLevelValidationService
Asserts serviceoffering labels consistency rules

«component»
Registry service
Retrieves and serves registry files

Checks Participant VC rules

Checks ServiceOffering VC rules

Checks ServiceOffering Labels VC rules

Fetches shapes & implemented classes

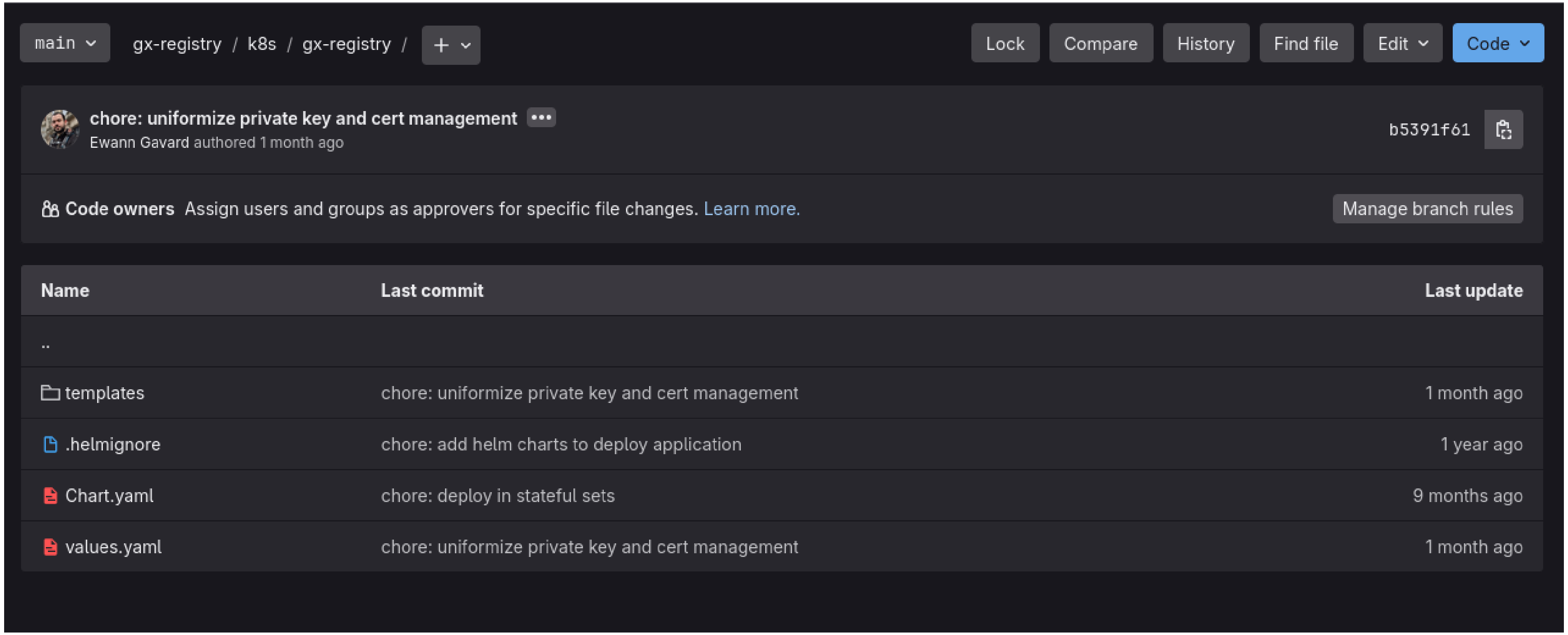
«component»
VcQueryService
Contains and runs graph queries

«component»
registry

«component»
Graph database

How to deploy ?

- Each component has an helm chart



chore: uniformize private key and cert management

Ewann Gavard authored 1 month ago

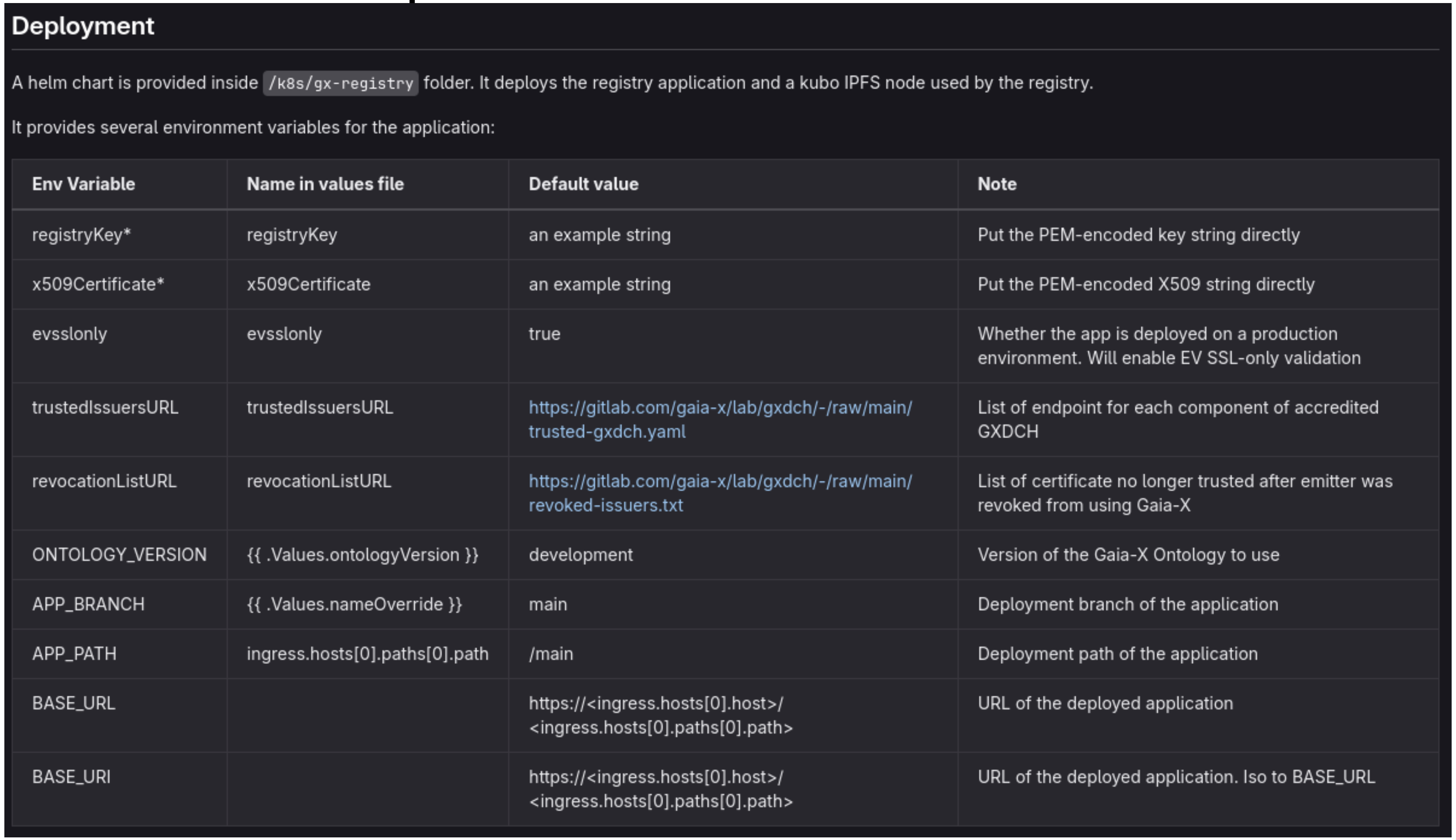
b5391f61

Code owners Assign users and groups as approvers for specific file changes. Learn more.

Name	Last commit	Last update
..		
templates	chore: uniformize private key and cert management	1 month ago
.helmignore	chore: add helm charts to deploy application	1 year ago
Chart.yaml	chore: deploy in stateful sets	9 months ago
values.yaml	chore: uniformize private key and cert management	1 month ago

How to deploy ?

- Each component has an explanation of the variables in their readme



Deployment

A helm chart is provided inside `/k8s/gx-registry` folder. It deploys the registry application and a kubo IPFS node used by the registry.

It provides several environment variables for the application:

Env Variable	Name in values file	Default value	Note
registryKey*	registryKey	an example string	Put the PEM-encoded key string directly
x509Certificate*	x509Certificate	an example string	Put the PEM-encoded X509 string directly
evsslonly	evsslonly	true	Whether the app is deployed on a production environment. Will enable EV SSL-only validation
trustedIssuersURL	trustedIssuersURL	https://gitlab.com/gaia-x/lab/gxdch/-/raw/main/trusted-gxdch.yaml	List of endpoint for each component of accredited GXDCH
revocationListURL	revocationListURL	https://gitlab.com/gaia-x/lab/gxdch/-/raw/main/revoked-issuers.txt	List of certificate no longer trusted after emitter was revoked from using Gaia-X
ONTOLOGY_VERSION	{{ .Values.ontologyVersion }}	development	Version of the Gaia-X Ontology to use
APP_BRANCH	{{ .Values.nameOverride }}	main	Deployment branch of the application
APP_PATH	ingress.hosts[0].paths[0].path	/main	Deployment path of the application
BASE_URL		https://<ingress.hosts[0].host>/<ingress.hosts[0].paths[0].path>	URL of the deployed application
BASE_URI		https://<ingress.hosts[0].host>/<ingress.hosts[0].paths[0].path>	URL of the deployed application. Iso to BASE_URL

I need more help to deploy

- A course exists in the Gaia-X Academy and a significant update will be published after the summit
- Each component has its own documentation (for v1 and the latest)
- An amazing community exists on Slack

Slack

Gaia-X Academy

Thank you!

Christophe Strnadl – Ewann Gavard
Gaia-X Association For Data And Cloud AISBL

Gaia-X Ontology The Service Offering Encyclopedia

gaia-x

Vincent Kelleher

Software Engineer

Gaia-X Lab Team

#GaiaXSummit24

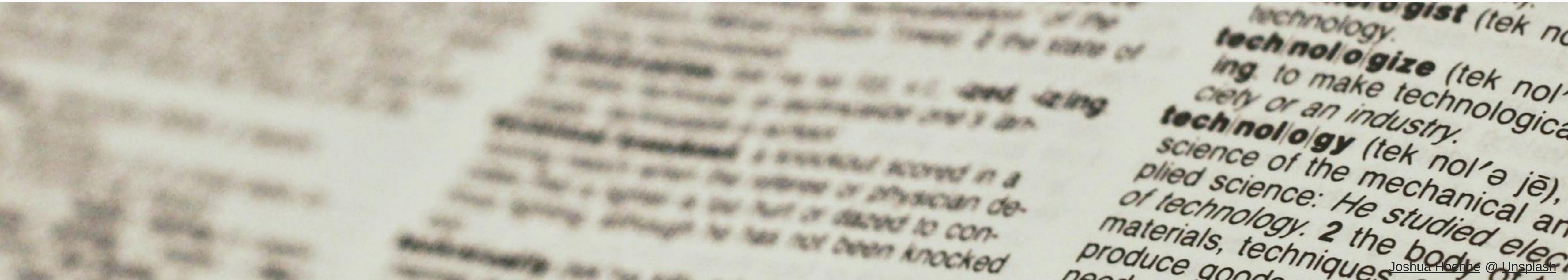
SUMMARY

Nienke Broeksema @ Unsplash

1. Ontology 101
2. Visualisation & Serialization
3. How It’s Built
4. Broadcasting The Ontology
5. EIY : Extend It Yourself

ONTOLOGY 101

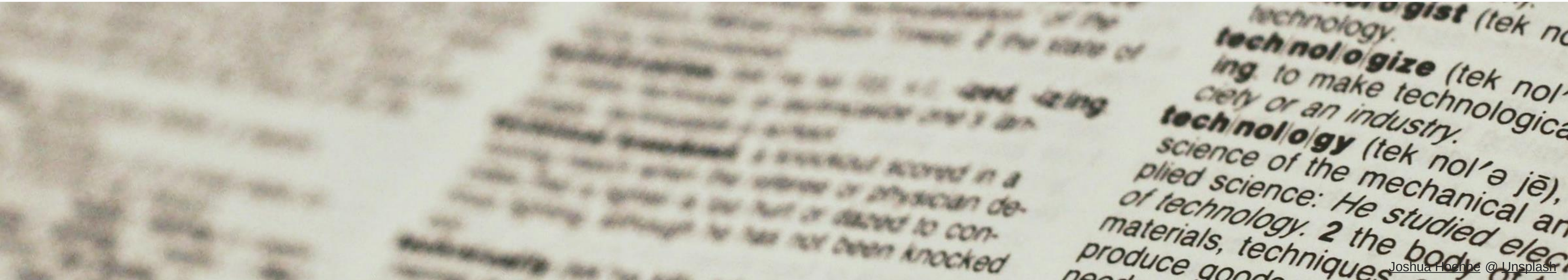
« An ontology is a formal description of knowledge as a set of concepts within a domain and the relationships that hold between them » - *Ontotext.com*


technology.
tech nol o gize (tek nol
ing. to make technologica
ciety or an industry.
tech nol o gy (tek nol′ə jē),
science of the mechanical an
plied science: He studied elec
of technology. 2 the body
materials, techniques
produce goods

Joshua Hoehne @ Unsplash

ONTOLOGY 101

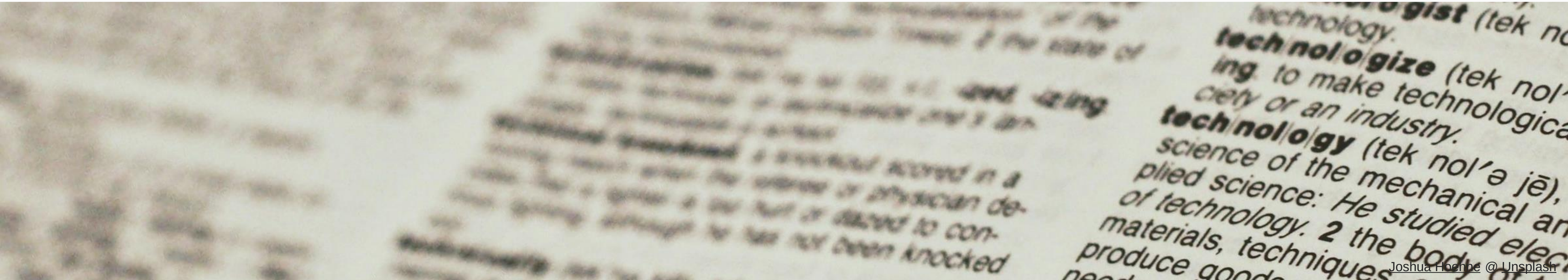
- Very appreciated in the artificial intelligence domain
- Improves communication and problem solving
- Acts as a single point of truth maintained by the community
- Ontologies are designed to be extended, reused and linked together



Joshua Hoehne @ Unsplash

ONTOLOGY 101

- Ideal for building knowledge graphs / semantic networks
- Great base for semantic reasoning
- Inference on knowledge graphs can extract new facts from existing knowledge
- Set theory can be applied on ontologies


technology.
tech nol o gize (tek nol
ing. to make technologica
ciety or an industry.
tech nol o gy (tek nol′ə jē),
science of the mechanical an
plied science: He studied elec
of technology. 2 the body
materials, techniques
produce goods

Joshua Hoehne @ Unsplash

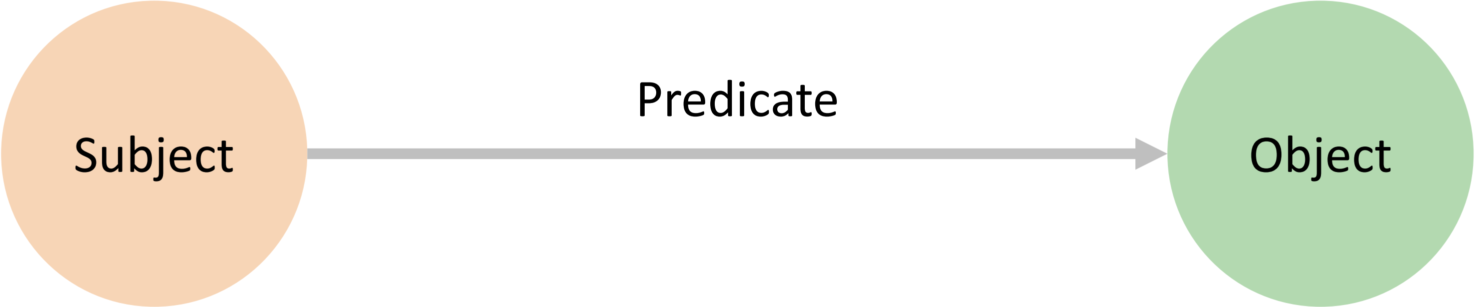
VISUALIZATION & SERIALIZATION

- Ontologies can be visualized as RDFS (Resource Description Framework Schema) graphs
- Each class is a node
- Each relationship is an edge
- RDFS is a semantic extension of RDF
- RDFS describes classes, RDF describes instances

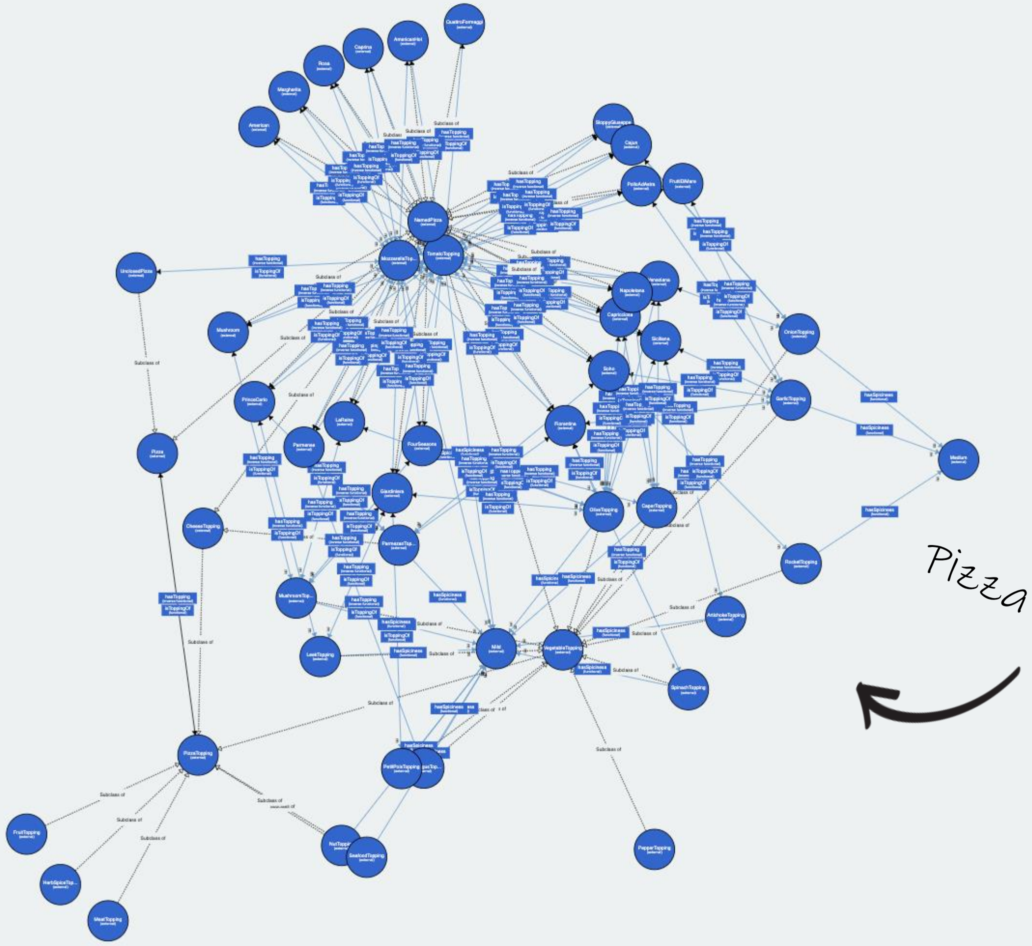
Maxim Berg @ Unsplash

Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION



VISUALIZATION & SERIALIZATION

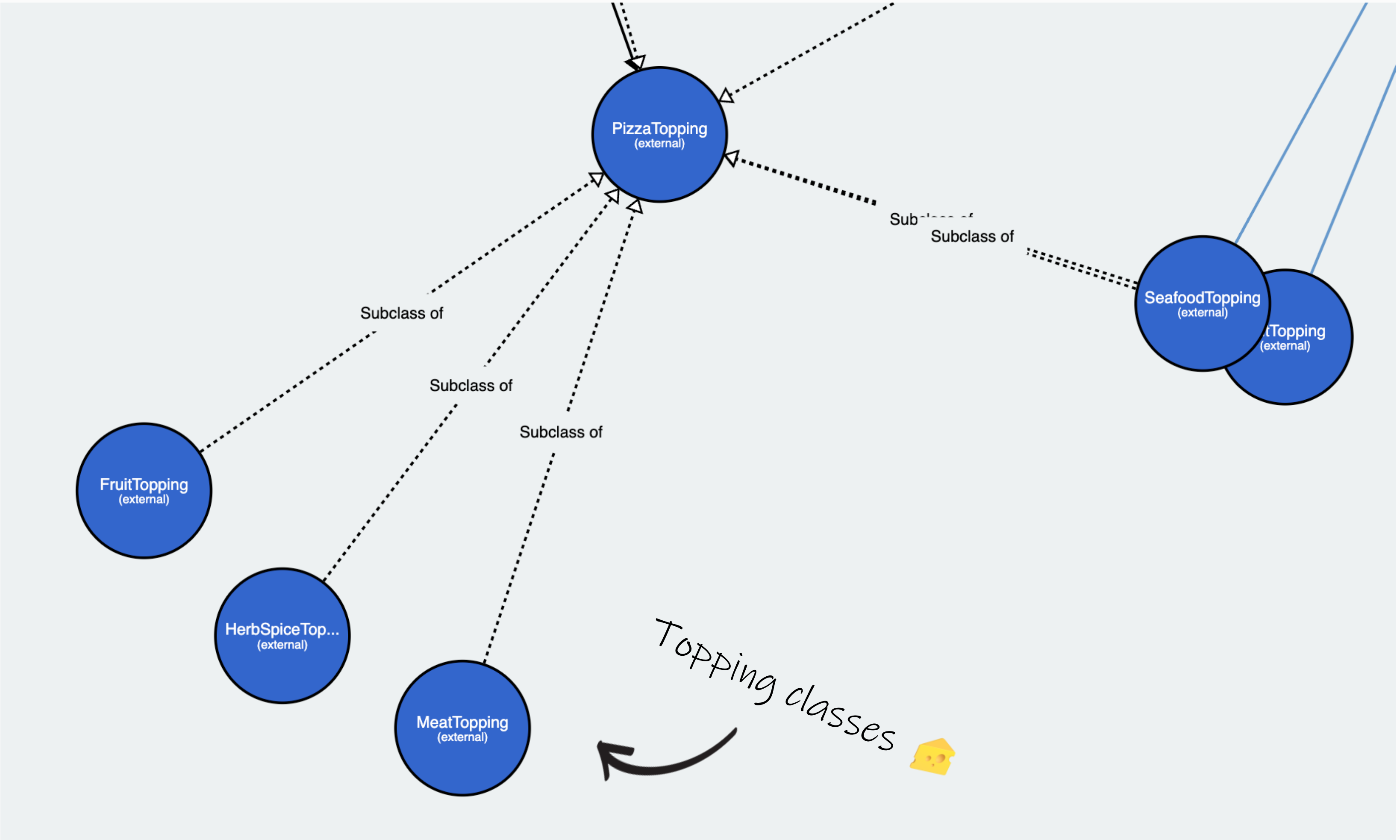


Pizza Ontology 🍕

Maxim Berg @ Unsplash

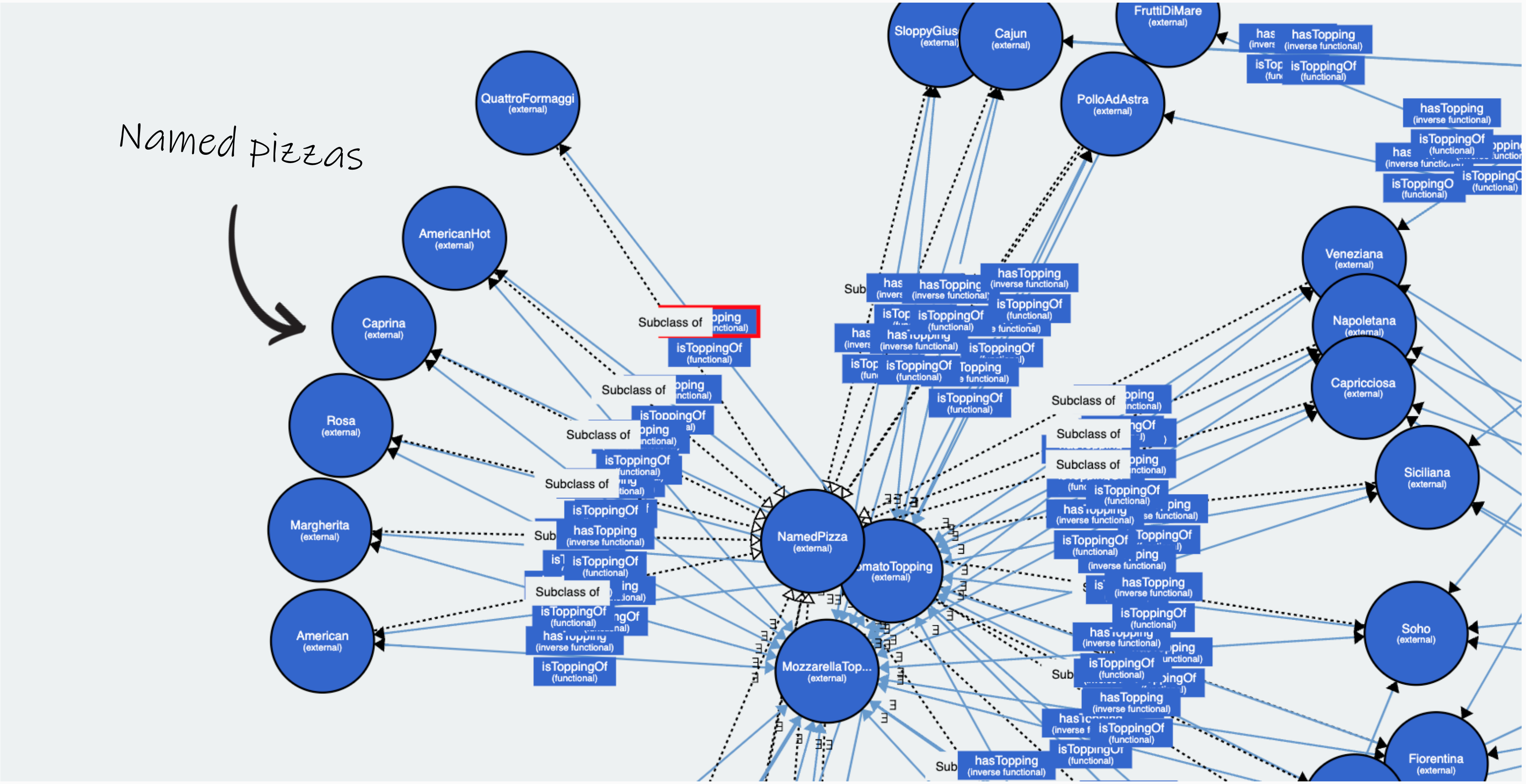
Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION



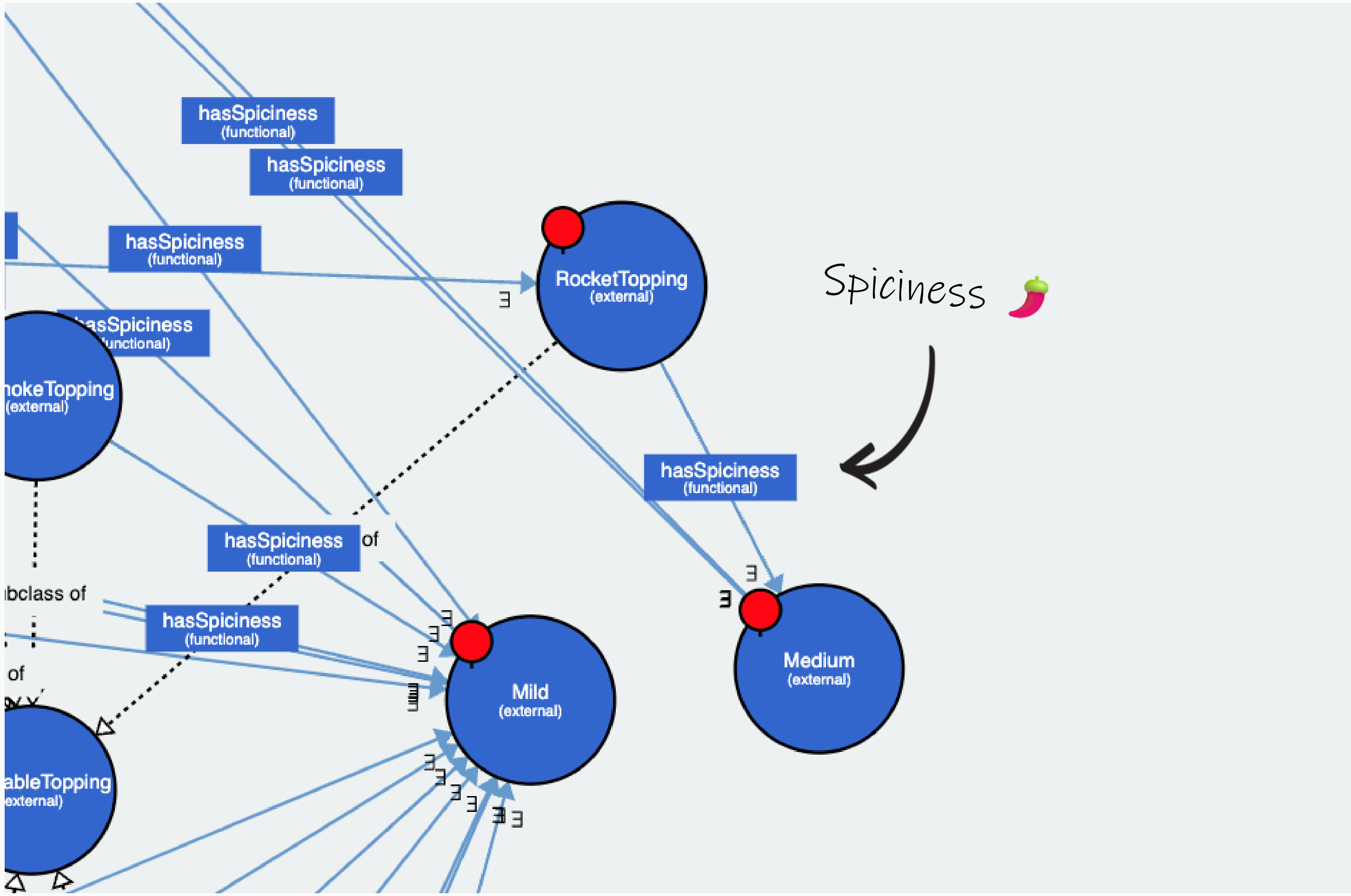
Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION



Named pizzas

VISUALIZATION & SERIALIZATION



hasSpiciness (functional)
hasSpiciness (functional)
hasSpiciness (functional)
asSpiciness unctional)
RocketTopping (external)
Spiciness 🌶️
hoke Topping (external)
hasSpiciness (functional)
hasSpiciness (functional)
of
bclass of
hasSpiciness (functional)
of
Mild (external)
Medium (external)
ableTopping external)
∃

Maxim Berg @ Unsplash

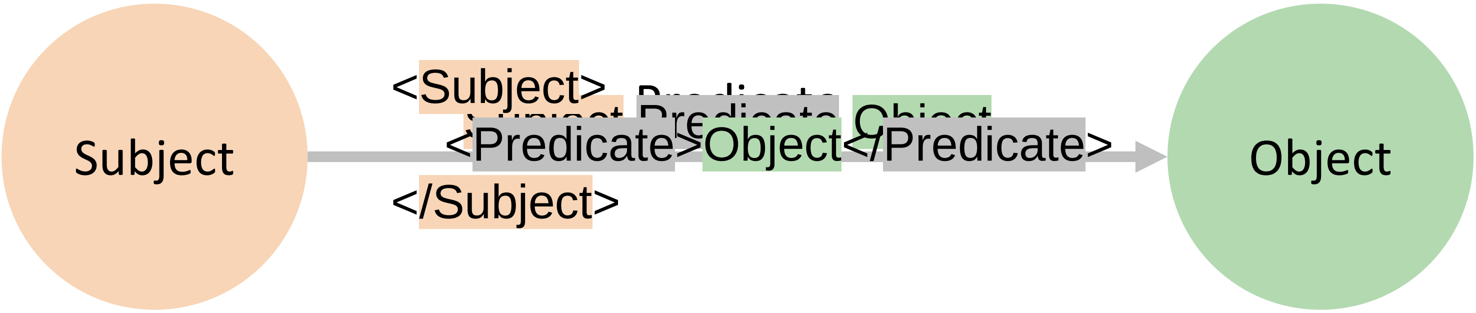
Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION

- RDF/RDFS graphs can be serialized into text formats
- OWL proposes syntaxes such as
  - RDF/XML
  - Turtle
  - Manchester Syntax
  - OWL/XML

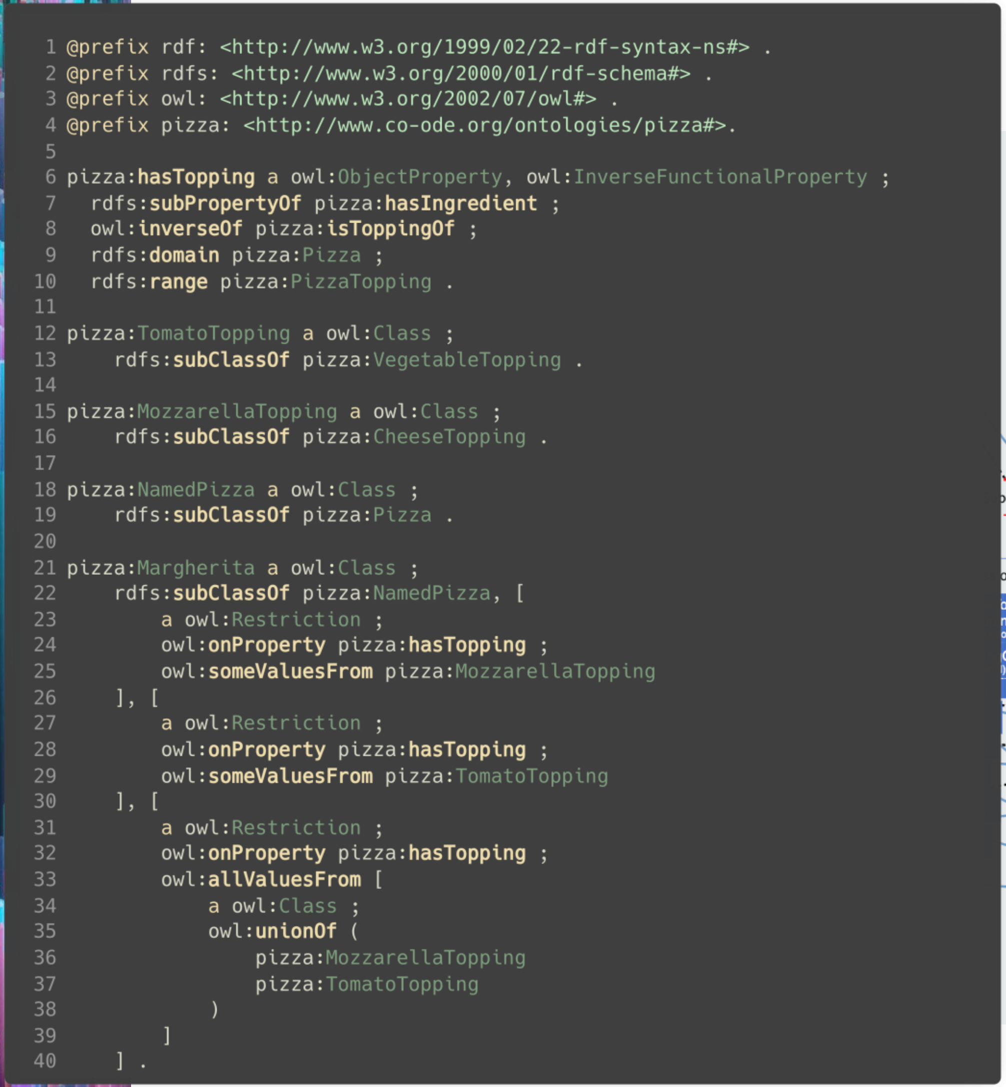
Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION

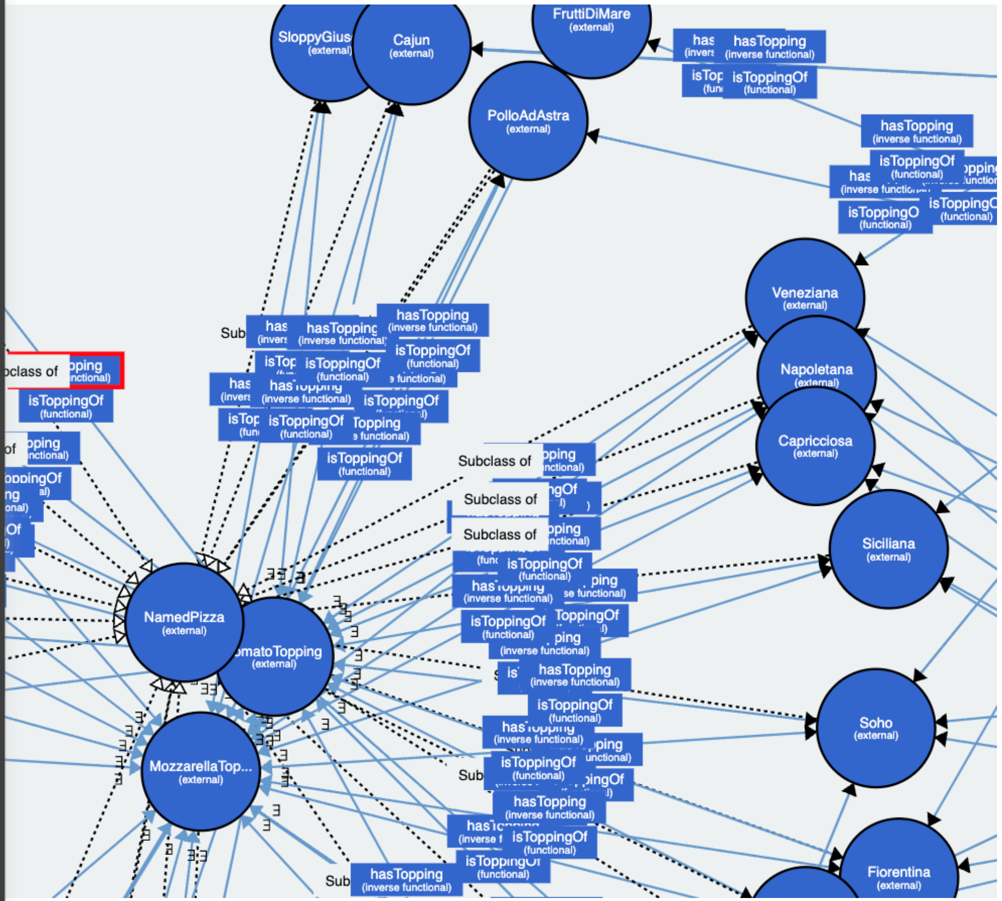


```
<Subject>
  <Predicate>Object</Predicate>
</Subject>
```

VISUALIZATION & SERIALIZATION



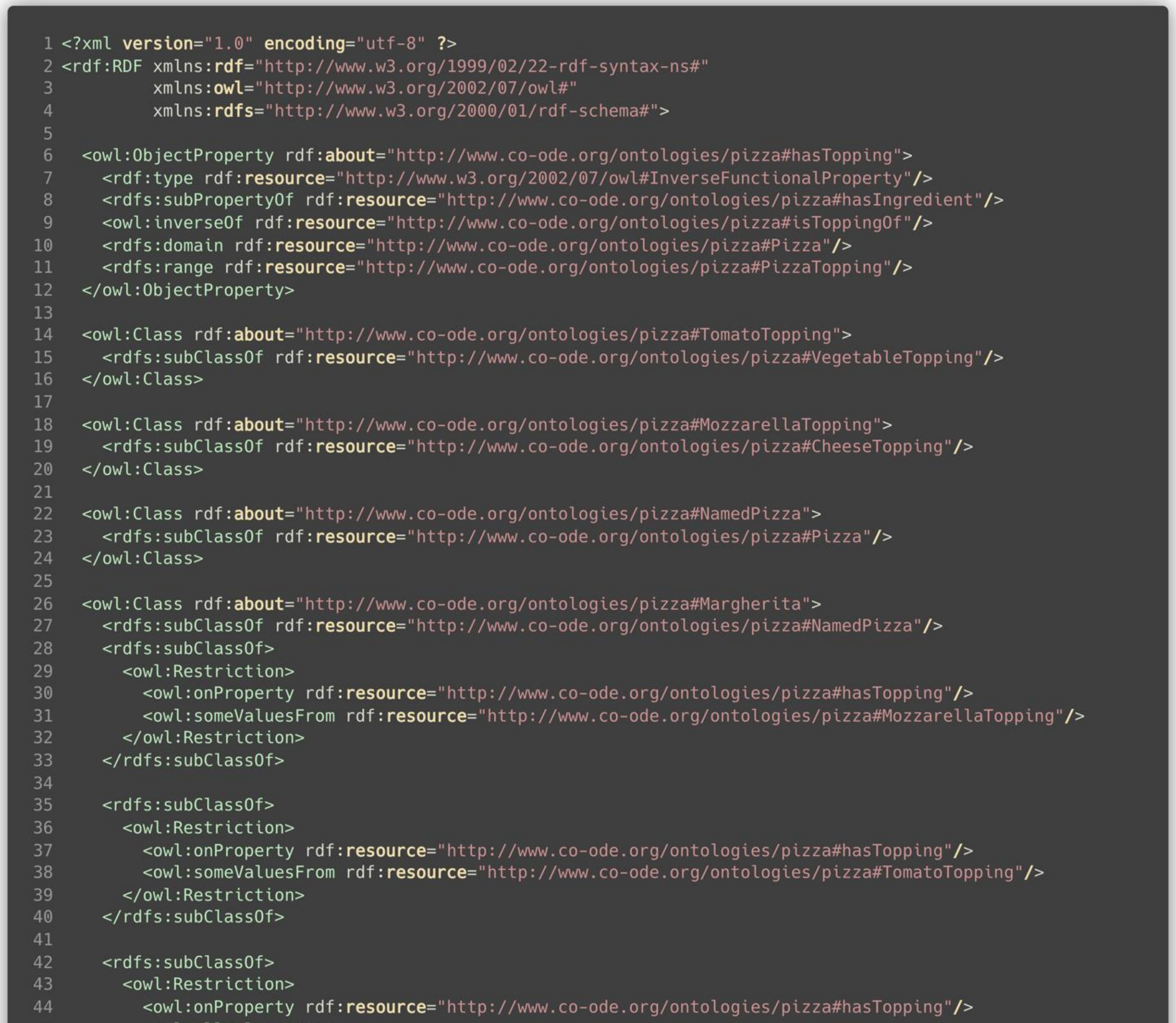
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix pizza: <http://www.co-ode.org/ontologies/pizza#>.

pizza:hasTopping a owl:ObjectProperty, owl:InverseFunctionalProperty ;
  rdfs:subPropertyOf pizza:hasIngredient ;
  owl:inverseOf pizza:isToppingOf ;
  rdfs:domain pizza:Pizza ;
  rdfs:range pizza:PizzaTopping .

pizza:TomatoTopping a owl:Class ;
    rdfs:subClassOf pizza:VegetableTopping .

pizza:MozzarellaTopping a owl:Class ;
    rdfs:subClassOf pizza:CheeseTopping .

pizza:NamedPizza a owl:Class ;
    rdfs:subClassOf pizza:Pizza .

pizza:Margherita a owl:Class ;
    rdfs:subClassOf pizza:NamedPizza, [
        a owl:Restriction ;
        owl:onProperty pizza:hasTopping ;
        owl:someValuesFrom pizza:MozzarellaTopping
    ], [
        a owl:Restriction ;
        owl:onProperty pizza:hasTopping ;
        owl:someValuesFrom pizza:TomatoTopping
    ], [
        a owl:Restriction ;
        owl:onProperty pizza:hasTopping ;
        owl:allValuesFrom [
            a owl:Class ;
            owl:unionOf (
                pizza:MozzarellaTopping
                pizza:TomatoTopping
            )
        ]
    ] .
```



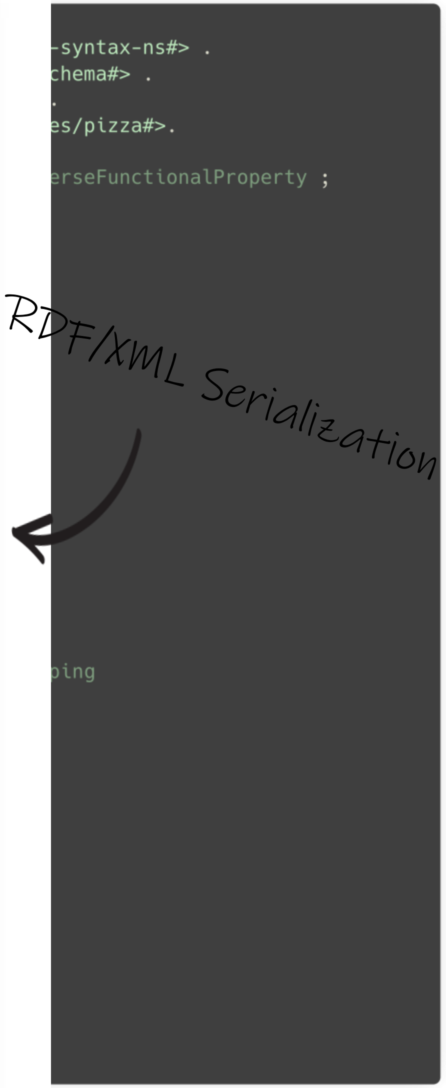
Maxim Berg @ Unsplash

Maxim Berg @ Unsplash

VISUALIZATION & SERIALIZATION



```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/pizza#hasTopping">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty"/>
    <rdfs:subPropertyOf rdf:resource="http://www.co-ode.org/ontologies/pizza#hasIngredient"/>
    <owl:inverseOf rdf:resource="http://www.co-ode.org/ontologies/pizza#isToppingOf"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/pizza#Pizza"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/pizza#PizzaTopping"/>
  </owl:ObjectProperty>

  <owl:Class rdf:about="http://www.co-ode.org/ontologies/pizza#TomatoTopping">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/pizza#VegetableTopping"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.co-ode.org/ontologies/pizza#MozzarellaTopping">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/pizza#CheeseTopping"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.co-ode.org/ontologies/pizza#NamedPizza">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/pizza#Pizza"/>
  </owl:Class>

  <owl:Class rdf:about="http://www.co-ode.org/ontologies/pizza#Margherita">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/pizza#NamedPizza"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="http://www.co-ode.org/ontologies/pizza#hasTopping"/>
        <owl:someValuesFrom rdf:resource="http://www.co-ode.org/ontologies/pizza#MozzarellaTopping"/>
      </owl:Restriction>
    </rdfs:subClassOf>

    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="http://www.co-ode.org/ontologies/pizza#hasTopping"/>
        <owl:someValuesFrom rdf:resource="http://www.co-ode.org/ontologies/pizza#TomatoTopping"/>
      </owl:Restriction>
    </rdfs:subClassOf>

    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="http://www.co-ode.org/ontologies/pizza#hasTopping"/>
```



```
-syntax-ns#> .
chema#> .
.
es/pizza#>.

erseFunctionalProperty ;

ping
```

RDF/XML Serialization

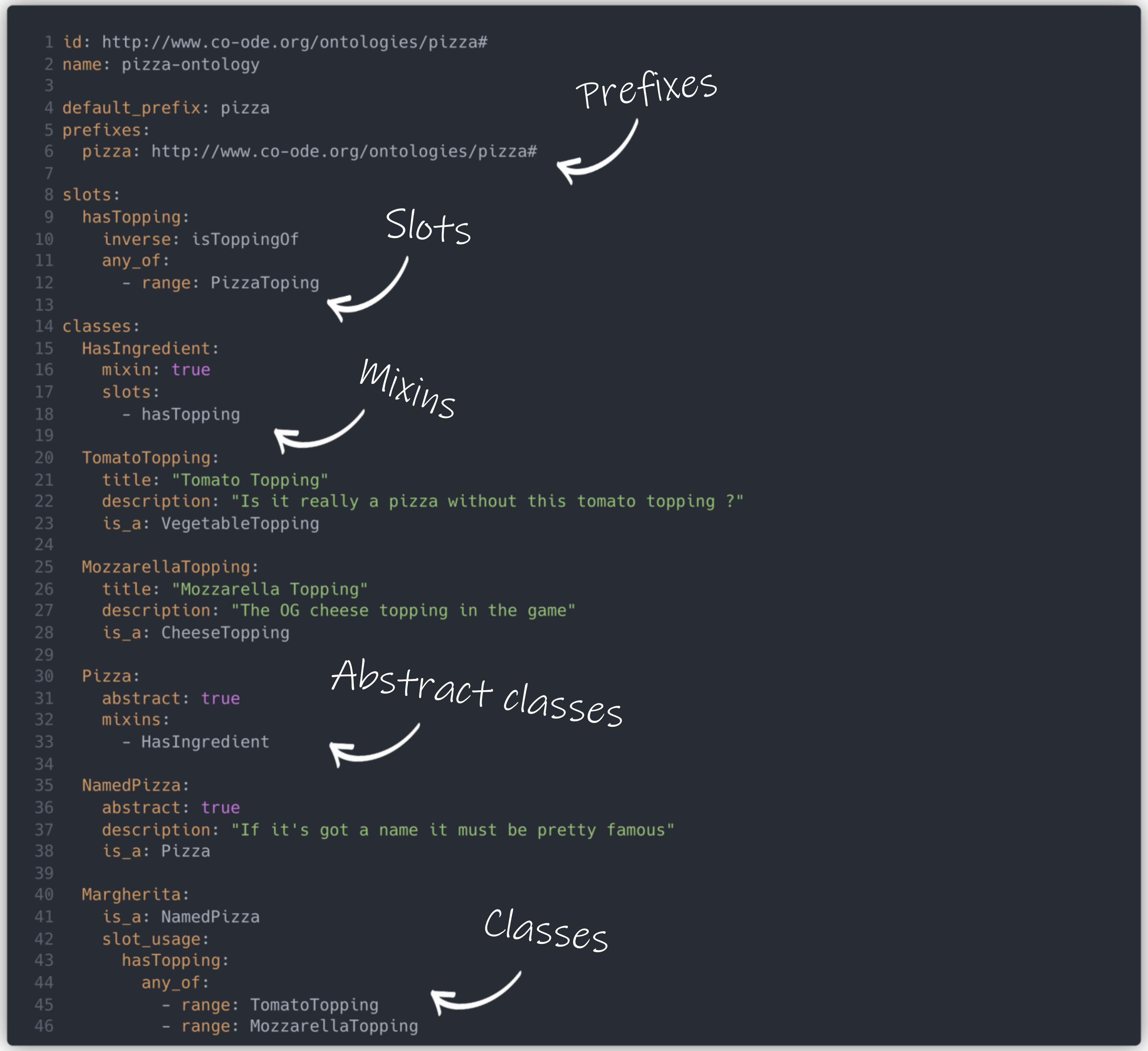
Christian Bass @ Unsplash

HOW IT'S BUILT

- Service Characteristics Working Group produces the LinkML ontology
- LinkML is a linked data modeling language
- Gaia-X Lab Team actively contributes to LinkML
- It's based on YAML files
- LinkML generates multiple deliverables from a single input
- Generation is done with Gitlab CI on the Gaia-X Ontology repository

HOW IT'S BUILT

Christian Bass @ Unsplash



```yaml
id: http://www.co-ode.org/ontologies/pizza#
name: pizza-ontology

default_prefix: pizza
prefixes:
  pizza: http://www.co-ode.org/ontologies/pizza#

slots:
  hasTopping:
    inverse: isToppingOf
    any_of:
      - range: PizzaToping

classes:
  HasIngredient:
    mixin: true
    slots:
      - hasTopping

  TomatoTopping:
    title: "Tomato Topping"
    description: "Is it really a pizza without this tomato topping ?"
    is_a: VegetableTopping

  MozzarellaTopping:
    title: "Mozzarella Topping"
    description: "The OG cheese topping in the game"
    is_a: CheeseTopping

  Pizza:
    abstract: true
    mixins:
      - HasIngredient

  NamedPizza:
    abstract: true
    description: "If it's got a name it must be pretty famous"
    is_a: Pizza

  Margherita:
    is_a: NamedPizza
    slot_usage:
      hasTopping:
        any_of:
          - range: TomatoTopping
          - range: MozzarellaTopping
```

Prefixes

Slots

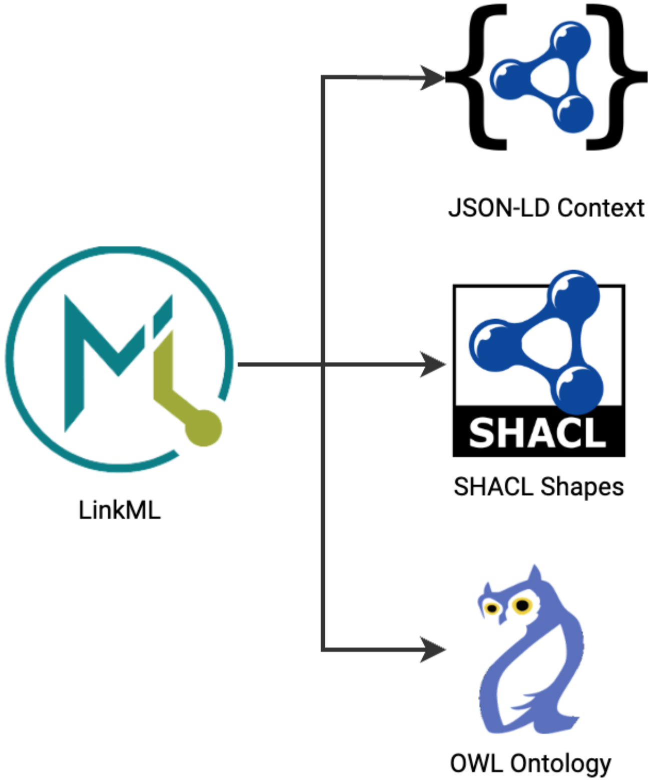
Mixins

Abstract classes

Classes

Christian Bass @ Unsplash

HOW IT'S BUILT



Gives semantical meaning to JSON-LD files by describing a common Gaia-X vocabulary.

W3C standardized format that defines conditions to ensure that an RDF graph is valid through shapes.

Semantic Web language used to represent things, groups of things and their relations within ontologies.

HOW IT'S BUILT

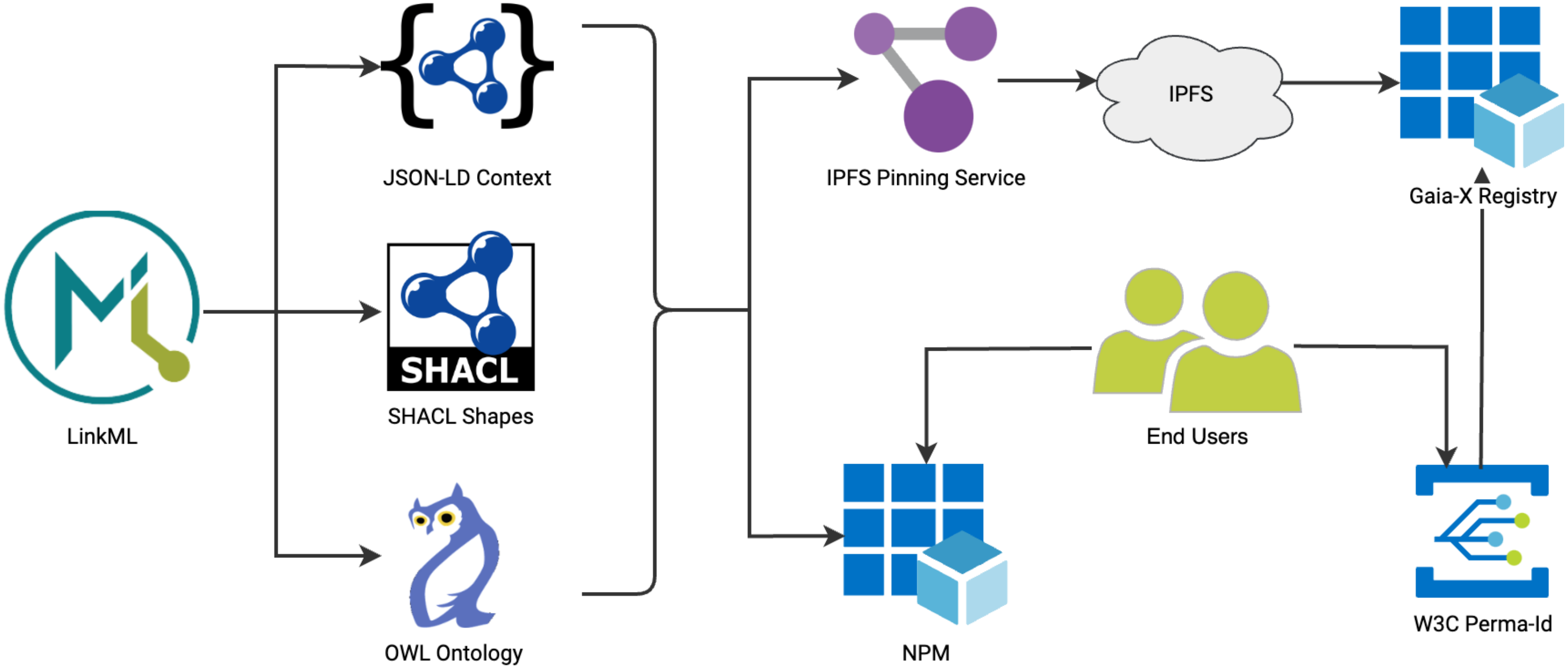
Christian Bass @ Unsplash

- Each entity is tested in the usage conditions of the Gaia-X Compliance Engine
- Tests use Typescript with Vitest
- JSON-LD test data files are used
- Generated JSON-LD contexts are used during the testing phase
- Generated OWL ontologies provide class hierarchy information in tests
- Generated SHACL shapes are finally used to validate each entity

BROADCASTING THE ONTOLOGY

gaia-x

力力摄影日记 @ Unsplash



力力摄影日记 @ Unsplash

BROADCASTING THE ONTOLOGY

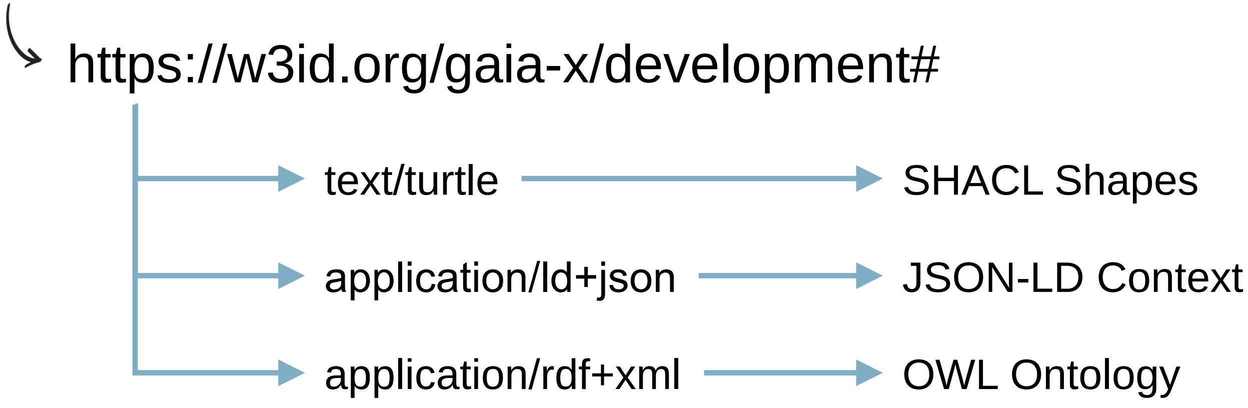
- Our goal is to be FAIR
  - **Findable** – Metadata & data should be easily findable
  - **Accessible** – Metadata & data be accessible through open/free protocols
  - **Interoperable** – Knowledge is represented in a broadly applicable language
  - **Reusable** – Data should be clearly described and documented for reuse

Gaia-x Ontology documentation

カカ摄影日记 @ Unsplash

BROADCASTING THE ONTOLOGY

- Redirected through w3id.org’s perma-id project
- Uses MIME type to specify which resource is queries



↳ https://w3id.org/gaia-x/development#

text/turtle	SHACL Shapes
application/ld+json	JSON-LD Context
application/rdf+xml	OWL Ontology

↳ https://w3id.org/gaia-x/development/linkml/types.yaml

EIY : EXTEND IT YOURSELF

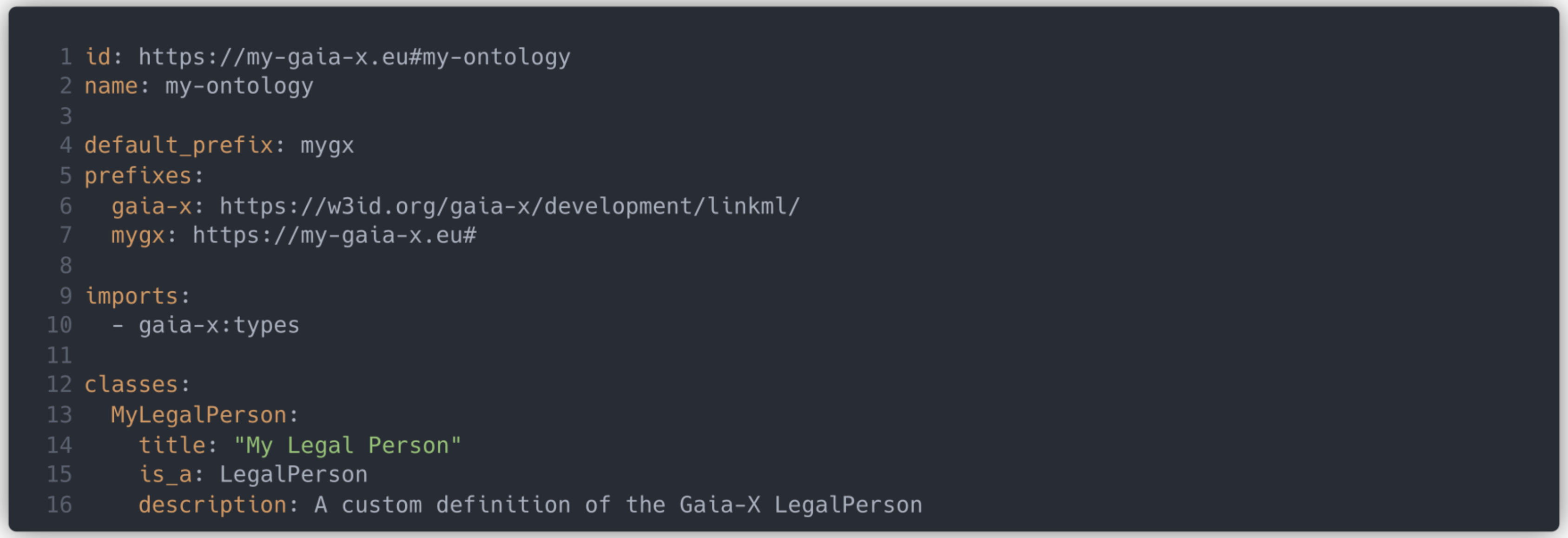
↳ https://w3id.org/gaia-x/development/linkml/types.yaml

- Specific URL serving a LinkML YAML file
- This YAML file contains the whole Gaia-X ontology
- Out-of-the-box extension support by LinkML
- **Anyone can extend the Gaia-X ontology via LinkML**

Photo by Nik on Unsplash

Photo by Nik on Unsplash

EIY : EXTEND IT YOURSELF

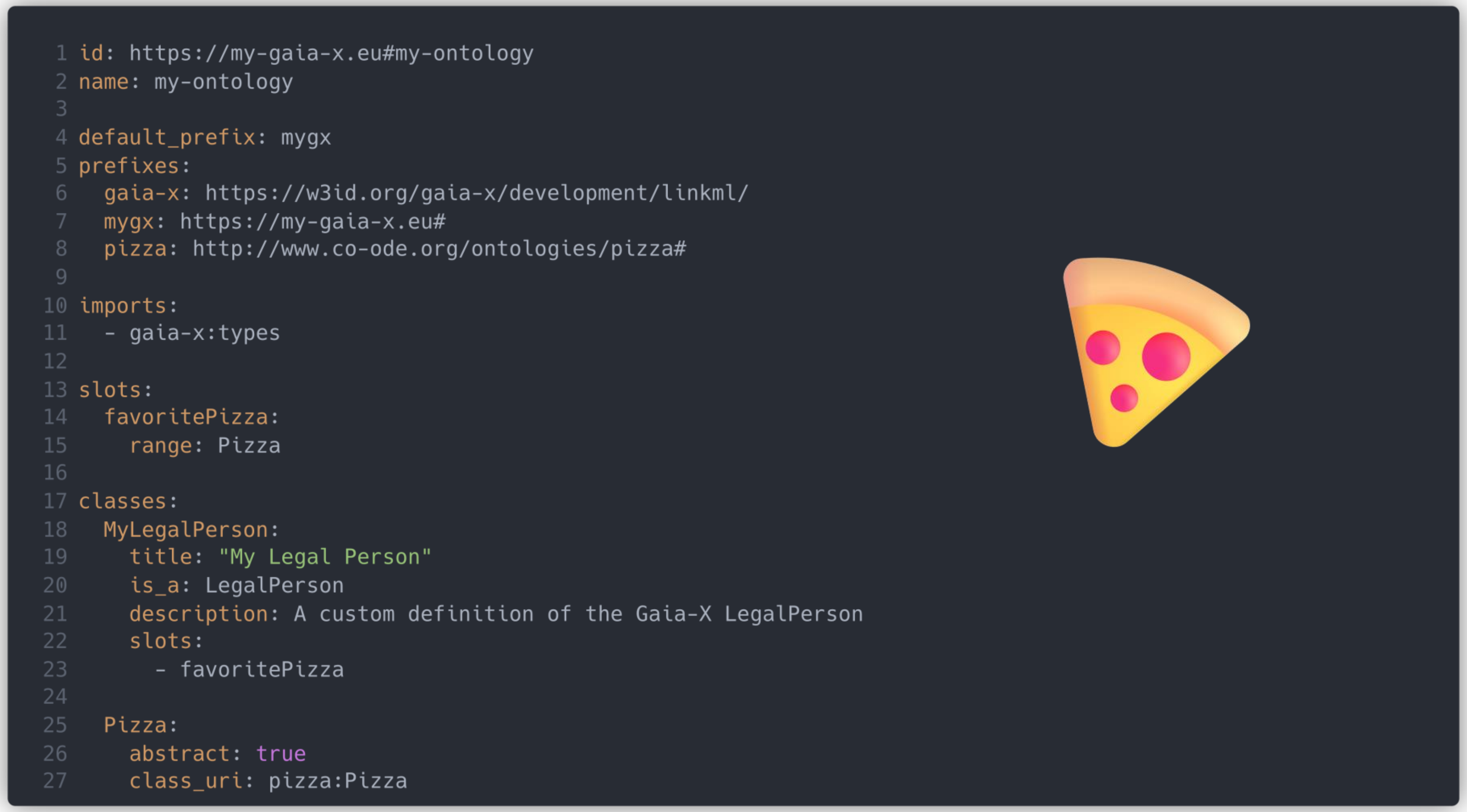


```yaml
id: https://my-gaia-x.eu#my-ontology
name: my-ontology

default_prefix: mygx
prefixes:
  gaia-x: https://w3id.org/gaia-x/development/linkml/
  mygx: https://my-gaia-x.eu#

imports:
  - gaia-x:types

classes:
  MyLegalPerson:
    title: "My Legal Person"
    is_a: LegalPerson
    description: A custom definition of the Gaia-X LegalPerson
```

- Add the Gaia-X prefix and import the related types
- LinkML resolves the URL from the previous slide
- An Gaia-X entity can be extended

Photo by Nik on Unsplash

EIY : EXTEND IT YOURSELF



```yaml
id: https://my-gaia-x.eu#my-ontology
name: my-ontology

default_prefix: mygx
prefixes:
  gaia-x: https://w3id.org/gaia-x/development/linkml/
  mygx: https://my-gaia-x.eu#
  pizza: http://www.co-ode.org/ontologies/pizza#

imports:
  - gaia-x:types

slots:
  favoritePizza:
    range: Pizza

classes:
  MyLegalPerson:
    title: "My Legal Person"
    is_a: LegalPerson
    description: A custom definition of the Gaia-X LegalPerson
    slots:
      - favoritePizza

  Pizza:
    abstract: true
    class_uri: pizza:Pizza
```

Thank you!

Vincent Kelleher
vincent.kelleher@gaia-x.eu

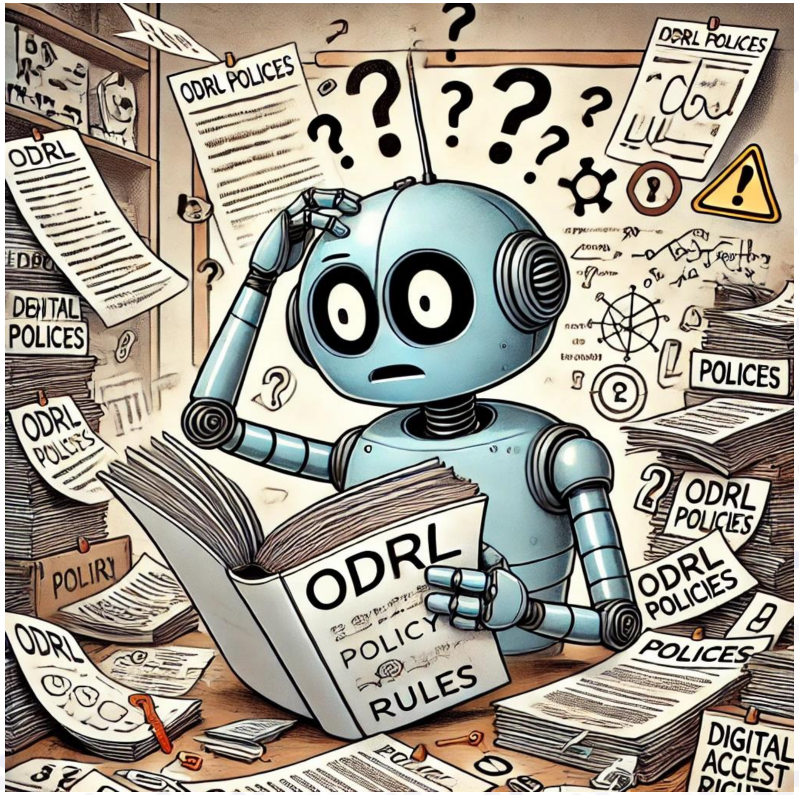
Cloud Interoperability, the Data Act, and a new Eclipse Foundation Open Source Working Group For Cloud

12:00 – 12:30

Christoph Strnadl, Chief Technology Officer, Gaia-X

Marco D'Angelo, Director, Cloud Ecosystem Lead Open Source, Huawei

ODRL: Our Data Rules, Literally!



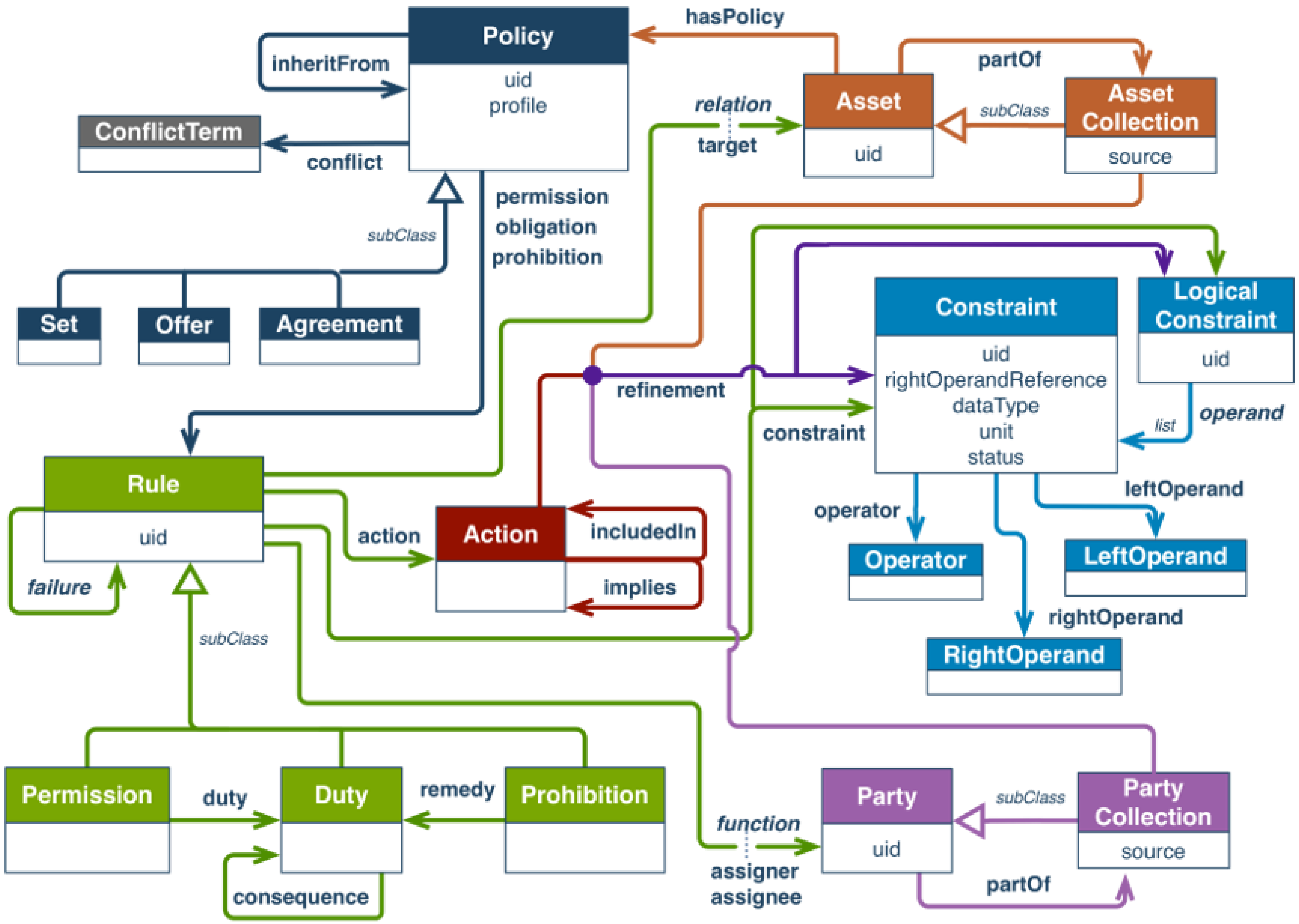
Yassir SELLAMI

Software Engineer

Gaia-X AISBL

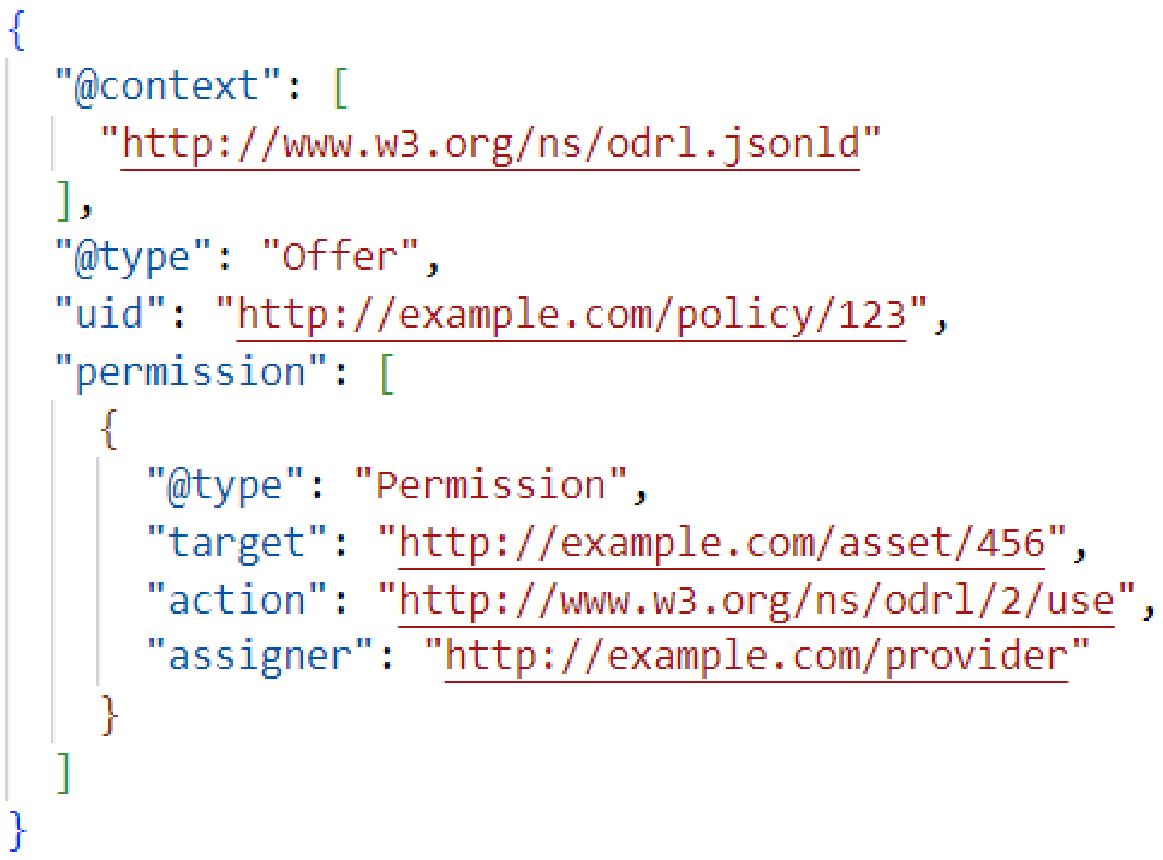
ODRL
ACCESS AND USAGE CONTROL
VENDOR-SPECIFIC SYSTEM
imgflip.com

Open Digital Rights Language

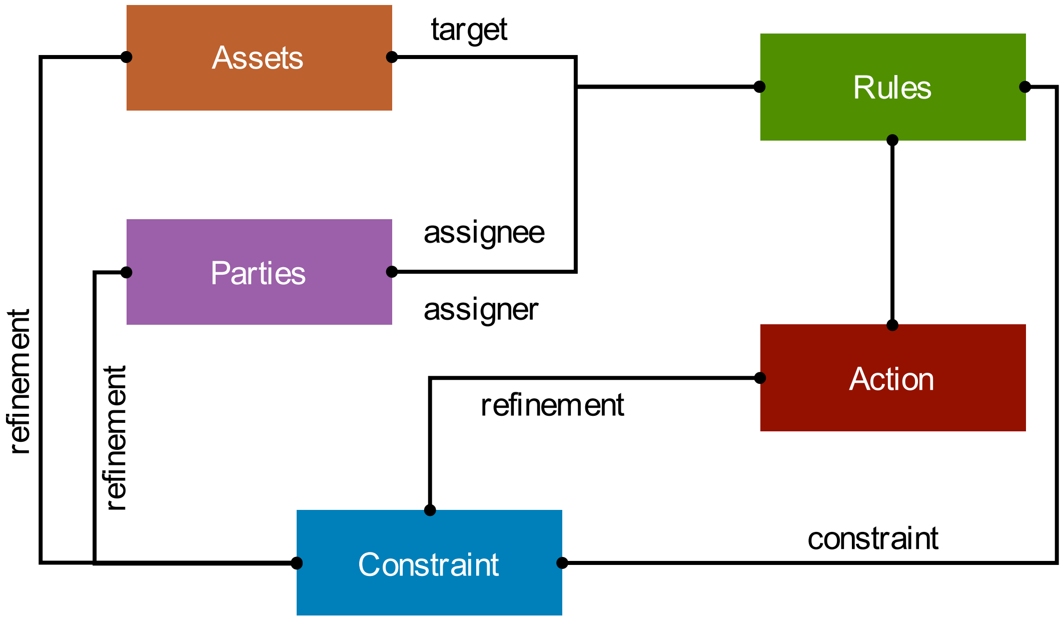


#GaiaXSummit24

Open Digital Rights Language

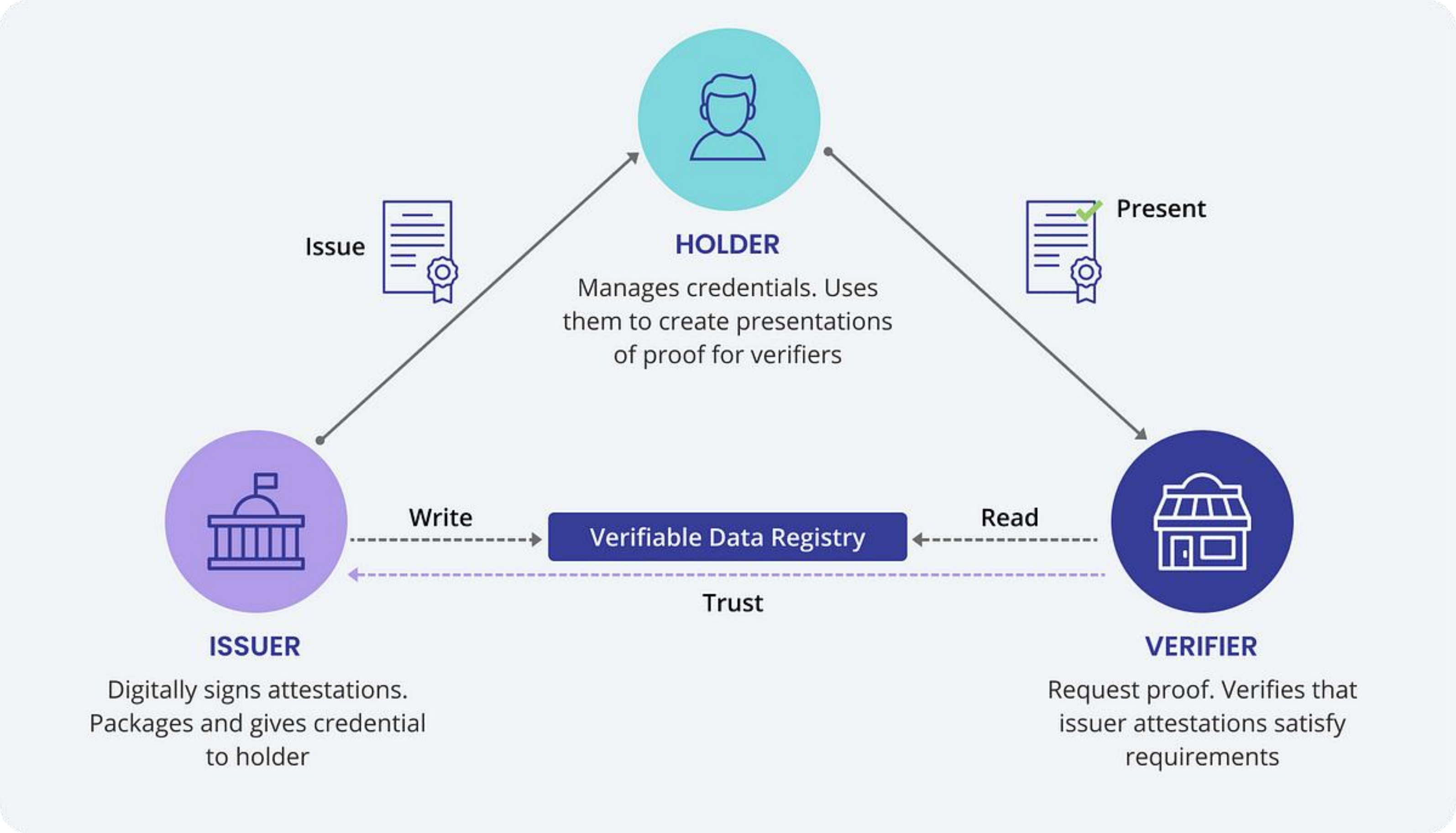


```
{
  "@context": [
    "http://www.w3.org/ns/odrl.jsonld"
  ],
  "@type": "Offer",
  "uid": "http://example.com/policy/123",
  "permission": [
    {
      "@type": "Permission",
      "target": "http://example.com/asset/456",
      "action": "http://www.w3.org/ns/odrl/2/use",
      "assigner": "http://example.com/provider"
    }
  ]
}
```

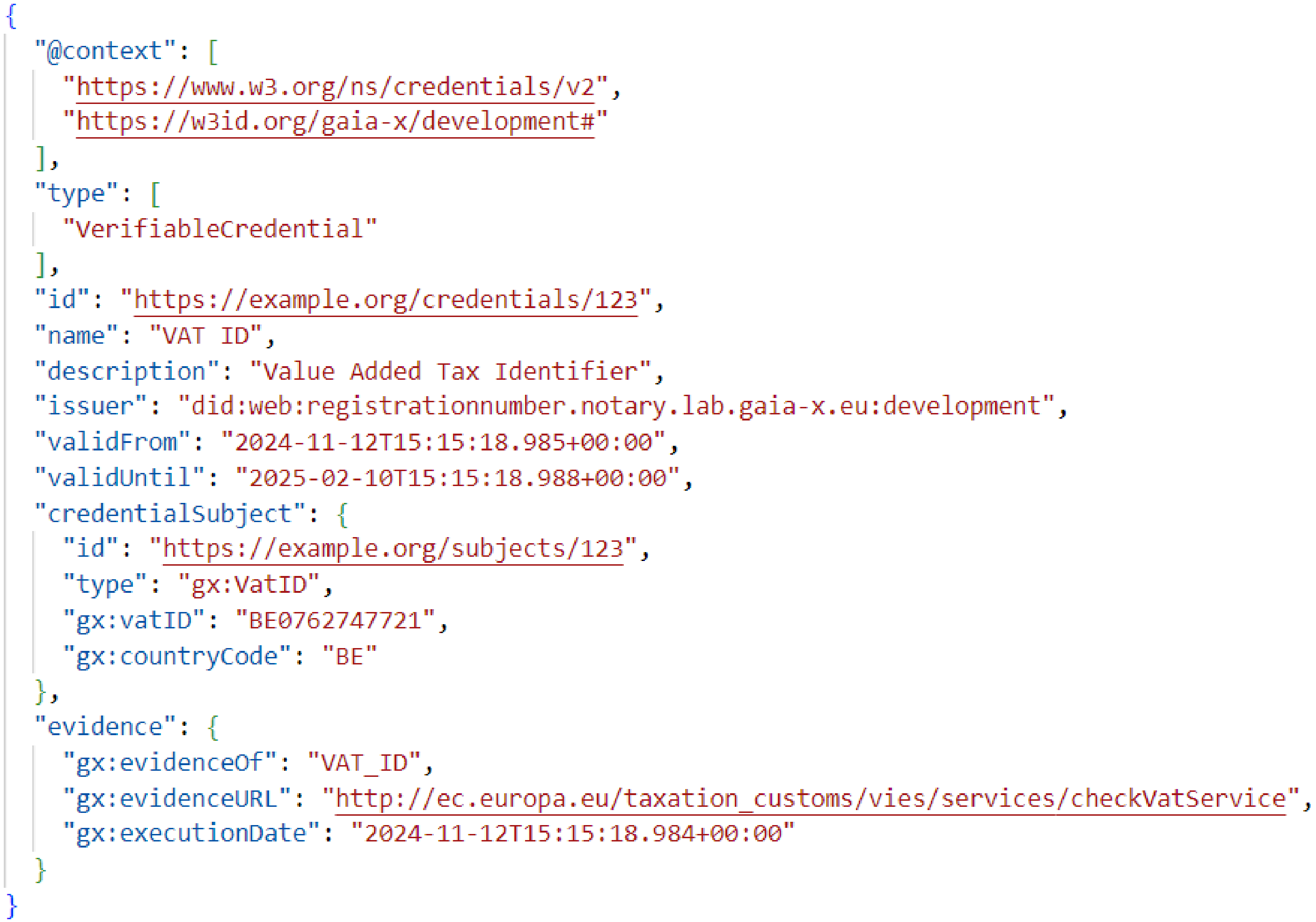


Verifiable Credentials

- VCs are cryptographically signed by the issuer, allowing to check data tampering and issuer's legitimacy
- Used in Gaia-X to represent everything, companies, resources, services
- Represents any form of credential, permits, license


HOLDER
Manages credentials. Uses them to create presentations of proof for verifiers
Issue
Present
ISSUER
Digitally signs attestations. Packages and gives credential to holder
Write
Verifiable Data Registry
Read
Trust
VERIFIER
Request proof. Verifies that issuer attestations satisfy requirements

Verifiable Credentials



```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "type": [
    "VerifiableCredential"
  ],
  "id": "https://example.org/credentials/123",
  "name": "VAT ID",
  "description": "Value Added Tax Identifier",
  "issuer": "did:web:registrationnumber.notary.lab.gaia-x.eu:development",
  "validFrom": "2024-11-12T15:15:18.985+00:00",
  "validUntil": "2025-02-10T15:15:18.988+00:00",
  "credentialSubject": {
    "id": "https://example.org/subjects/123",
    "type": "gx:VatID",
    "gx:vatID": "BE0762747721",
    "gx:countryCode": "BE"
  },
  "evidence": {
    "gx:evidenceOf": "VAT_ID",
    "gx:evidenceURL": "http://ec.europa.eu/taxation_customs/vies/services/checkVatService",
    "gx:executionDate": "2024-11-12T15:15:18.984+00:00"
  }
}
```

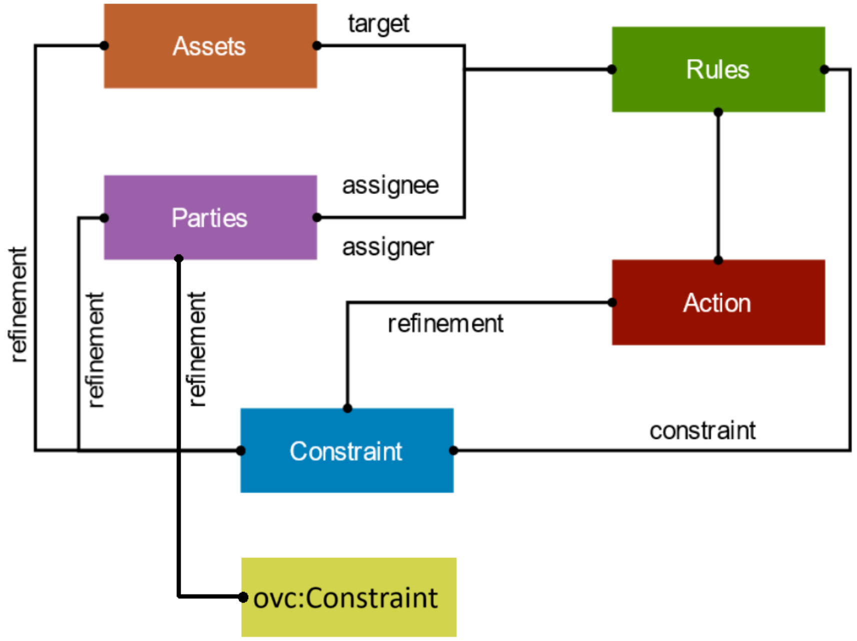
```
eyJhbGciOiJSUzI1NiIsInR5cCI6InZjK2xkK2p
zb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsIm
lzcyI6ImRpZDp3ZWI6YmFrZXVwLmlvIiwia2lkI
joiZGlkOndlYjpiYWtldXAuaW8jUlNBLTIwMjUi
fQ.eyJAY29udGV4dCI6WyJodHRwczovL3d3dy53
My5vcmcvbnMvY3JlZGVudGlhbHMvdjIiLCJodHR
wczovL3d3dy53My5vcmcvbnMvY3JlZGVudGlhbH
MvZXhhbXBsZXMvdjIiXSwidHlwZSI6IlZlcmlma
```

Using Verifiable Credential Claims within a policy definition

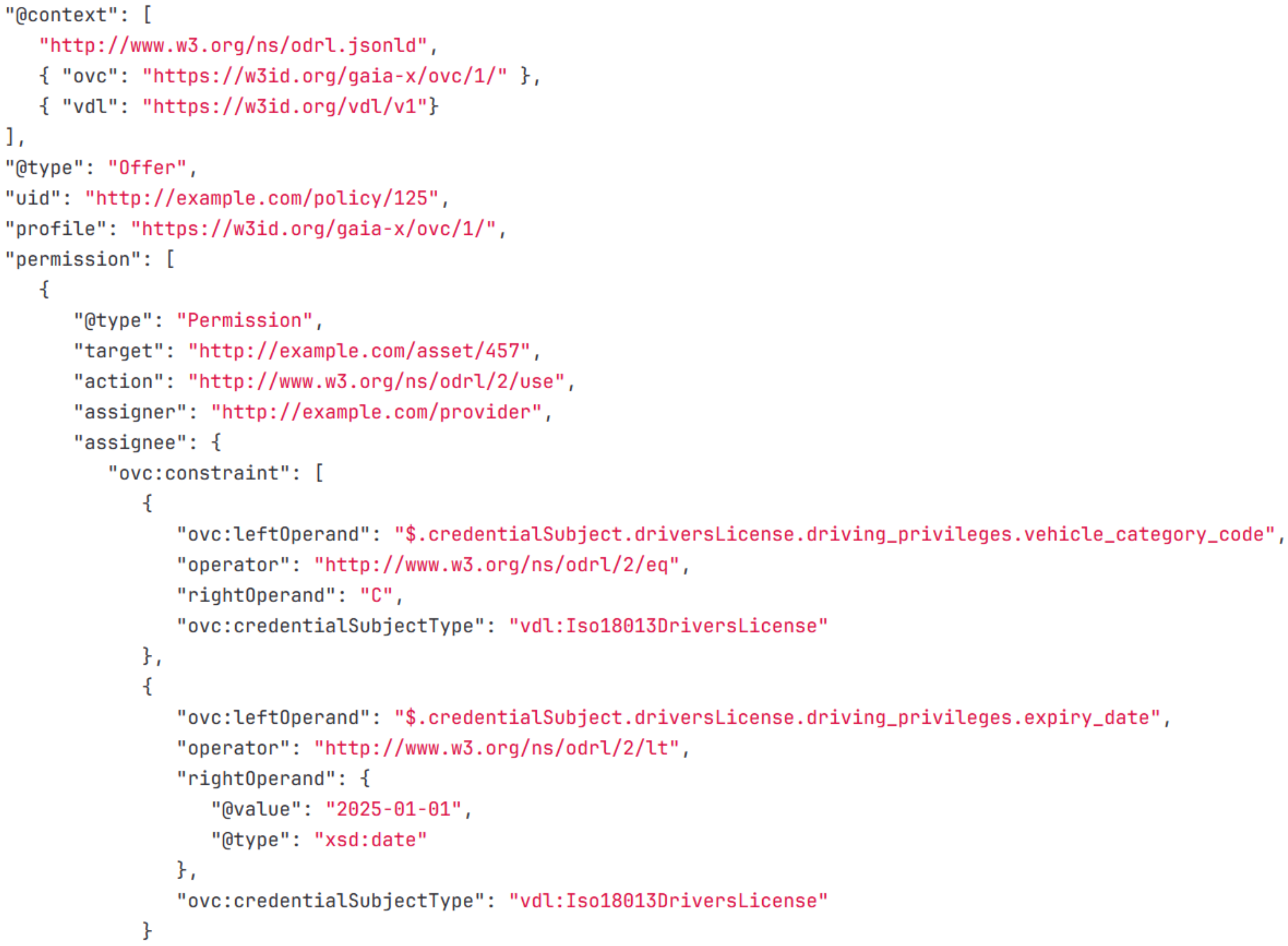
- The Open Digital Rights Language (ODRL) is a policy expression language that provides a flexible and interoperable information model for representing statements about the usage of data and services.
  - But…
- There is no easy way to verify and assess an access request in a trustworthy verifiable manner from that same policy, especially using a software component.
  - So why not leverage Verifiable Credentials inside an ODRL Policy !

ODRL Profile

- Compatible with the base ODRL Information Model
- Would rely on ODRL, VC and JSONPath base specification
- Give clear definition and syntax for custom constraint values
- Custom value formats for `ovc:leftOperand and` ovc:credentialSubjectType
  - ovc:leftOperand to contain a JSONPath for the intended attribute to evaluate
  - ovc:credentialSubjectType refer to the credential type



Driver’s License example



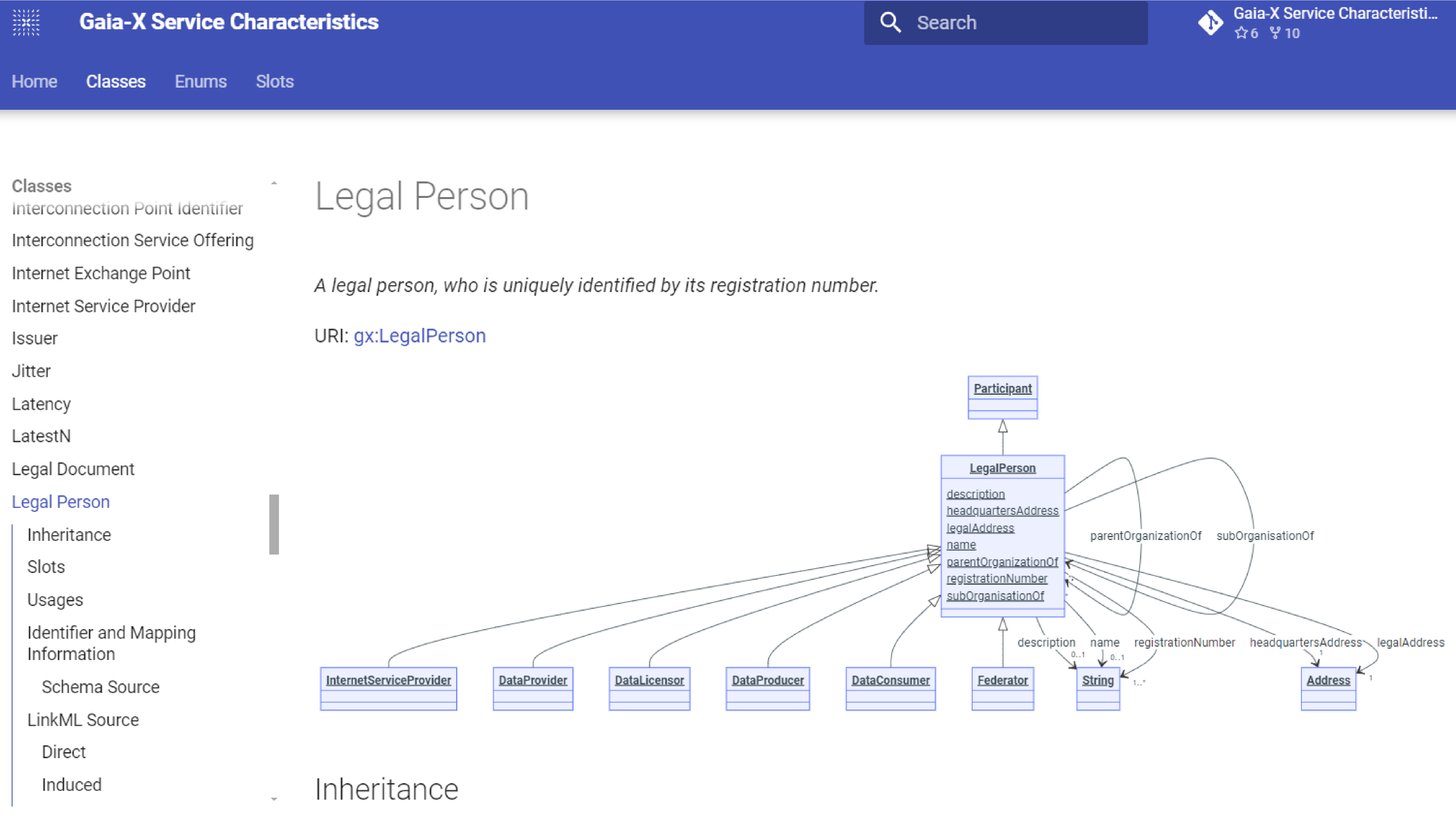
```
"@context": [
   "http://www.w3.org/ns/odrl.jsonld",
   { "ovc": "https://w3id.org/gaia-x/ovc/1/" },
   { "vdl": "https://w3id.org/vdl/v1"}
],
"@type": "Offer",
"uid": "http://example.com/policy/125",
"profile": "https://w3id.org/gaia-x/ovc/1/",
"permission": [
   {
      "@type": "Permission",
      "target": "http://example.com/asset/457",
      "action": "http://www.w3.org/ns/odrl/2/use",
      "assigner": "http://example.com/provider",
      "assignee": {
         "ovc:constraint": [
            {
               "ovc:leftOperand": "$.credentialSubject.driversLicense.driving_privileges.vehicle_category_code",
               "operator": "http://www.w3.org/ns/odrl/2/eq",
               "rightOperand": "C",
               "ovc:credentialSubjectType": "vdl:Iso18013DriversLicense"
            },
            {
               "ovc:leftOperand": "$.credentialSubject.driversLicense.driving_privileges.expiry_date",
               "operator": "http://www.w3.org/ns/odrl/2/lt",
               "rightOperand": {
                  "@value": "2025-01-01",
                  "@type": "xsd:date"
               },
               "ovc:credentialSubjectType": "vdl:Iso18013DriversLicense"
            }
```

Source: https://www.iso.org/standard/69084.html

Use cases

- A catalog only accepting Gaia-X compliant participants (Using the Gaia-X Compliance VC)
- A provider giving access to only certain countries/regions (Using Gaia-X Legal Person VC)
- A company giving access to employees (Right delegation)
- The government giving access to only certified profesionnals (Using a driving license VC)
- An access right given only to Gaia-X members (Using the Gaia-X membership VC)
- A website checking the age of a user

I AM USING ODRL
WE ARE INTEROPERABLE
RIGHT ?
imgflip.com



Gaia-X Service Characteristics
Search
Gaia-X Service Characteristi...
6
10
Home
Classes
Enums
Slots

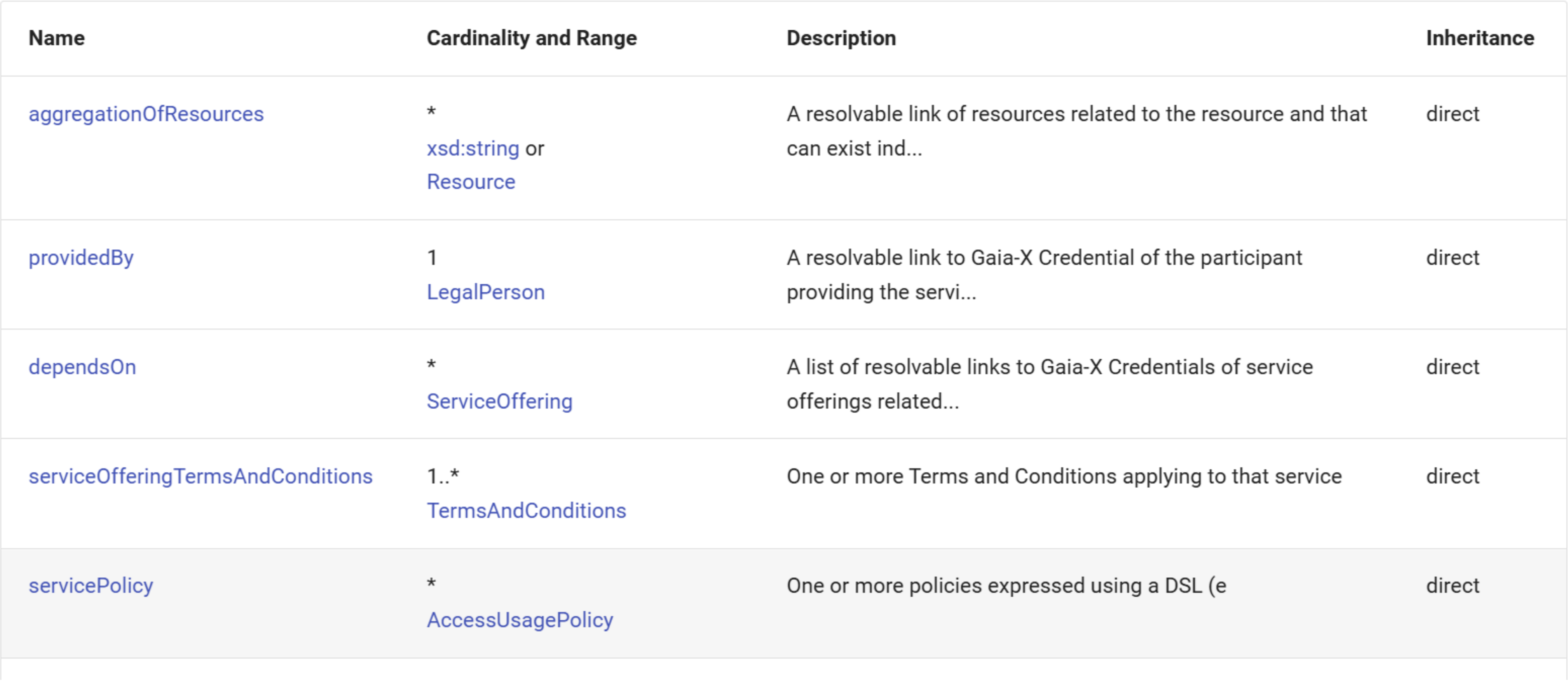
Legal Person

A legal person, who is uniquely identified by its registration number.

URI: gx:LegalPerson

Inheritance

Gaia-X Service Offering



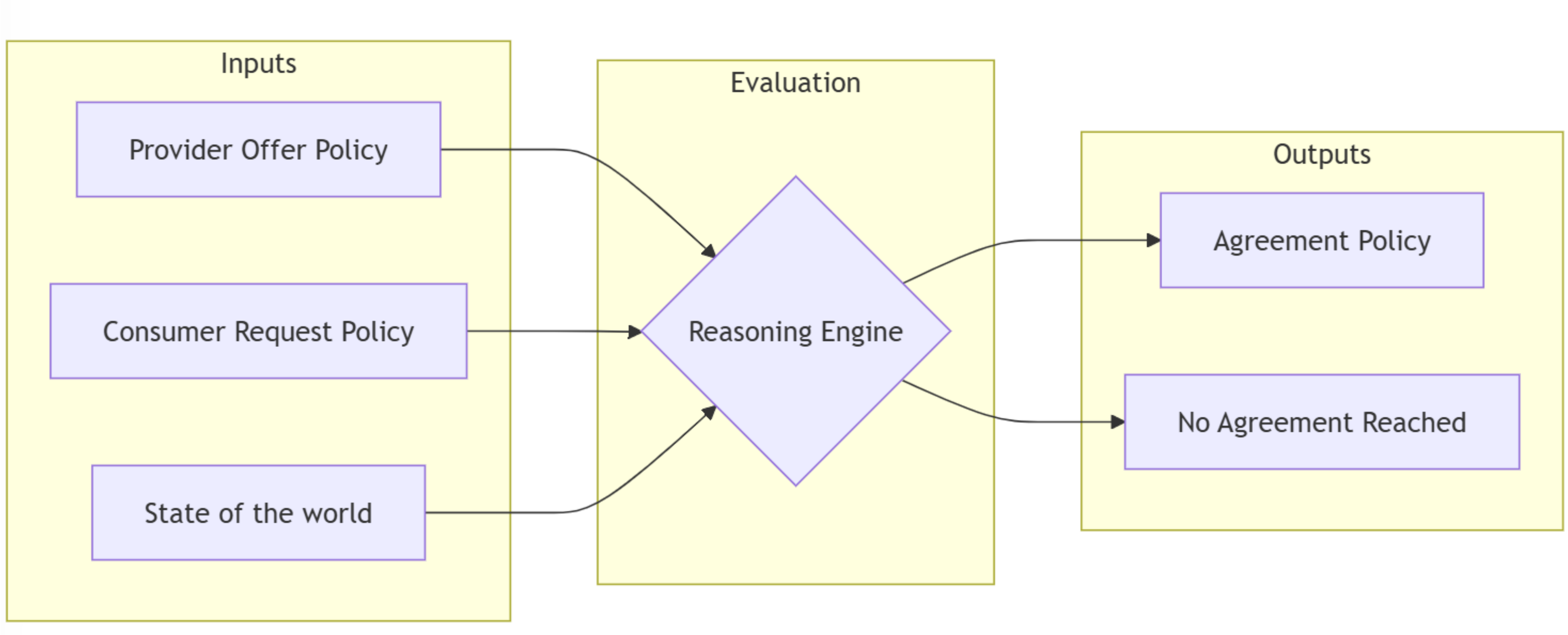
Name	Cardinality and Range	Description	Inheritance
aggregationOfResources	* xsd:string or Resource	A resolvable link of resources related to the resource and that can exist ind...	direct
providedBy	1 LegalPerson	A resolvable link to Gaia-X Credential of the participant providing the servi...	direct
dependsOn	* ServiceOffering	A list of resolvable links to Gaia-X Credentials of service offerings related...	direct
serviceOfferingTermsAndConditions	1..* TermsAndConditions	One or more Terms and Conditions applying to that service	direct
servicePolicy	* AccessUsagePolicy	One or more policies expressed using a DSL (e	direct

THIS ODRL POLICIES SEEM NICE
I BET IT'S EASY TO IMPLEMENT
imgflip.com

The issue with ODRL

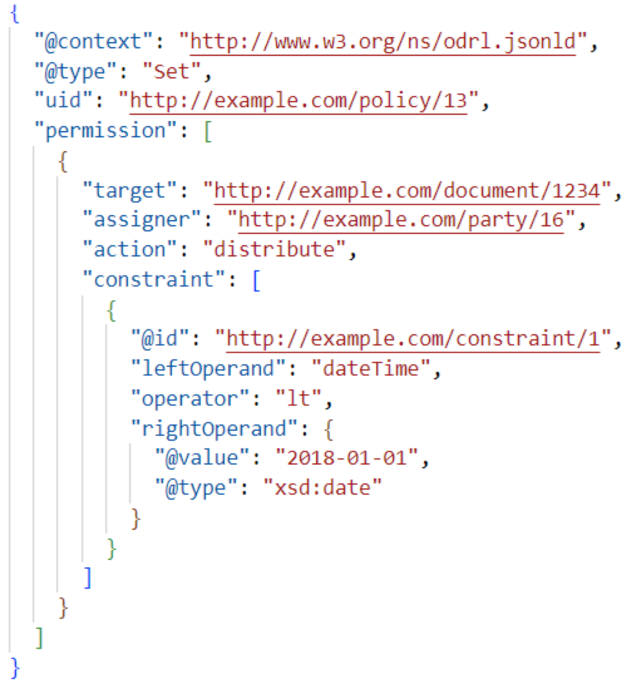
- Not easy to implement
- Potential performance issues
- Hard to be completely generic and cover all possible cases
- Multiple implementations may not perform the same way
- How can an ODRL Evaluator be predictable and trustworthy

Evaluation Engine

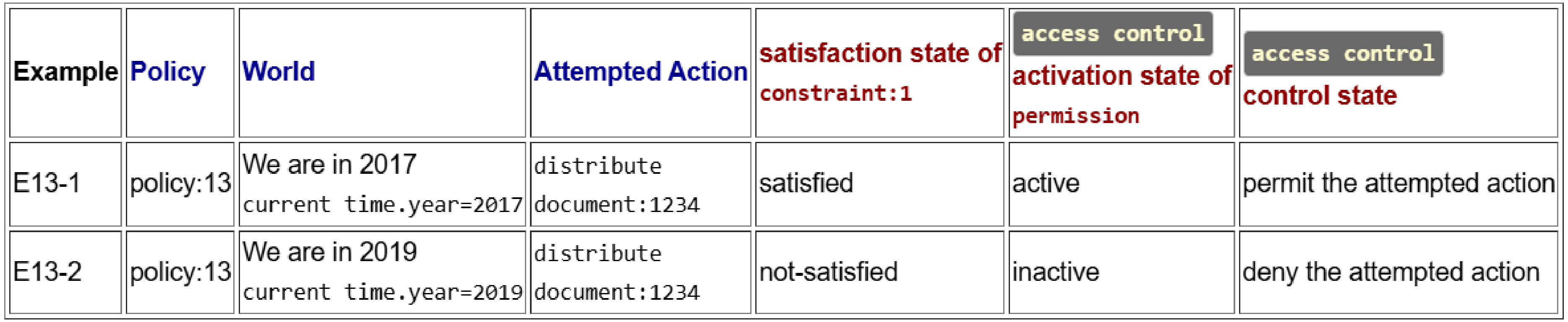


ODRL Formal Semantics

- Access control scenario
- Policy monitoring scenario



```
{
  "@context": "http://www.w3.org/ns/odrl.jsonld",
  "@type": "Set",
  "uid": "http://example.com/policy/13",
  "permission": [
    {
      "target": "http://example.com/document/1234",
      "assigner": "http://example.com/party/16",
      "action": "distribute",
      "constraint": [
        {
          "@id": "http://example.com/constraint/1",
          "leftOperand": "dateTime",
          "operator": "lt",
          "rightOperand": {
            "@value": "2018-01-01",
            "@type": "xsd:date"
          }
        }
      ]
    }
  ]
}
```



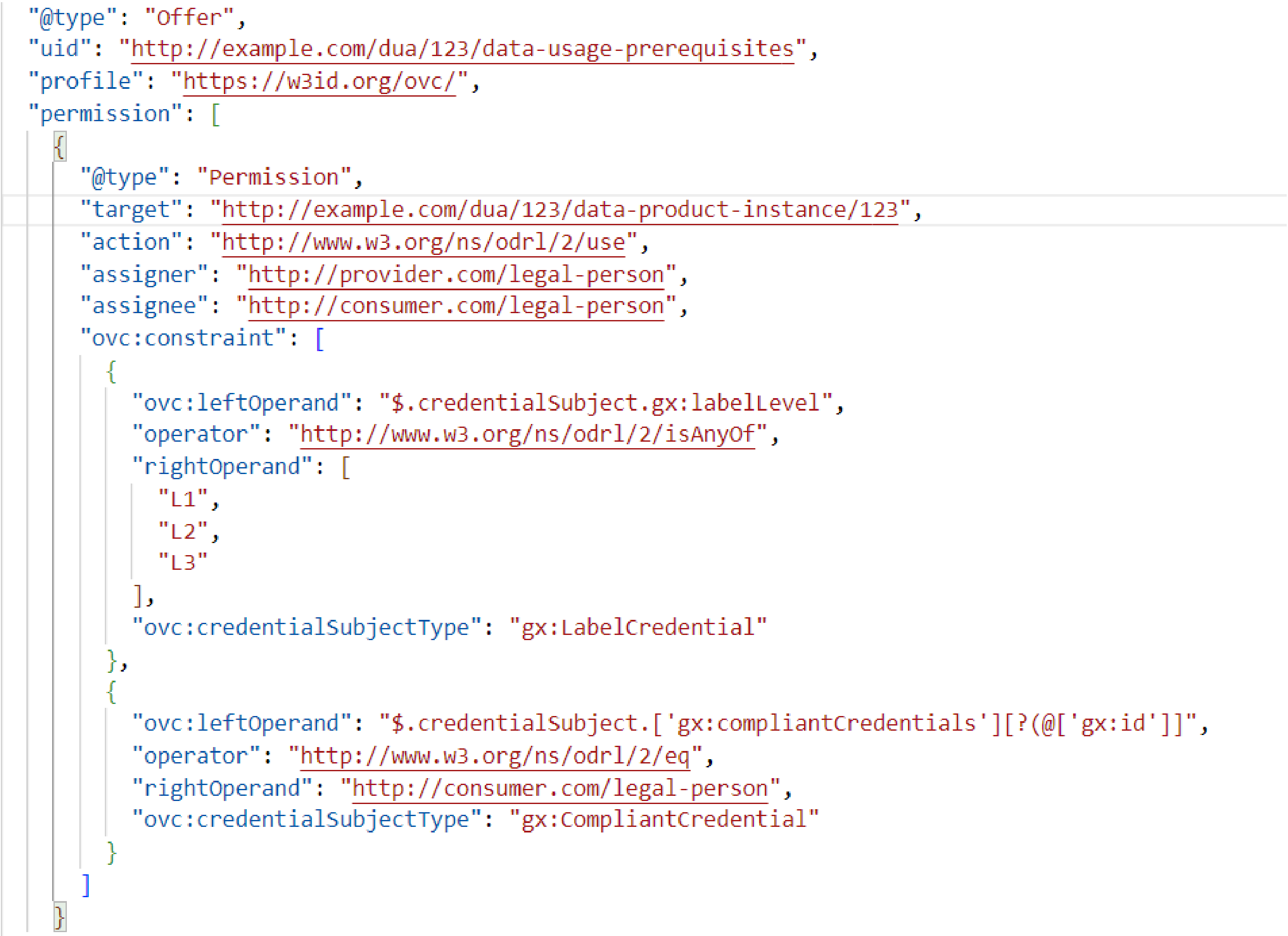
Example	Policy	World	Attempted Action	satisfaction state of constraint:1	access control activation state of permission	access control control state
E13-1	policy:13	We are in 2017 current time.year=2017	distribute document:1234	satisfied	active	permit the attempted action
E13-2	policy:13	We are in 2019 current time.year=2019	distribute document:1234	not-satisfied	inactive	deny the attempted action

Source: https://w3c.github.io/odrl/formal-semantics/#example-of-a-constrained-permission

Why verifiable credentials are ideal in this case

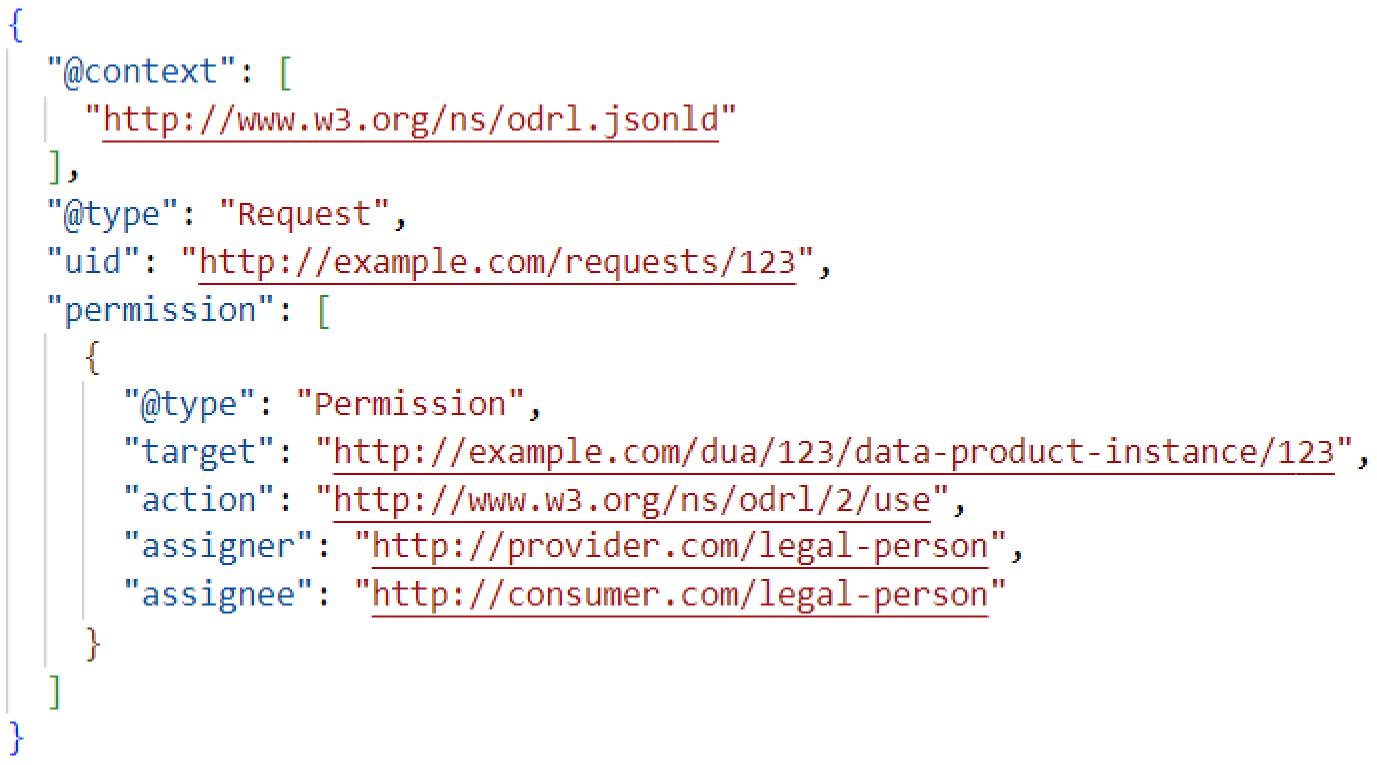
- Trustworthy claims
- Easy to target an attribute
- You can choose who to trust or not
- No need to reinvent the wheel
  - Signature verification
  - Issuer validation
  - OID4VP protocol

Use Case: ODRL Offer



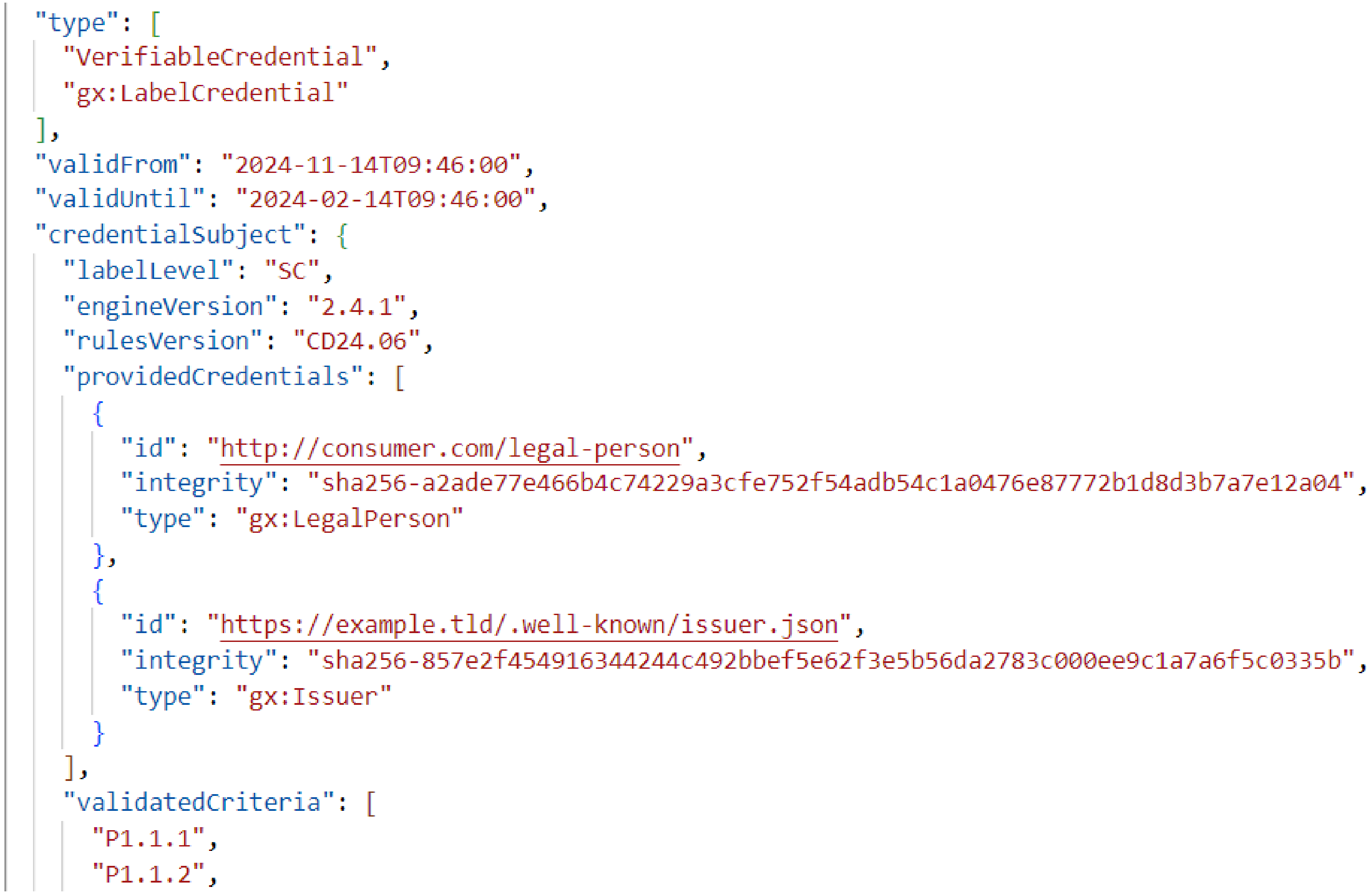
```
"@type": "Offer",
"uid": "http://example.com/dua/123/data-usage-prerequisites",
"profile": "https://w3id.org/ovc/",
"permission": [
  {
    "@type": "Permission",
    "target": "http://example.com/dua/123/data-product-instance/123",
    "action": "http://www.w3.org/ns/odrl/2/use",
    "assigner": "http://provider.com/legal-person",
    "assignee": "http://consumer.com/legal-person",
    "ovc:constraint": [
      {
        "ovc:leftOperand": "$.credentialSubject.gx:labelLevel",
        "operator": "http://www.w3.org/ns/odrl/2/isAnyOf",
        "rightOperand": [
          "L1",
          "L2",
          "L3"
        ],
        "ovc:credentialSubjectType": "gx:LabelCredential"
      },
      {
        "ovc:leftOperand": "$.credentialSubject.['gx:compliantCredentials'][?(@['gx:id']]",
        "operator": "http://www.w3.org/ns/odrl/2/eq",
        "rightOperand": "http://consumer.com/legal-person",
        "ovc:credentialSubjectType": "gx:CompliantCredential"
      }
    ]
  }
```

Use Case: ODRL Request



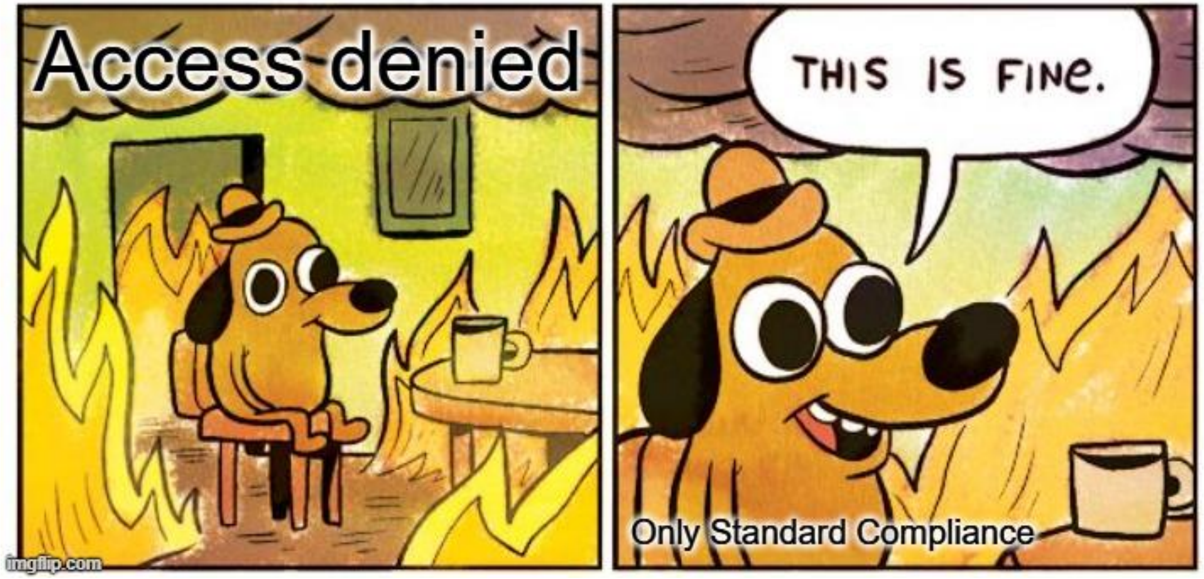
```
{
  "@context": [
    "http://www.w3.org/ns/odrl.jsonld"
  ],
  "@type": "Request",
  "uid": "http://example.com/requests/123",
  "permission": [
    {
      "@type": "Permission",
      "target": "http://example.com/dua/123/data-product-instance/123",
      "action": "http://www.w3.org/ns/odrl/2/use",
      "assigner": "http://provider.com/legal-person",
      "assignee": "http://consumer.com/legal-person"
    }
  ]
}
```

Use Case: State of the world (consumer's credentials)



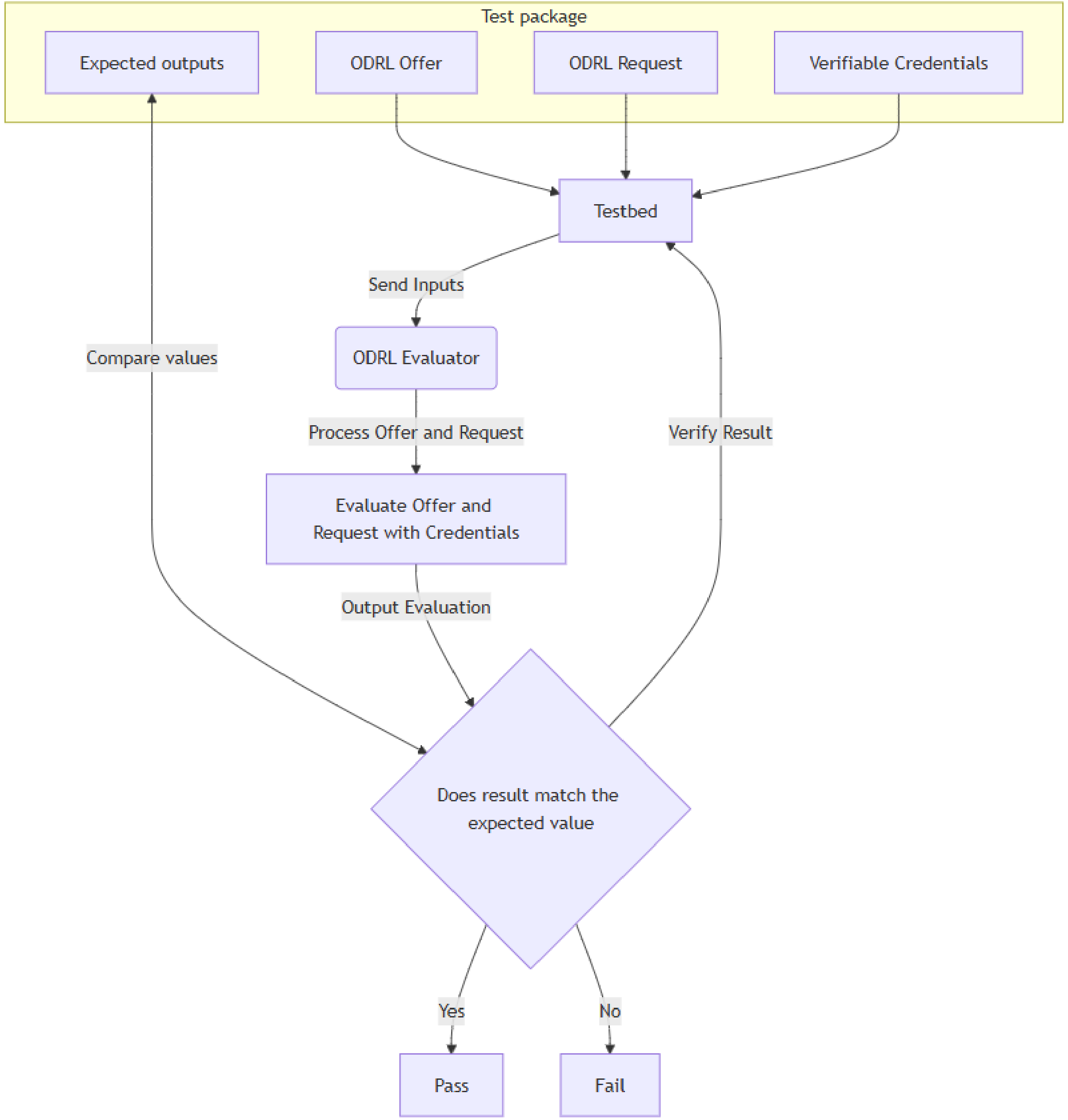
```
"type": [
  "VerifiableCredential",
  "gx:LabelCredential"
],
"validFrom": "2024-11-14T09:46:00",
"validUntil": "2024-02-14T09:46:00",
"credentialSubject": {
  "labelLevel": "SC",
  "engineVersion": "2.4.1",
  "rulesVersion": "CD24.06",
  "providedCredentials": [
    {
      "id": "http://consumer.com/legal-person",
      "integrity": "sha256-a2ade77e466b4c74229a3cfe752f54adb54c1a0476e87772b1d8d3b7a7e12a04",
      "type": "gx:LegalPerson"
    },
    {
      "id": "https://example.tld/.well-known/issuer.json",
      "integrity": "sha256-857e2f454916344244c492bbef5e62f3e5b56da2783c000ee9c1a7a6f5c0335b",
      "type": "gx:Issuer"
    }
  ],
  "validatedCriteria": [
    "P1.1.1",
    "P1.1.2",
```

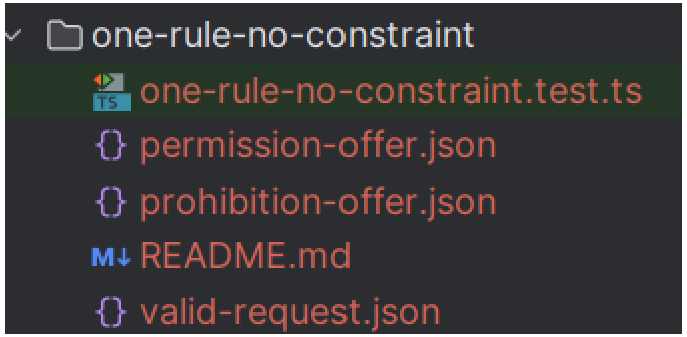
Use Case: Results



What if we start with use cases ?

- All we need is :
  - Quick description of the use case
  - An ODRL Offer(s)
  - An ODRL Request(s)
  - A possible state of the world (credentials)
  - Expected outcome(s)
- Can be used as a Test-Driven Development approach
- Adopt KISS (no need for evaluation to be more complex than required)
- It’s essential to foster contribution and collaboration
- Start by covering the 20/80 rule





```
one-rule-no-constraint
    one-rule-no-constraint.test.ts
    permission-offer.json
    prohibition-offer.json
    README.md
    valid-request.json
```

https://gitlab.com/gaia-x/lab/policy-reasoning/odrl-toolbox

You need to test it
It's working with my ecosystem
imgflip.com

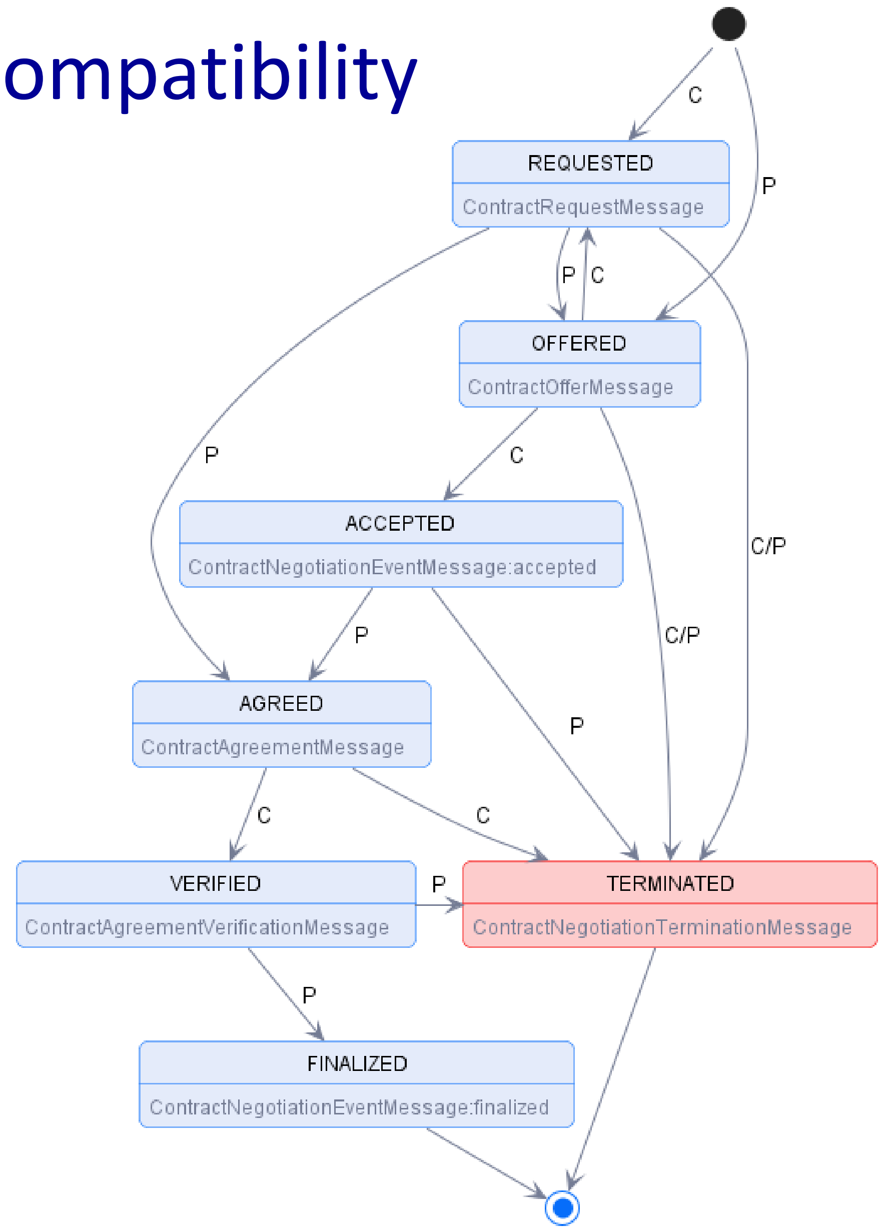
Why do we need a testbed ?

- Might showcase a limitation/misconception about the specification
- Might show possible flows when testing with complex cases
- It can be a way to issue a “compliance” for implementations
  - Which would ensure trust in the software (Predictable)
  - Have to add an extra layer of trust while executing in a TEE (Trusted Execution Environment)
- Ensure interoperability between ecosystems

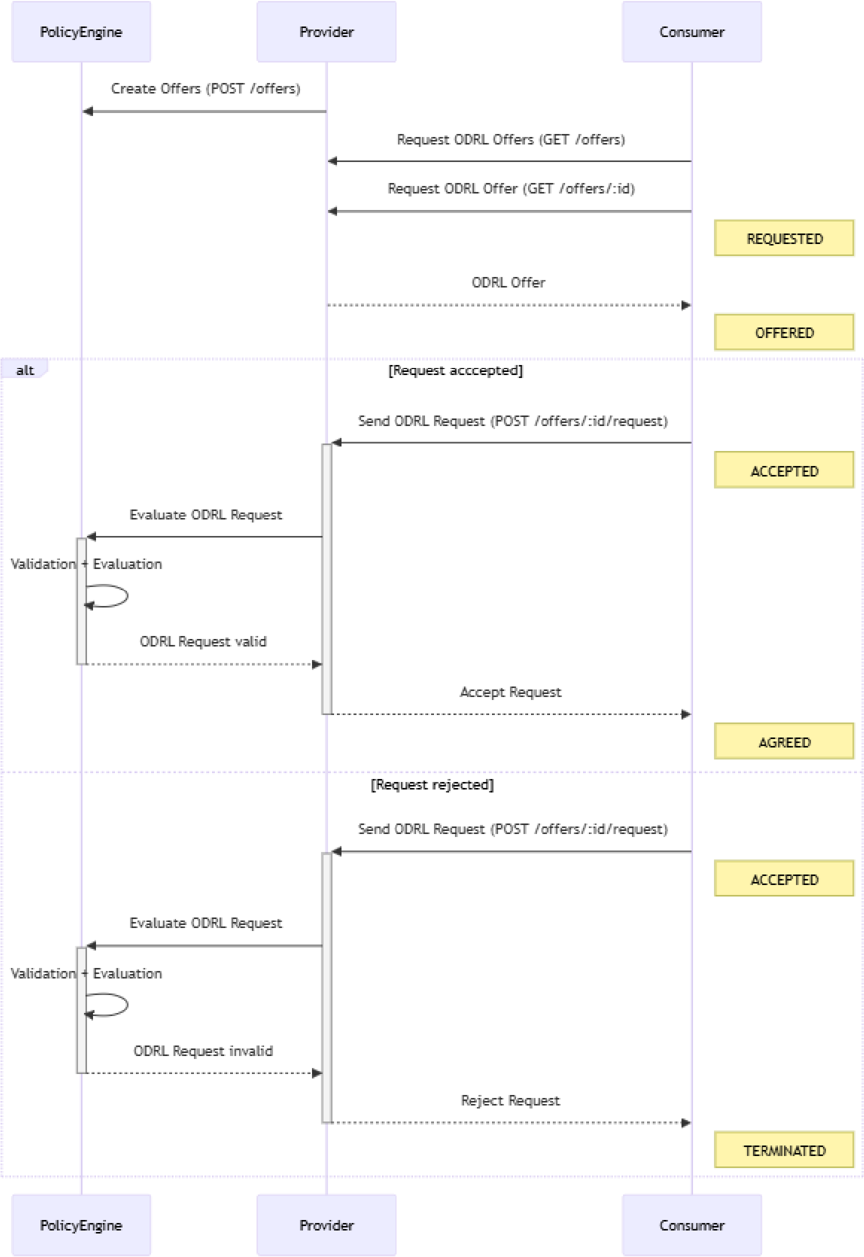
What’s coming next in the ODRL toolbox ?

- More and more uses case packages
- All contributions are welcome
- Better validation using SHACL
- Modular extensible policy evaluation
- More user-friendly policy generation

Dataspace Protocol compatibility


C
P
REQUESTED
ContractRequestMessage
P
C
OFFERED
ContractOfferMessage
C
P
C/P
ACCEPTED
ContractNegotiationEventMessage:accepted
C/P
P
AGREED
ContractAgreementMessage
P
C
C
VERIFIED
ContractAgreementVerificationMessage
P
TERMINATED
ContractNegotiationTerminationMessage
P
FINALIZED
ContractNegotiationEventMessage:finalized

Source: https://docs.internationaldataspaces.org/ids-knowledgebase/dataspace-protocol/contract-negotiation/contract.negotiation.protocol



PolicyEngine

Provider

Consumer

Create Offers (POST /offers)

Request ODRL Offers (GET /offers)

Request ODRL Offer (GET /offers/:id)

REQUESTED

ODRL Offer

OFFERED

alt

[Request acccepted]

Send ODRL Request (POST /offers/:id/request)

ACCEPTED

Evaluate ODRL Request

Validation + Evaluation

ODRL Request valid

Accept Request

AGREED

[Request rejected]

Send ODRL Request (POST /offers/:id/request)

ACCEPTED

Evaluate ODRL Request

Validation + Evaluation

ODRL Request invalid

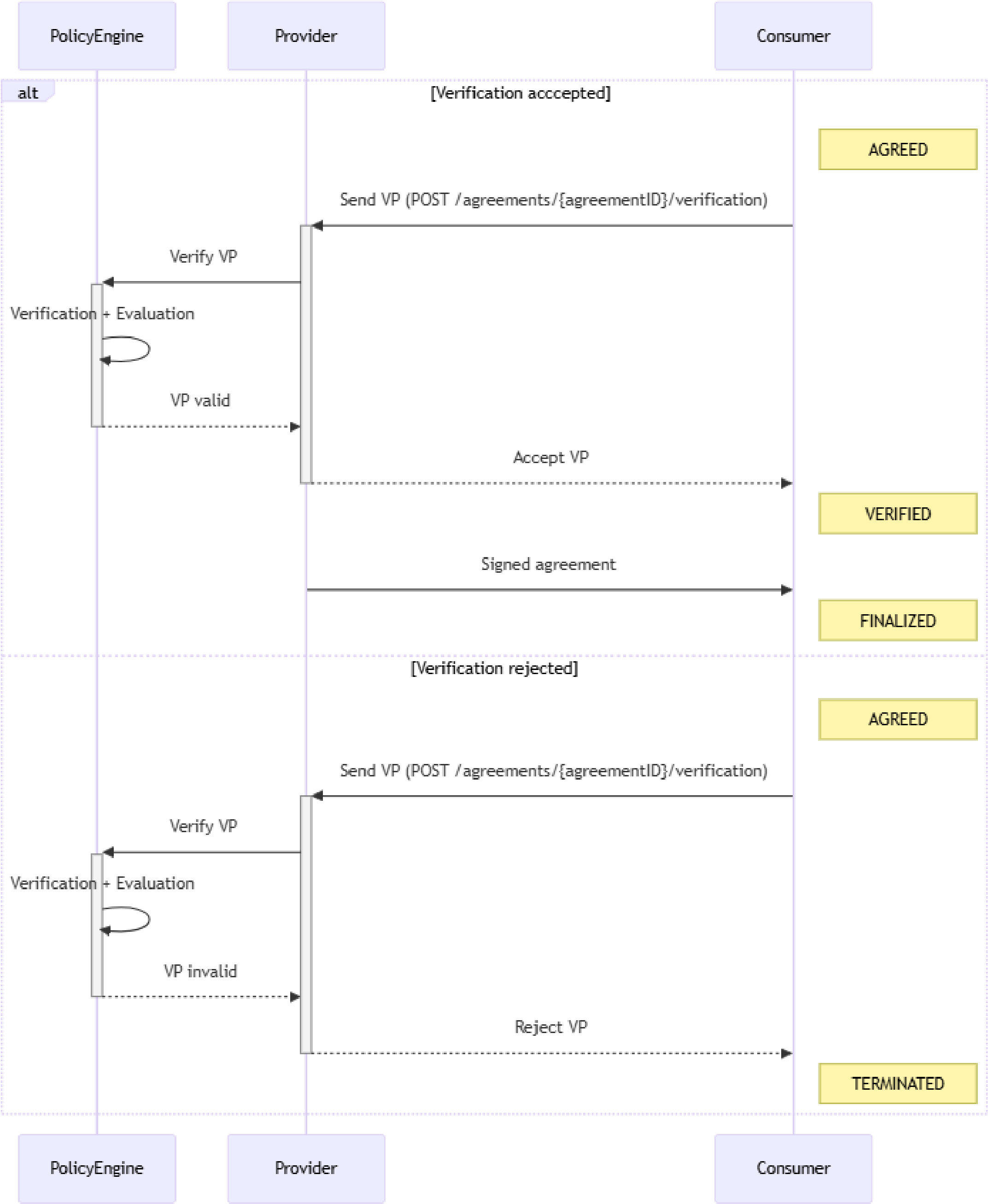
Reject Request

TERMINATED

PolicyEngine

Provider

Consumer



PolicyEngine

Provider

Consumer

alt

[Verification acccepted]

AGREED

Send VP (POST /agreements/{agreementID}/verification)

Verify VP

Verification + Evaluation

VP valid

Accept VP

VERIFIED

Signed agreement

FINALIZED

[Verification rejected]

AGREED

Send VP (POST /agreements/{agreementID}/verification)

Verify VP

Verification + Evaluation

VP invalid

Reject VP

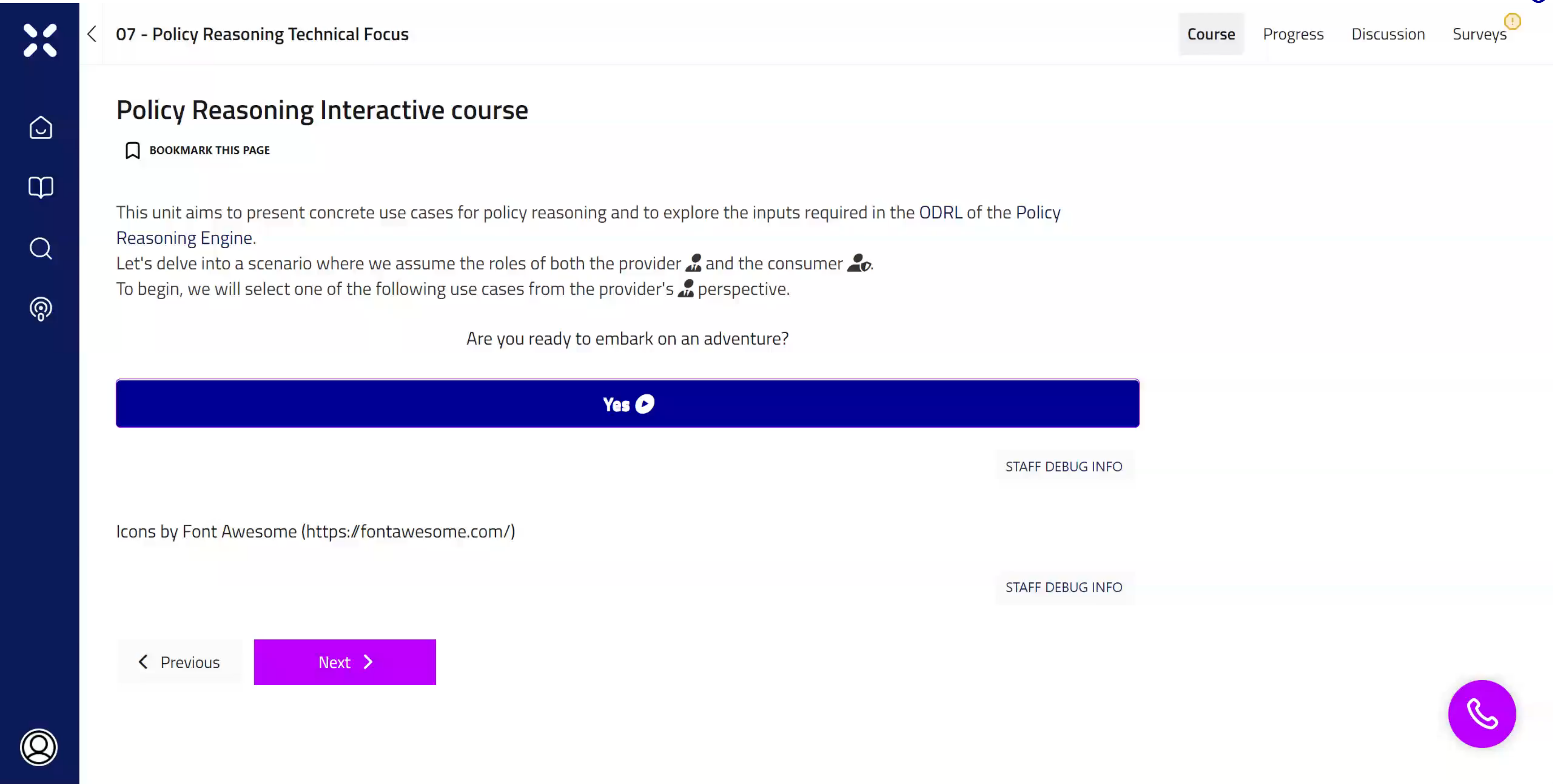
TERMINATED

PolicyEngine

Provider

Consumer

Policy Reasoning course on the Gaia-X Academy



Policy Reasoning Interactive course

This unit aims to present concrete use cases for policy reasoning and to explore the inputs required in the ODRL of the Policy Reasoning Engine.
Let's delve into a scenario where we assume the roles of both the provider and the consumer .
To begin, we will select one of the following use cases from the provider's perspective.

Are you ready to embark on an adventure?

Icons by Font Awesome (https://fontawesome.com/)

Try it yourself !

Thank you!

Yassir SELLAMI, yassir.sellami@gaia-x.eu

Open discussion/Q&A with the Gaia-X Tech Team

14:30 – 15:00

CTO Team