OWL

Web Ontology Language
Part 1

OWL Concept vocabulary

Core ontology-level classes

- **owl:Ontology** – denotes an ontology document.
- **owl:OntologyProperty** – properties about ontologies (like `owl:imports`).

Class constructors

- **owl:Class** – generic class.
- **owl:Restriction** – restrictions on properties.
- **owl:Thing** – the universal class (everything).
- **owl:Nothing** – the empty class.
- **owl:unionOf**, **intersectionOf**, **complementOf**, **oneOf** – set-theoretic class constructors.

Individuals

- **owl:NamedIndividual** – explicitly named individual.

Properties

- **owl:ObjectProperty** – property between individuals.
- **owl:DatatypeProperty** – property between individual and literal.
- **owl:AnnotationProperty** – property used for metadata.
- **owl:TransitiveProperty**, **SymmetricProperty**, **FunctionalProperty**, **InverseFunctionalProperty**, **ReflexiveProperty**, **IrreflexiveProperty**, **AsymmetricProperty** – property characteristics.

Property relationships

- **owl:inverseOf** – inverse properties.
- **owl:equivalentProperty** – property equivalence.
- **owl:propertyChainAxiom** – define a property as a chain of others.
- **owl:propertyDisjointWith** – disjoint properties.

OWL Concept vocabulary

Class relationships

- **owl:equivalentClass** – class equivalence.
- **owl:disjointWith** – class disjointness.
- **owl:disjointUnionOf** – class partition.

Keys

- **owl:hasKey** – declare identifying keys for a class.

Datatypes

- **owl:Datatype** – datatype class.
- **owl:DataRange** – datatype restrictions (e.g., `xsd:integer` between 0–10).
-

Restrictions

OWL uses RDF lists with these properties:

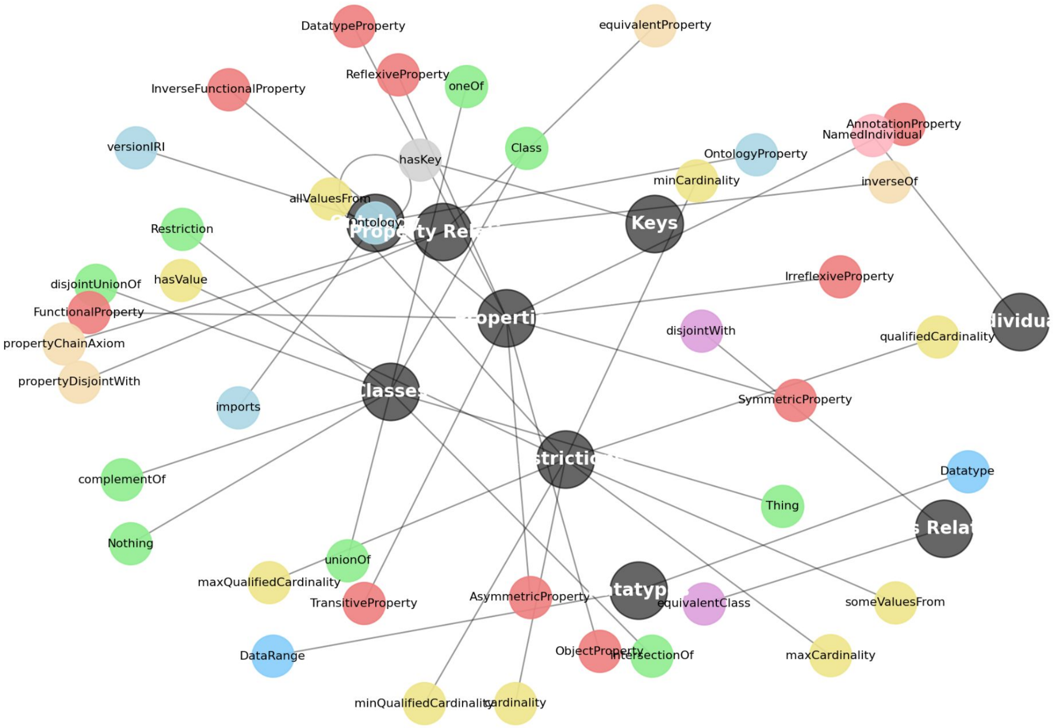
- **owl:allValuesFrom**, **someValuesFrom** – value restrictions.
- **owl:hasValue** – fixed value restriction.
- **owl:minCardinality**, **maxCardinality**, **cardinality** – cardinality restrictions.
- **owl:qualifiedCardinality**, **minQualifiedCardinality**, **maxQualifiedCardinality** – qualified versions.

Ontology management

- **owl:imports** – include another ontology.
- **owl:versionIRI** – ontology version.
- **owl:priorVersion**, **backwardCompatibleWith**, **incompatibleWith** – versioning relations.

OWL Concept vocabulary

OWL 2 Vocabulary Grouped by Function (http://www.w3.org/2002/07/owl#)



OWL ‘species’

OWL 2 Full

- Maximum freedom: any RDF graph is a valid OWL 2 Full ontology.
- No separation of classes/individuals/properties (metamodeling allowed everywhere).
- Incompatible with DL restrictions.
- Reasoning is **undecidable** — no complete DL reasoners.
- Supported only by RDF rule-based reasoners (like Jena).

OWL 2 DL

- Based on Description Logics.
- Enforces syntactic restrictions (strict separation of classes, properties, individuals — except via *punning*).
- Decidable reasoning: complete reasoners exist.
- Used in Protégé + HermiT, Pellet, FaCT++.

OWL ‘profiles’

OWL 2 EL

- Tailored for very large ontologies with many classes (e.g., biomedical vocabularies).
- Polynomial-time reasoning.
- Supports existential restrictions, property chains.
- Used in SNOMED CT, Gene Ontology.

OWL 2 QL

- Optimized for efficient query answering over very large instance data (ABoxes).
- Reasoning can be delegated to standard relational databases.
- Great when the ontology is relatively simple but the dataset is huge.

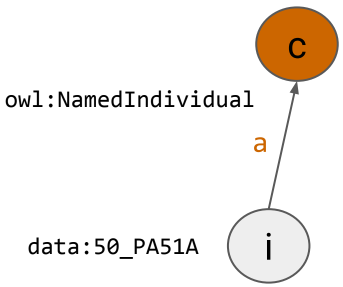
OWL 2 RL

- Designed to be implemented with rule engines.
- Scalable to large datasets.
- Sacrifices some expressivity, but reasoning is forward-chaining rule application.

OWL - Main components

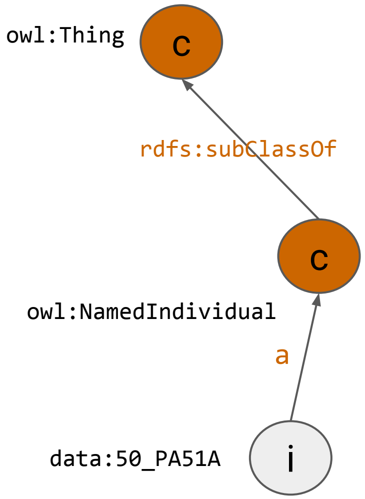
OWL - Main components

Named individual



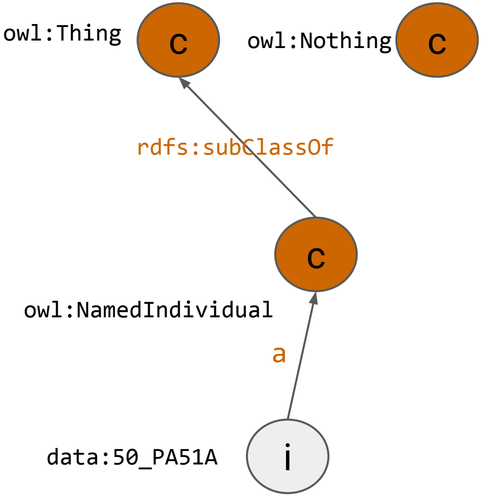
OWL - Main components

Named individual



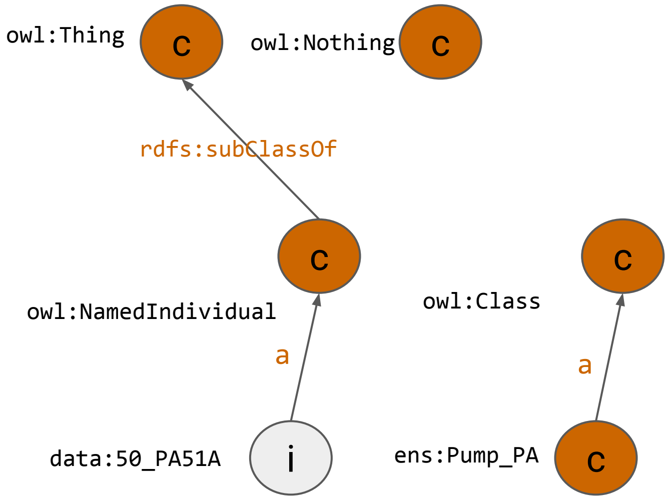
OWL - Main components

Named individuals



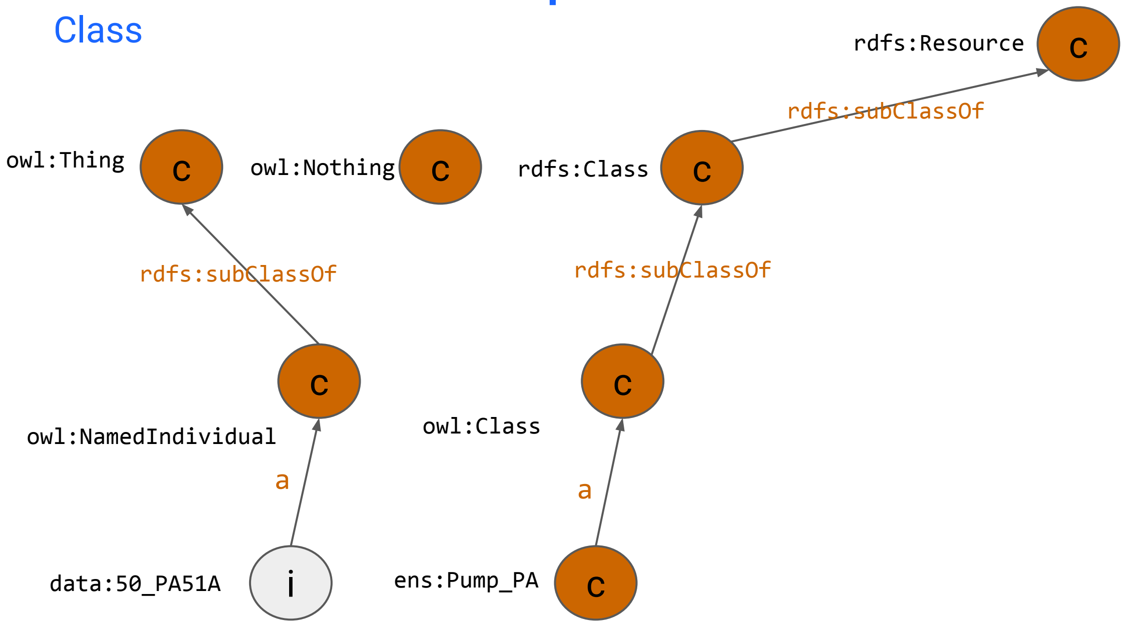
OWL - Main components

Class



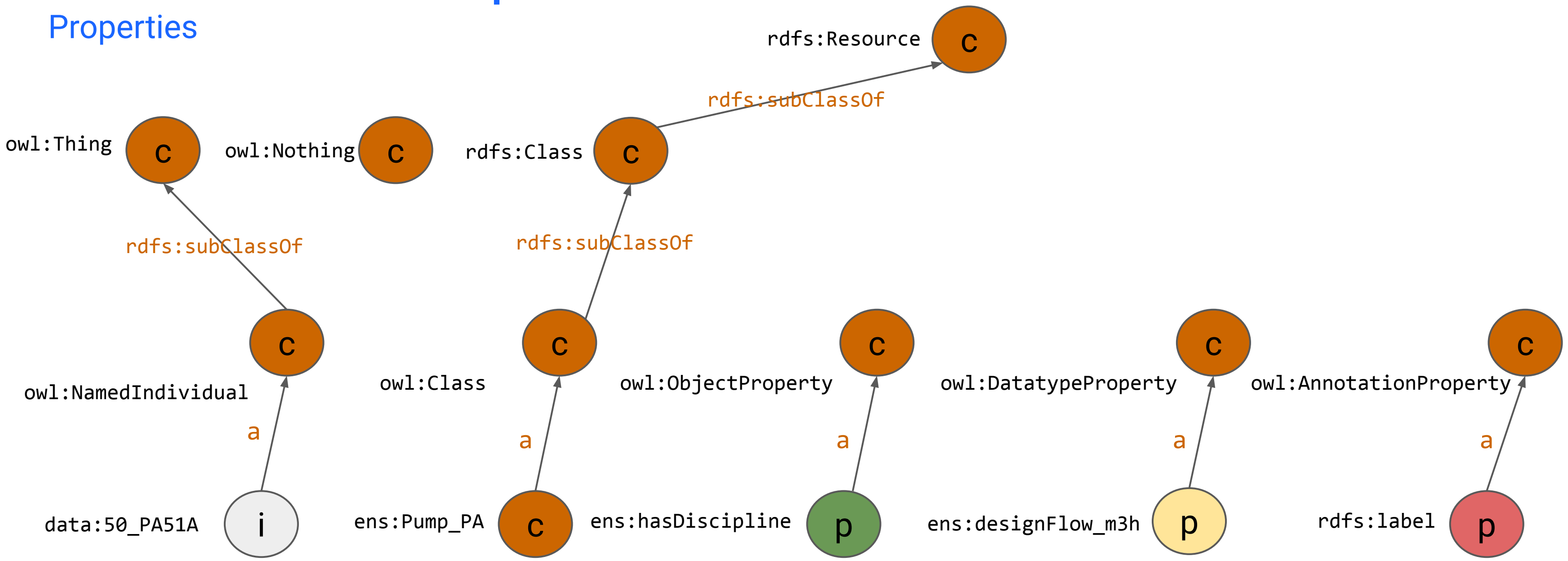
OWL - Main components

Class



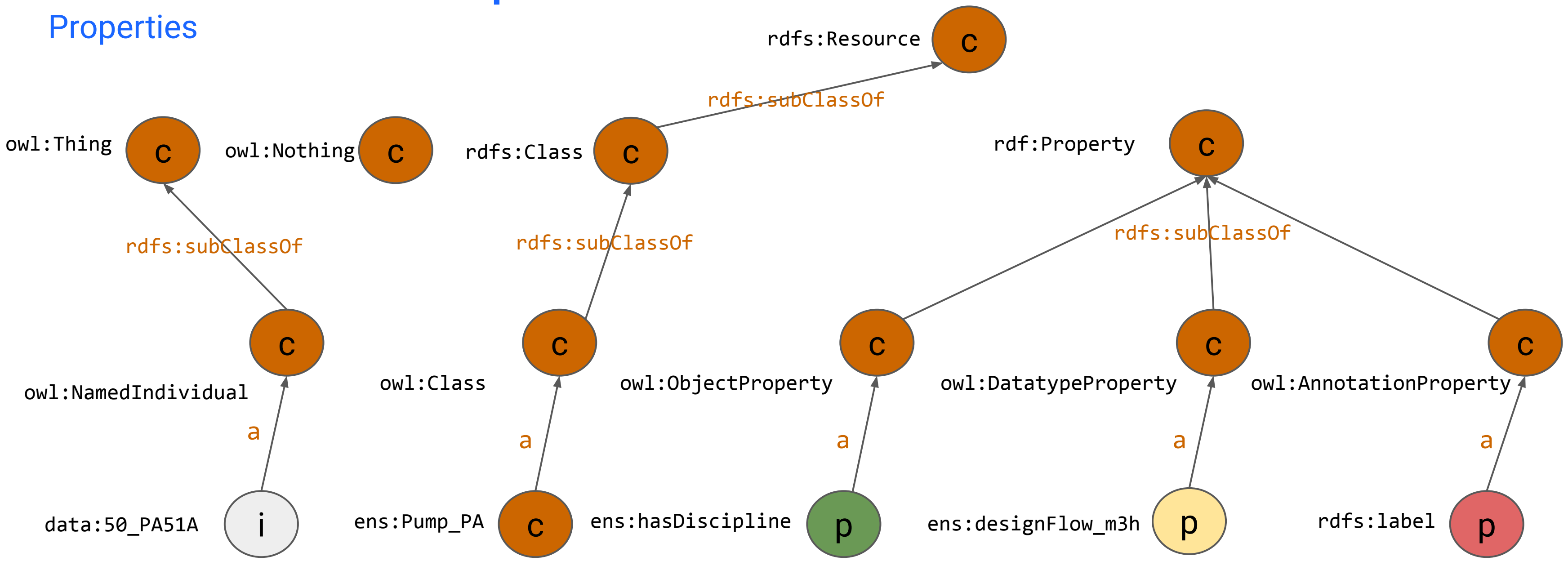
OWL - Main components

Properties



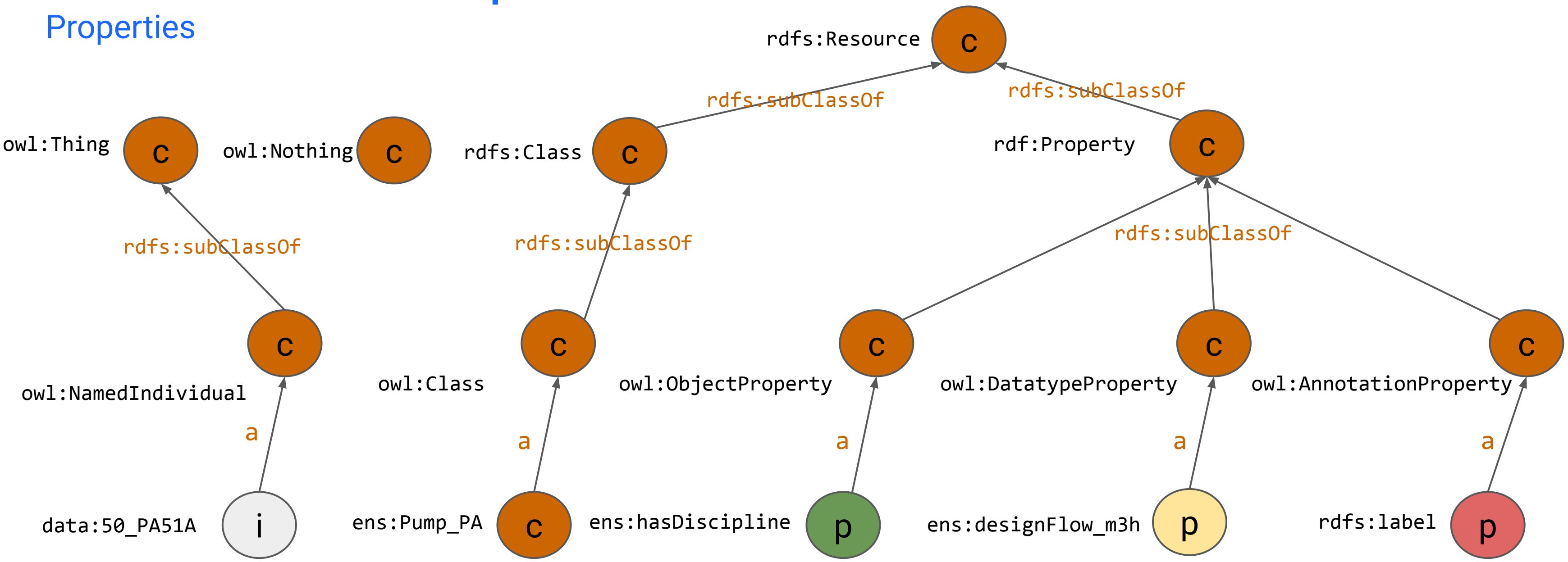
OWL - Main components

Properties



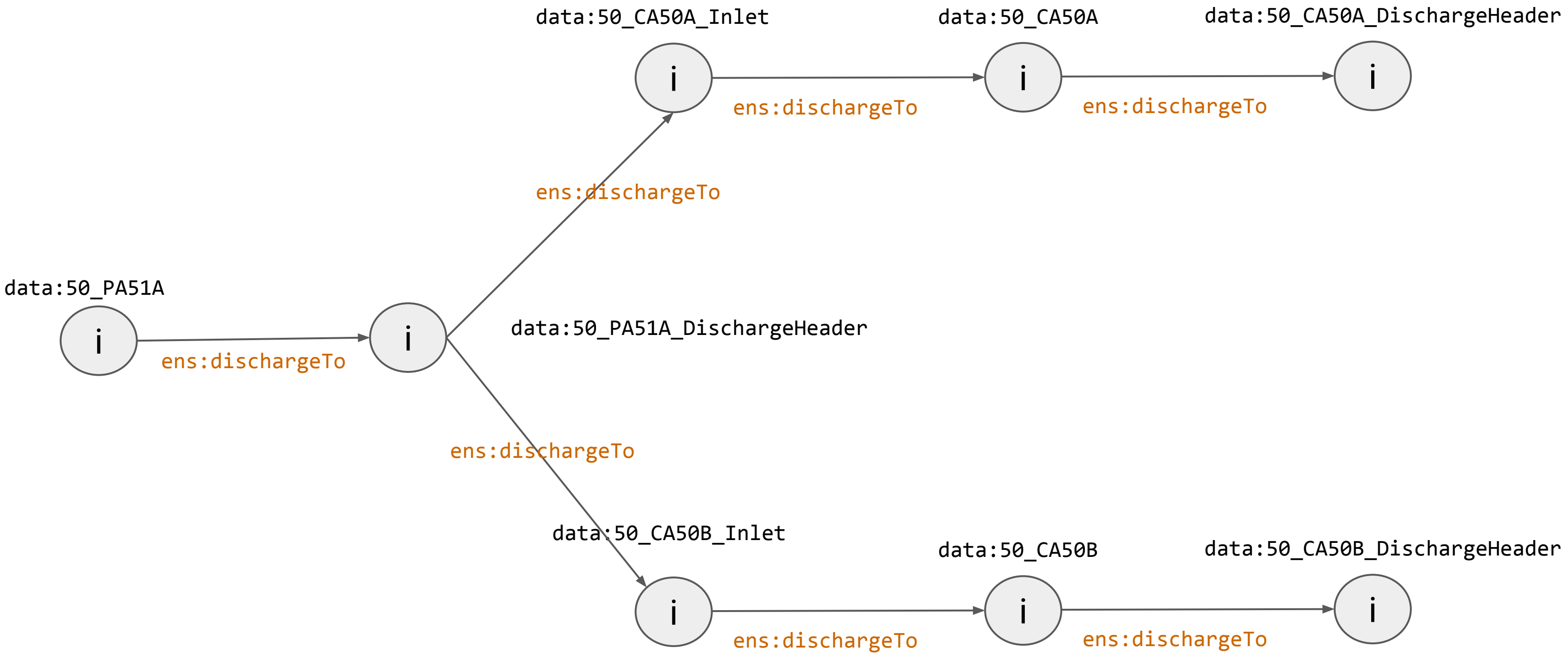
OWL - Main components

Properties



Transitive reasoning

Transitive reasoning



Transitive reasoning

```
# QUERY 1: GENERIC TRANSITIVE CLOSURE
PREFIX owl: <http://www.w3.org/2002/07/owl#>

CONSTRUCT { ?x ?p ?z }
WHERE
{
    ?p a owl:TransitiveProperty .
    ?x ?p ?y .
    ?y ?p ?z .

    FILTER ( ?x != ?z )
}
```

Investigating transitive properties

```
# QUERY 2: TRANSITIVE ELEMENTS FROM A STARTING POINT
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

SELECT DISTINCT ?downstream
WHERE
{
  VALUES ?start { data:50_PH50A }

  ?start (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?downstream .
}
```

Investigating transitive properties

```
# QUERY 3: TRANSITIVE CLOSURE TO A STOPPING POINT
PREFIX data:   <http://www.webstep.no/workshop/data/>
PREFIX ens:    <http://www.webstep.no/workshop/ens/>

SELECT DISTINCT ?upstream
WHERE
{
  VALUES ?stop { data:Sea }

  ?upstream (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?stop .
}
```

Investigating transitive properties

```
# QUERY 4: RETURNING FINAL SINKS
PREFIX data:   <http://www.webstep.no/workshop/data/>
PREFIX ens:    <http://www.webstep.no/workshop/ens/>

SELECT DISTINCT ?sink
WHERE
{
    VALUES ?start { data:50_PH50A }
    ?start (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?sink .

    FILTER NOT EXIST
    {
        ?sink (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?anyFurther .
    }
}
```

Investigating transitive properties

```
# QUERY 5: DOES IT REACH THE SEA?
PREFIX data:   <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

ASK
{
  VALUES ?start { data:50_PH50A }

  ?start (ens:dischargesTo | ens:feeds | ens:reliefTo)*  data:Sea .
}
```

Investigating transitive properties

```
# QUERY 6: CONSTRUCTING THE PATH
PREFIX data:   <http://www.webstep.no/workshop/data/>
PREFIX ens:    <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
    ?s ens:dischargesTo ?o .
}
WHERE
{
  VALUES ?start { data:50_PH50A }

  ?start (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?s .
  ?start (ens:dischargesTo | ens:feeds | ens:reliefTo)*  ?o .

  ?s (ens:dischargesTo | ens:feeds | ens:reliefTo)  ?o .
}
```

Transitive reasoning

Reasoning time

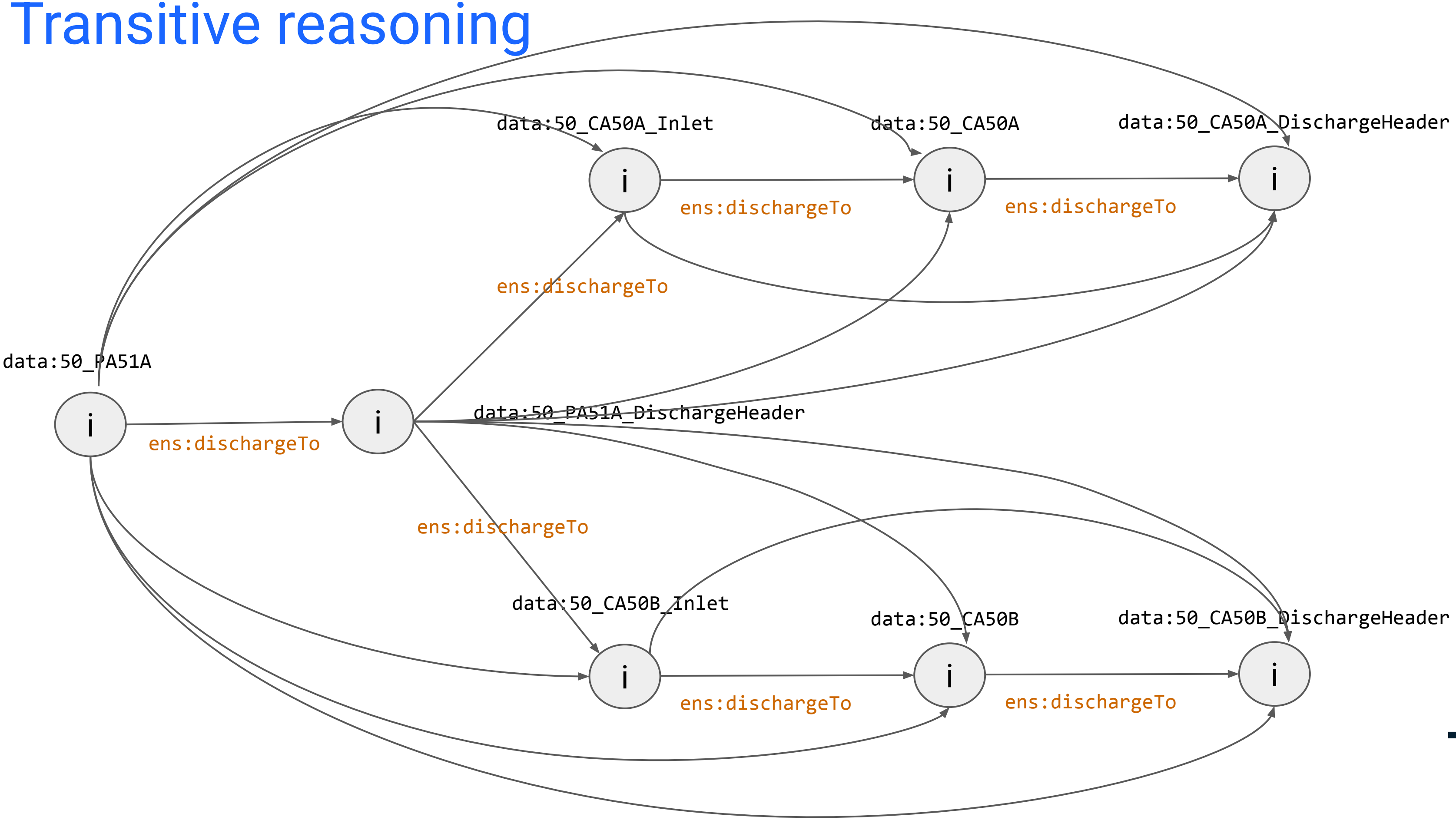
Transitive reasoning

```
# QUERY 7: TRANSITIVE ELEMENTS FROM A STARTING POINT
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema#>
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

SELECT DISTINCT ?downstream
WHERE
{
  VALUES ?start { data:50_PH50A }

  ?start ens:dischargesTo  ?downstream .
}
```

Transitive reasoning

```
# QUERY 10: CONSTRUCTING THE PATH - ATTEMPT 1
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
    ?s ens:dischargesTo ?o .
}
WHERE
{
  VALUES ?start { data:50_PH50A }

  ?start ens:dischargesTo*  ?s .
  ?start ens:dischargesTo  ?o .

  ?s ens:dischargesTo  ?o .
}
```

Transitive reasoning



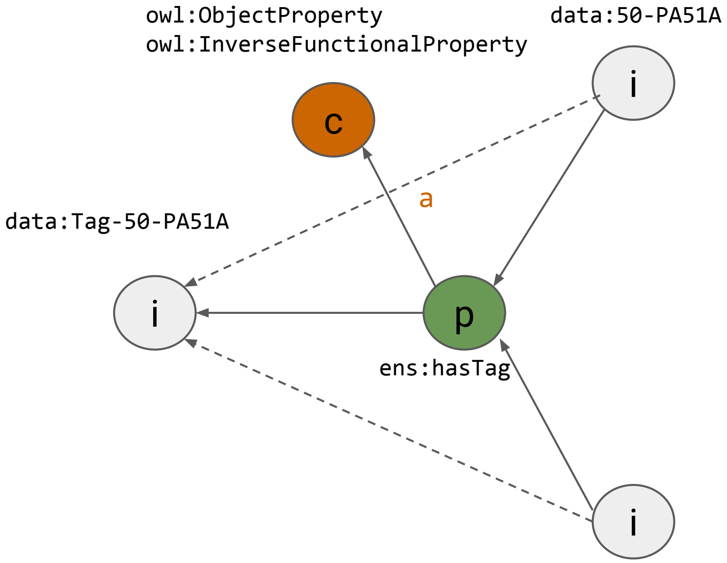
Transitive reasoning

```
# QUERY 10: CONSTRUCTING THE PATH - ATTEMPT 2
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
      ?s ens:dischargesTo ?o .
}
WHERE
{
  VALUES ?start { data:50_PH50A }

  ?start ens:dischargesTo*  ?s .
  ?start ens:dischargesTo  ?o .

  ?s ens:dischargesTo  ?o .
  FILTER (?s != ?o)

  FILTER NOT EXISTS {
      ?s ens:dischargesTo ?mid .
    ?mid ens:dischargesTo ?o .
    FILTER ( ?mid != ?s && ?mid != ?o )
  }
}
```

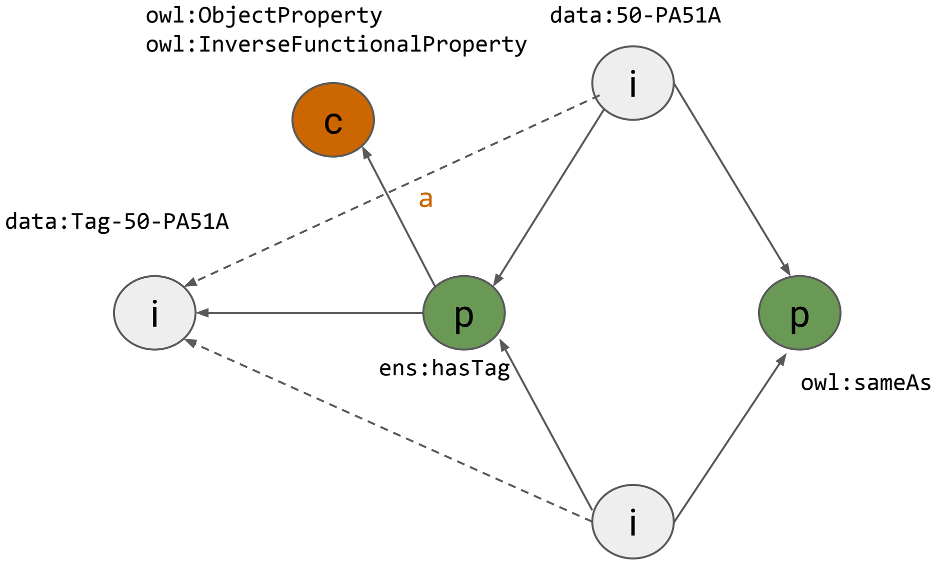
Inverse functional property

Off with reasoning

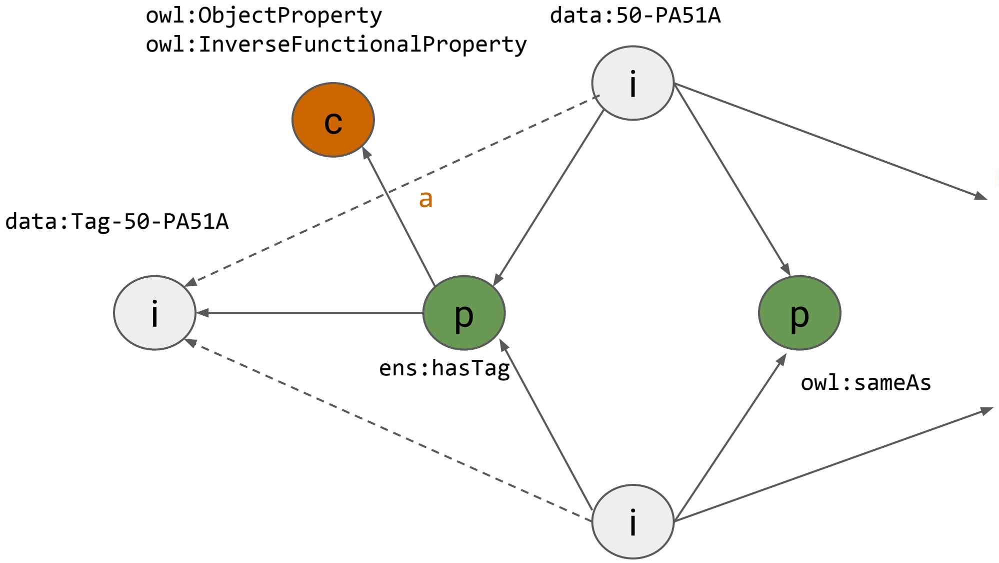
Inverse functional property



Inverse functional property



Inverse functional property



```
data:50-PA51B  rdf:type          ens:Pump_PA , ens:Equipment;
        ens:diffPressure_barg  4.2;
        ens:dischargesTo       data:50_PA51B_DischargeHeader;
        ens:hasDiscipline      data:DiscE , data:DiscS , data:DiscJ;
        ens:hasManufacturer    "Webstep";
        ens:hasTag             data:Tag-50-PA51B;
        ens:minFlow_m3h        32.8;
        ens:partOfSystem       data:System50;
        ens:tagNumber          "50-PA51B" .
```

Inverse functional property

```
# QUERY 1: TRYING TO DUPLICATE INVERSE FUNCTIONAL REASONING
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

CONSTRUCT {
       ?canon ?p ?o .
       ?s ?p ?canon .
}
WHERE {
  VALUES ?seed { data:50_PA51B }
  ?seed ens:hasTag ?tag .
  {
       SELECT ?tag (MIN(STR(?m)) AS ?canonStr)
       WHERE { ?m ens:hasTag ?tag }
       GROUP BY ?tag
  }
  BIND( IRI(?canonStr) AS ?canon )

  ?oneOrOther ens:hasTag ?tag .

  { ?oneOrOther ?p ?o }
  UNION
  { ?s ?p ?oneOrOther }
}
```

Inverse functional property

On with reasoning

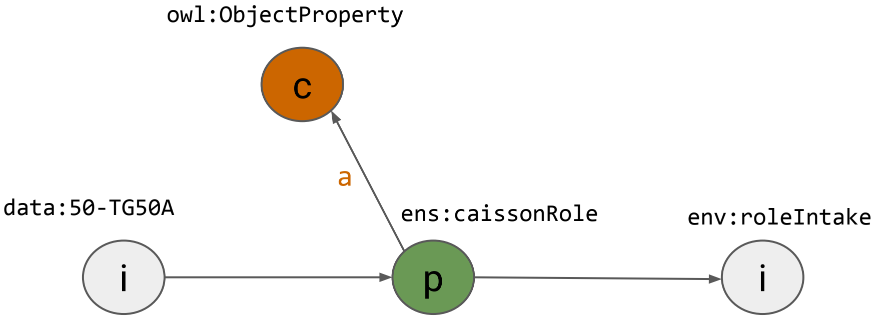
Inverse functional property

```
# QUERY 2: INVERSE FUNCTIONAL PROPERTY REASONING
PREFIX data:  <http://www.webstep.no/workshop/data/>
PREFIX ens:   <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
    ?s ?p ?o .
}
WHERE
{
  ?s ?p ?o .

  FILTER ( ? = data:50_PA51B )
  #FILTER ( ? = data:50-PA51B )
}
```

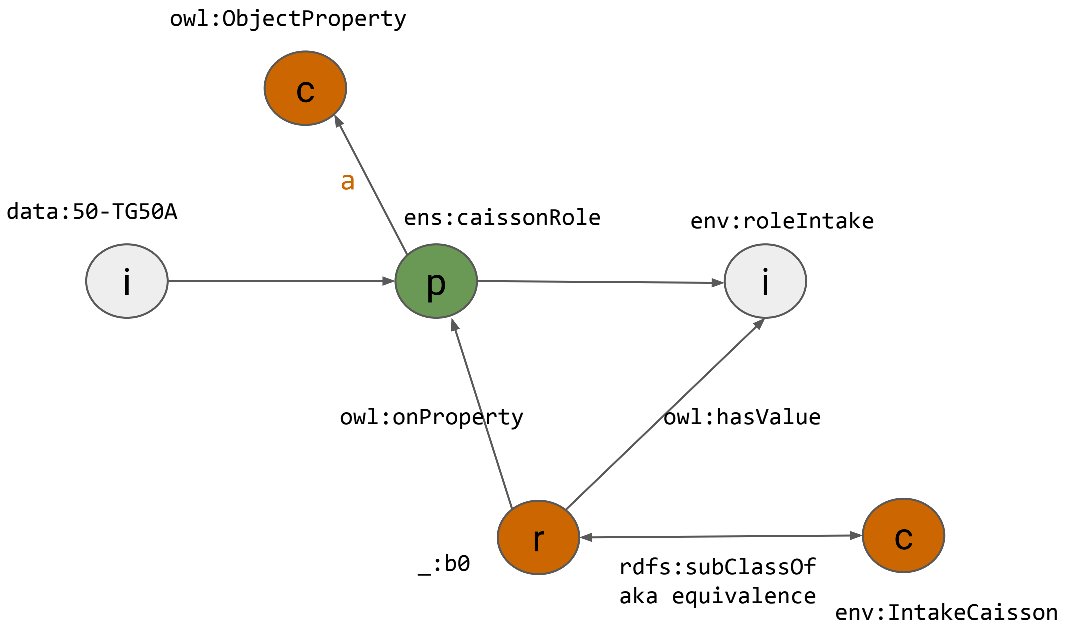
Restrictions

Off with reasoning

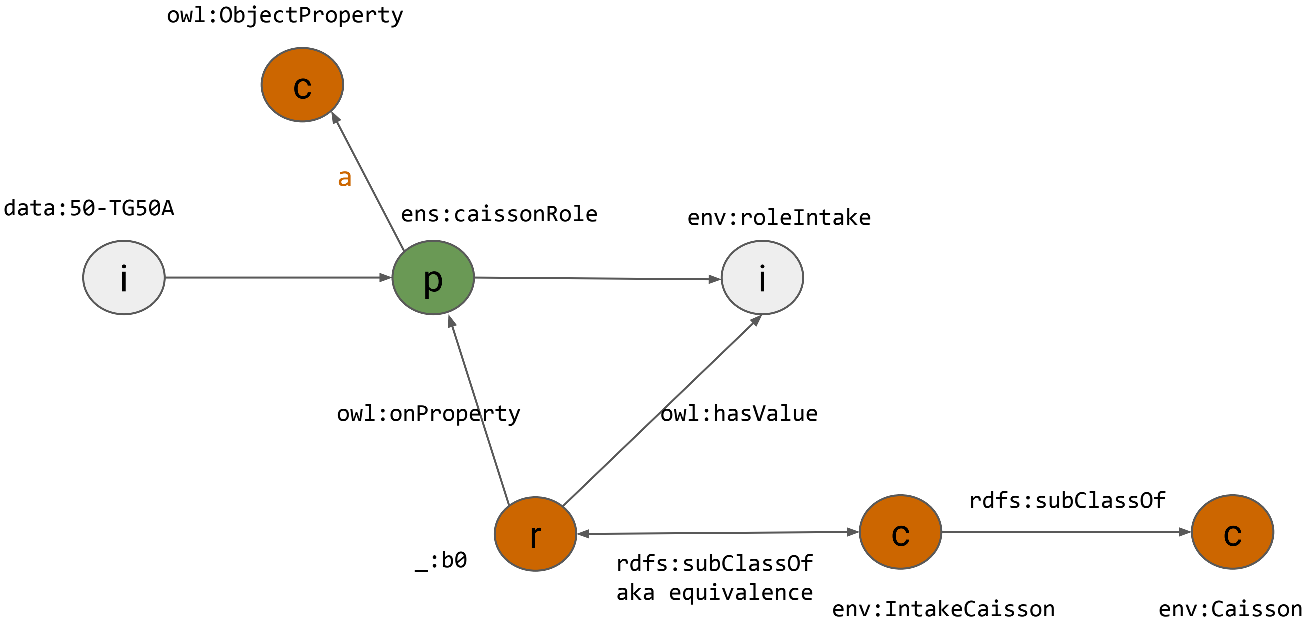
Restrictions



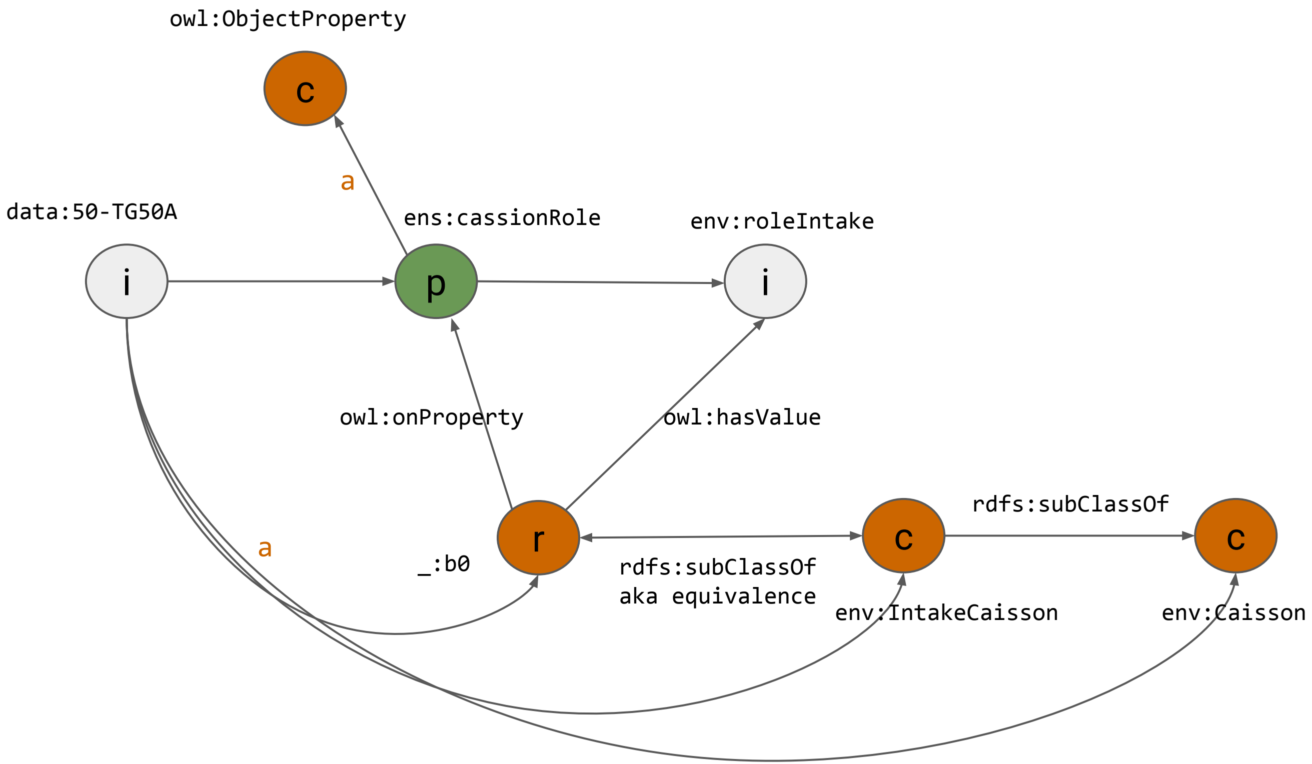
Restrictions



Restrictions



Restrictions



Restrictions

```
# QUERY 1: REASONING PATTERN FOR RESTRICTION WITH HAS VALUE
PREFIX owl:  <http://www.w3.org/2002/07/owl#>
PREFIX data: <http://www.webstep.no/workshop/data/>
PREFIX ens:  <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
     ?x a ?R, ?C .
}
WHERE
{
  ?C owl:equivalentClass ?R .
  ?R a owl:Restriction ;
       owl:onProperty ?p ;
       owl:hasValue ?v .

   ?x ?p ?v .
}
```

Restrictions

On with reasoning

Restrictions

```
# QUERY 2: VALIDATING RESTRICTION REASONING
PREFIX owl:  <http://www.w3.org/2002/07/owl#>
PREFIX data: <http://www.webstep.no/workshop/data/>
PREFIX ens:  <http://www.webstep.no/workshop/ens/>

CONSTRUCT
{
     ?s ?p ?o.
}
WHERE
{
  ?typeOfCaisson rdfs:subClassOf ens:Caisson .
  ?s a ?typeOfCaisson ;
       ?p ?o .
}
```