University of
Central Lancashire
UCLan

CO3409
Distributed Enterprise Systems

Distributed knowledge
The semantic web
JSON for linked data

Where opportunity creates success

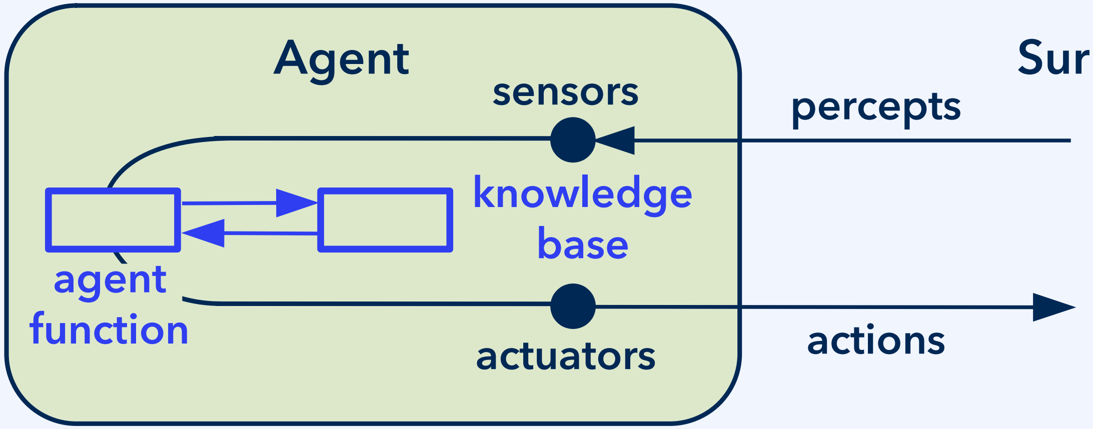
University of Central Lancashire
UCLan

Distributed knowledge

CO3409

24th February 2022

Knowledge-based agents



The agent function interacts with the **knowledge base (KB)** in three ways:

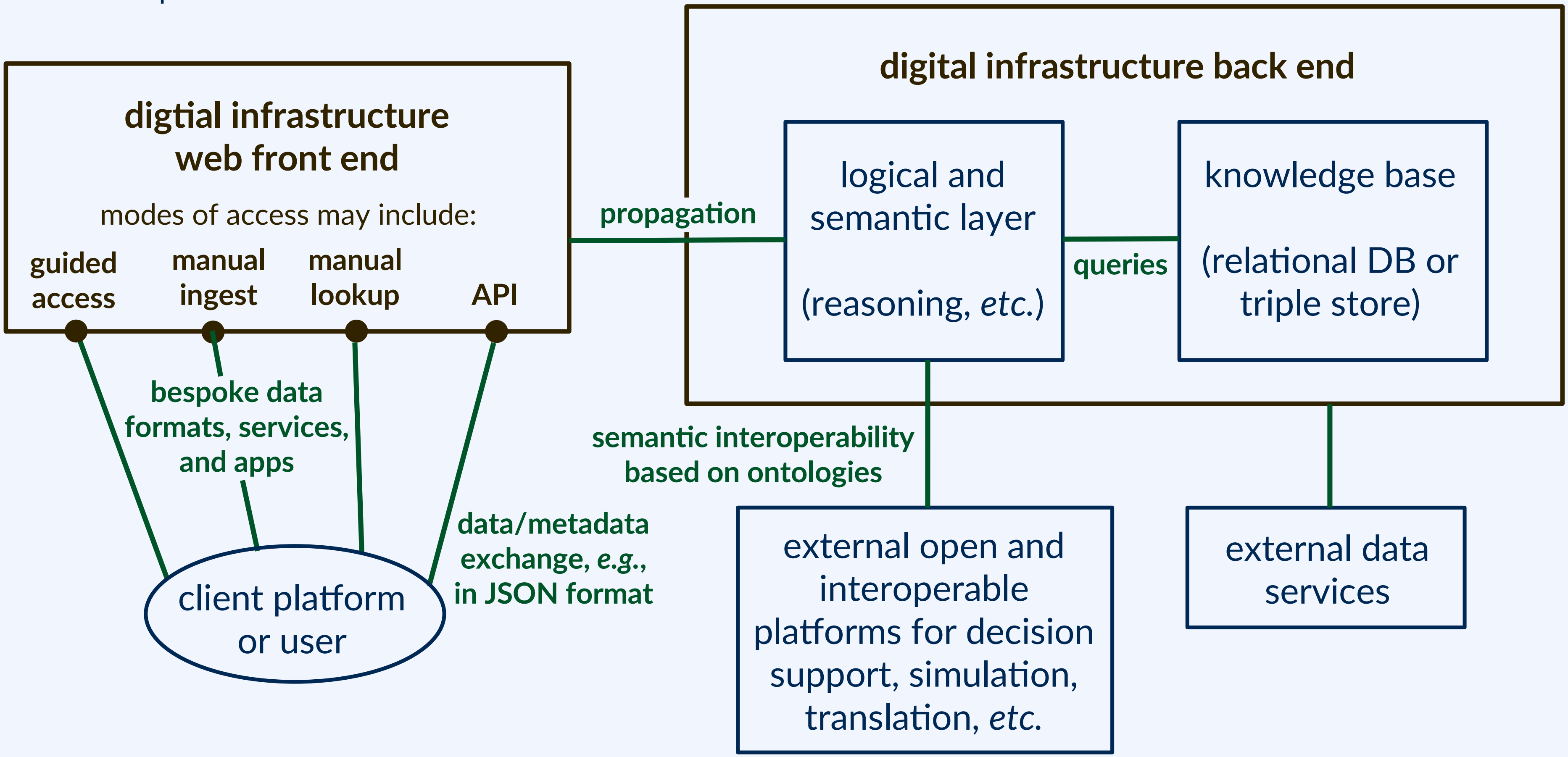
1) First, the agent function **ingests** relevant percepts into the KB.
2) Second, it **queries** the KB for information needed in decision making.
3) Third, it **ingests** information about its own actions into the KB.”

Interactions with the knowledge base take two forms:

– **Data ingest** (“tell”) to extend or update the information about the world.
– **Data retrieval** based on **querying** (“ask”).

Knowledge-based enterprise systems

Example architecture:[1]

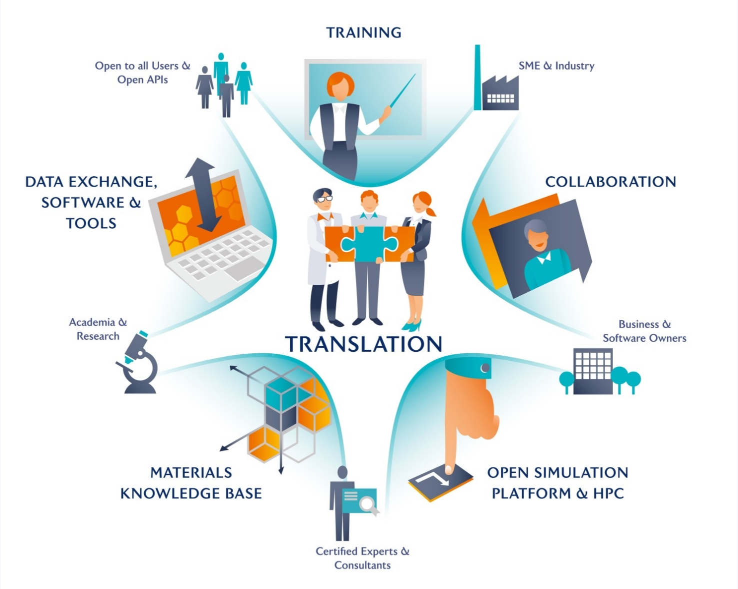


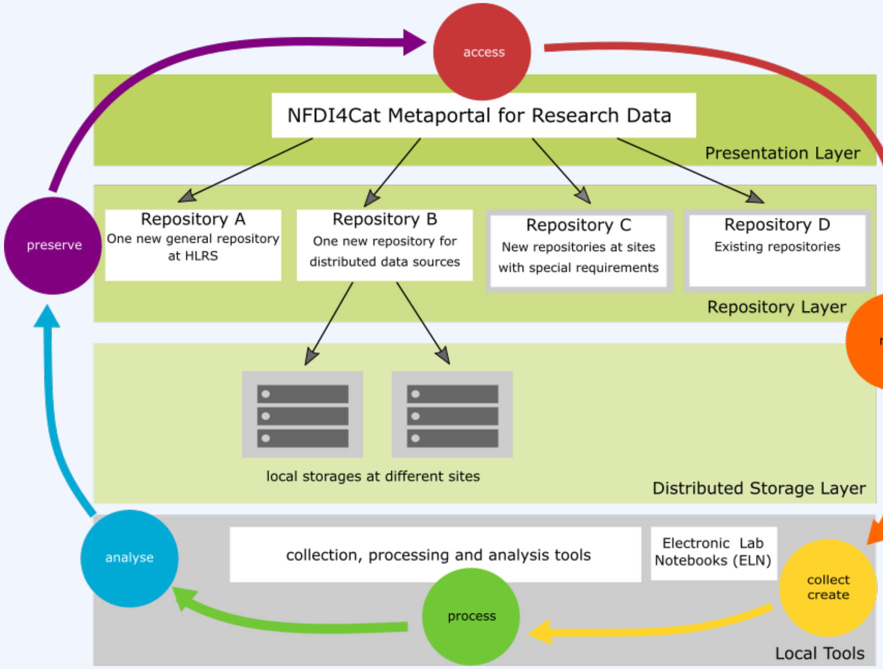
digital infrastructure back end
digtial infrastructure web front end
modes of access may include:
guided access
manual ingest
manual lookup
API
propagation
logical and semantic layer
(reasoning, *etc.*)
queries
knowledge base
(relational DB or triple store)
bespoke data formats, services, and apps
semantic interoperability based on ontologies
data/metadata exchange, *e.g.*, in JSON format
client platform or user
external open and interoperable platforms for decision support, simulation, translation, *etc.*
external data services

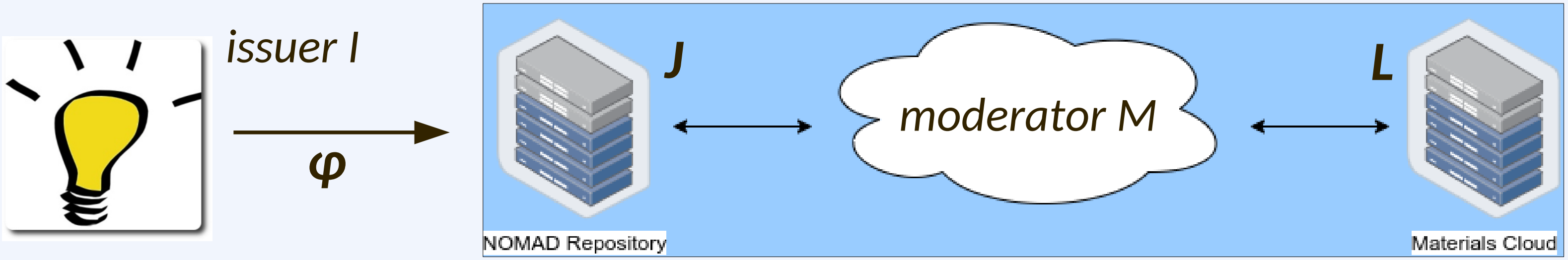
[1] Heinen *et al.*, doi:10.1007/978-3-030-80602-6_36, in *HPC in Science & Engineering '20*, **2021**.

Knowledge exchange in distributed architectures

Example architectures:[1]


TRAINING
Open to all Users & Open APIs
SME & Industry
DATA EXCHANGE, SOFTWARE & TOOLS
COLLABORATION
Academia & Research
TRANSLATION
Business & Software Owners
MATERIALS KNOWLEDGE BASE
OPEN SIMULATION PLATFORM & HPC
Certified Experts & Consultants


access
NFDI4Cat Metaportal for Research Data
Presentation Layer
preserve
Repository A
One new general repository at HLRS
Repository B
One new repository for distributed data sources
Repository C
New repositories at sites with special requirements
Repository D
Existing repositories
Repository Layer
reuse
local storages at different sites
Distributed Storage Layer
analyse
collection, processing and analysis tools
Electronic Lab Notebooks (ELN)
collect create
process
Local Tools



issuer I

φ

J

moderator M

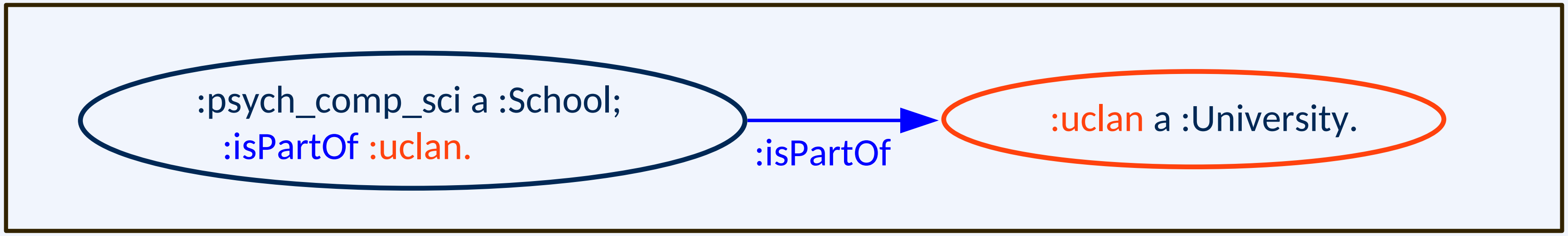
L

[1] Sources: DiplIng (BMBF no. 16FDM008), NFDI4Cat (https://nfdi4cat.org/), VIMMP (https://vimmp.eu/).

Knowledge graphs

Modern knowledge bases represent knowledge about the state of affairs as **knowledge graphs**. These graphs are understood as part of one **semantic web**.

They visualize simple propositions: **Triples** of a **subject**, a **predicate**, an **object**.



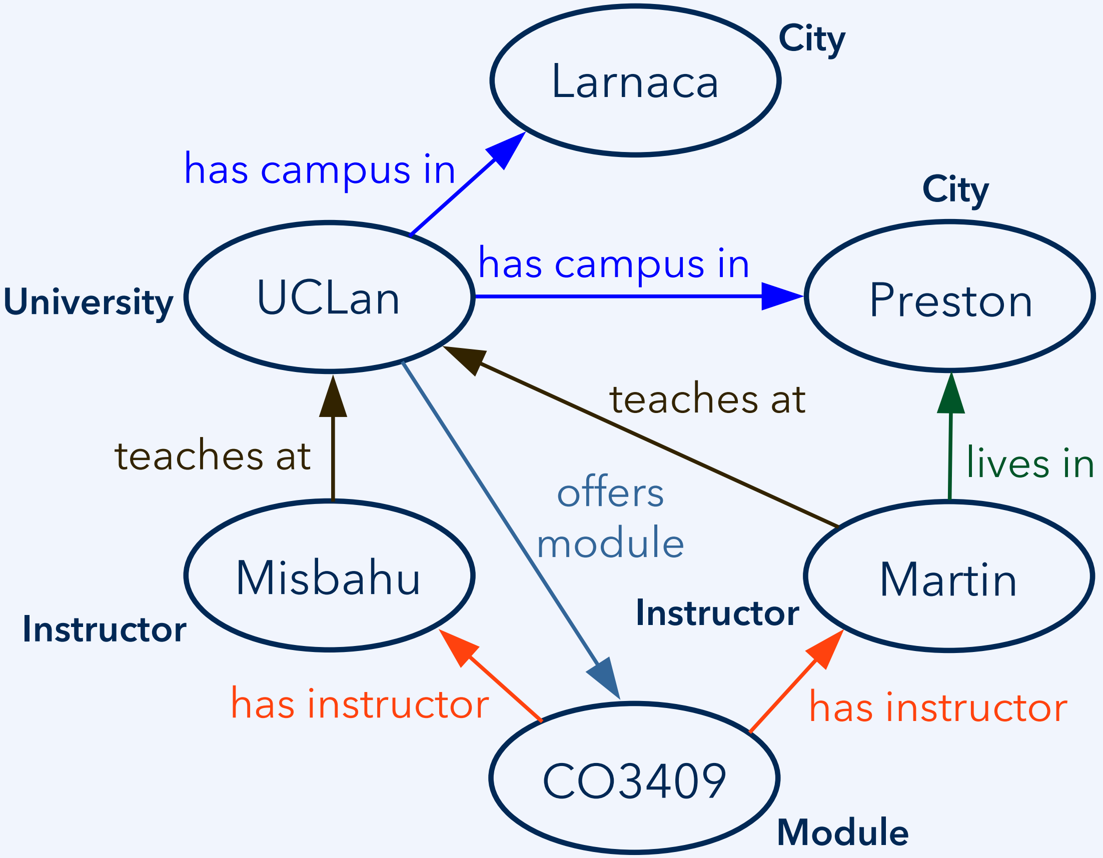
```
:psych_comp_sci  :isPartOf  :uclan.
```

RDF triple, consisting of subject, predicate, and object

Semantics (*i.e.*, meaning) of the graph above: "The School of Psychology and Computer Science is a school. It is part of UCLan which is a university."

Knowledge graphs (university example)

Modern knowledge bases represent knowledge about the state of affairs as **knowledge graphs**. These graphs are understood as part of one **semantic web**.



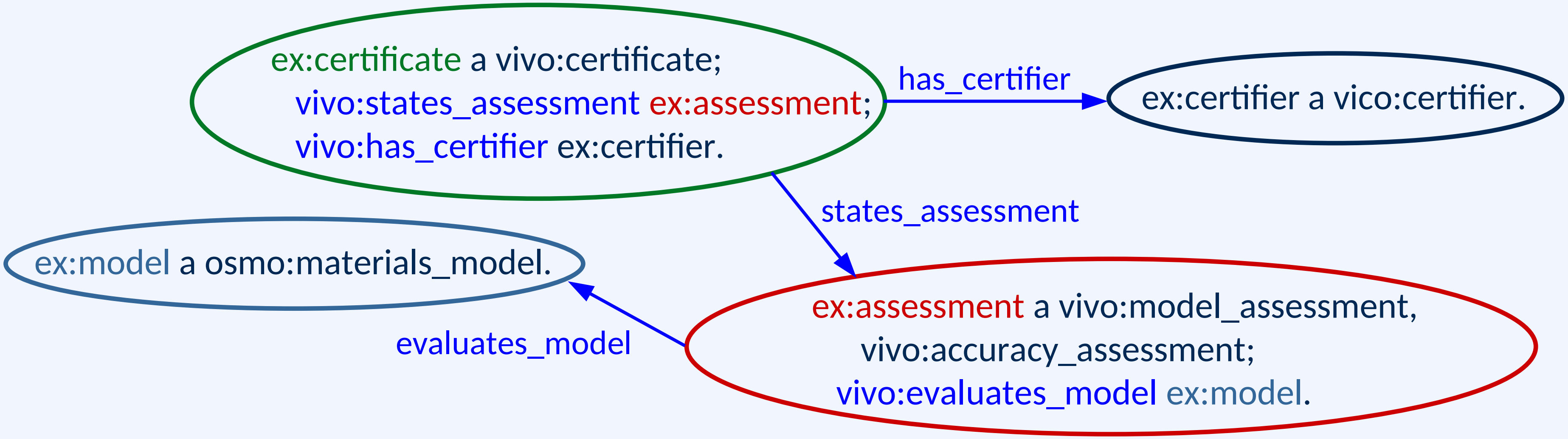
Knowledge graphs contain **individuals** (objects) as nodes.

They contain **relations** (binary predicates) as edges.

They may also visually represent the instantiation of **concepts** (classes).

Knowledge graphs (marketplace example)

„A certifier has issued a certificate with a model accuracy assessment that evaluates a materials model.“



ex:certificate a vivo:certificate;
vivo:states_assessment ex:assessment;
vivo:has_certifier ex:certifier.

ex:certifier a vico:certifier.

ex:model a osmo:materials_model.

ex:assessment a vivo:model_assessment,
vivo:accuracy_assessment;
vivo:evaluates_model ex:model.

A certifier (the individual ex:certifier, instantiating the concept vico:certifier) **has issued a certificate** (the individual ex:certificate, instantiating the concept vivo:certificate) **with a model accuracy assessment** (the individual ex:assessment, instantiating the concepts vivo:accuracy_assessment and vivo:model_assessment) **that evaluates a materials model** (the individual ex:model, instantiating the concept osmo:materials_model)**.**

Why “semantic” web?

Three branches of the theory of formal languages:

- **Syntax** (theory of the **structure** of language)
- **Semantics** (theory of the **meaning** of language)
- **Pragmatics** (theory of the **use** of language[1])

Generally speaking, semantics refers to “meaning,” as opposed to syntax, which refers to “proper grammar and notation.”

Under many typical circumstances (particularly in computing), a code, formula, statement, *etc.*, can only have a semantic content if it has correct syntax. Human language pragmatics permits people to also make sense of utterances that are not grammatically correct.

The same semantics can be encoded in many ways, using many languages. Specifying semantics directly, in whatever format, increases portability.

University of Central Lancashire
UCLan

The semantic web

CO3409

24th February 2022

The semantic web

Semantic technology can facilitate the integration of data and software into a coherent framework, permitting multiple components to become interoperable.

On the semantic web, data and metadata are provided as RDF triples:

Triples: Individual Relation Individual. (Subject Predicate Object.)

Example: theFox eats theChicken.

(Other kind of triples: Individual “a” Concept. Example: theFox a Fox.)

RDF is the Resource Description Framework, which specifies the semantic web. In this context, a **resource** is any of the following:

an **individual** (*i.e.*, object); a **concept** (*i.e.*, class); a property/**relation**.

Resources are referenced by using **Internationalized Resource Identifiers (IRIs)**.

Terse triple language (TTL)

Terse triple language, also known as **turtle format**,[1] is a compact notation for triples that is easy to write in a text editor.

TTL format[1]

[1] https://www.w3.org/TR/turtle/

RDF: Resource Description Framework

RDF triples

```
ex:certificate a vivo:certificate;
  vivo:states_assessment ex:assessment;
  vivo:has_certifier ex:certifier.
```

"ex:certificate is a certificate.
It states an assessment, namely, ex:assessment.
It has a certifier, namely, ex:certifier."

```
subject    a                 class_of_subject;
           has_property      first_object, second_object;
           other_property    another_object.
```

Internationalized resource identifiers (IRIs)

In the Resource Description Framework (RDF), all **individuals** (objects), **relations** (properties), and **concepts** (classes) are regarded as resources. Resource *identifiers need not be resolvable*; if they are, they become *locators*.

IRIs as resource identifiers

prefix:suffix

The prefix acts like a namespace. In TTL format, it may be empty, as in ":suffix_only".

RDF triples in TTL format

```
ex:certificate a vivo:certificate;
  vivo:states_assessment ex:assessment;
  vivo:has_certifier ex:certifier.
```

"ex:certificate is a certificate.
It states an assessment, namely, ex:assessment.
It has a certifier, namely, ex:certifier."

These short prefixes act as abbreviations for the full first part of the IRI:

```
@prefix vivo: <https://purl.vimmp.eu/semantics/vivo/vivo.ttl#>.
```

Principles of the semantic web

Triples: Individual Relation Individual. (Subject Predicate Object.)

(1) Frank is_father_of Robert.

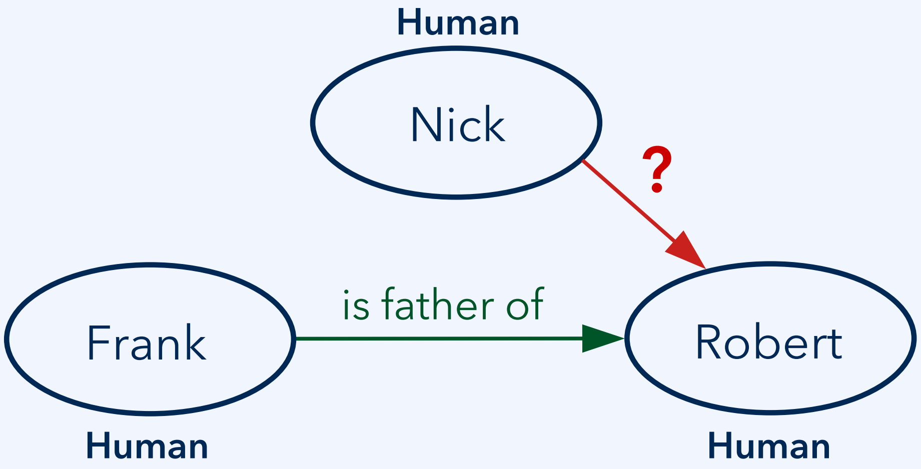
Q: “Is Nick the father of Robert?”

Human is a concept.

Frank, Robert, etc., are Humans.

Cardinality restriction:

Every Human has exactly 1 father.



Principles of the semantic web

Triples: Individual Relation Individual. (Subject Predicate Object.)

(1) Frank is_father_of Robert.

Q: "Is Nick the father of Robert?"
A: "We don't know!"

Human is a concept.
Frank, Robert, etc., are Humans.

Cardinality restriction:
Every Human has exactly 1 father.

Principle: Non-unique name assumption

Unless stated otherwise, multiple identifiers may refer to the same resource.
This is useful for data integration from different sources:

first-namespace:name-here is_same_as second-namespace:name-there.

Principles of the semantic web

Triples: Individual Relation Individual. (Subject Predicate Object.)

(1) Frank is_father_of Robert.

(2) Frank is_different_from Nick.

Q: "Is Nick the father of Robert?"

A: "No, he is not."

(3) Frank is_father_of Anna.

Q: "How many children does Frank have?"

Human is a concept.

Frank, Robert, etc., are Humans.

Cardinality restriction:

Every Human has exactly 1 father.

Anna is_different_from Robert.

"How many different X are there such that Frank is_father_of X?"

Principle: Non-unique name assumption

Unless stated otherwise, multiple identifiers may refer to the same resource. This is useful for data integration from different sources:

first-namespace:name-here is_same_as second-namespace:name-there.

Principles of the semantic web

Triples: Individual Relation Individual. (Subject Predicate Object.)

(1) Frank is_father_of Robert.

(2) Frank is_different_from Nick.

Q: "Is Nick the father of Robert?"
A: "No, he is not."

Human is a concept.
Frank, Robert, etc., are Humans.

Cardinality restriction:
Every Human has exactly 1 father.

(3) Frank is_father_of Anna.

Q: "How many children does Frank have?"
A: "At least two."

Anna is_different_from Robert.

"How many different X are there such that Frank is_father_of X?"

Principle: Open world assumption

Since relevant information may distributed over the semantic web, rather than from the presently considered source only, **available knowledge is assumed to be incomplete**. (Contrast this with a closed, monolithic database architechture.)

JSON for linked data

CO3409

24th February 2022

JSON in digital infrastructures

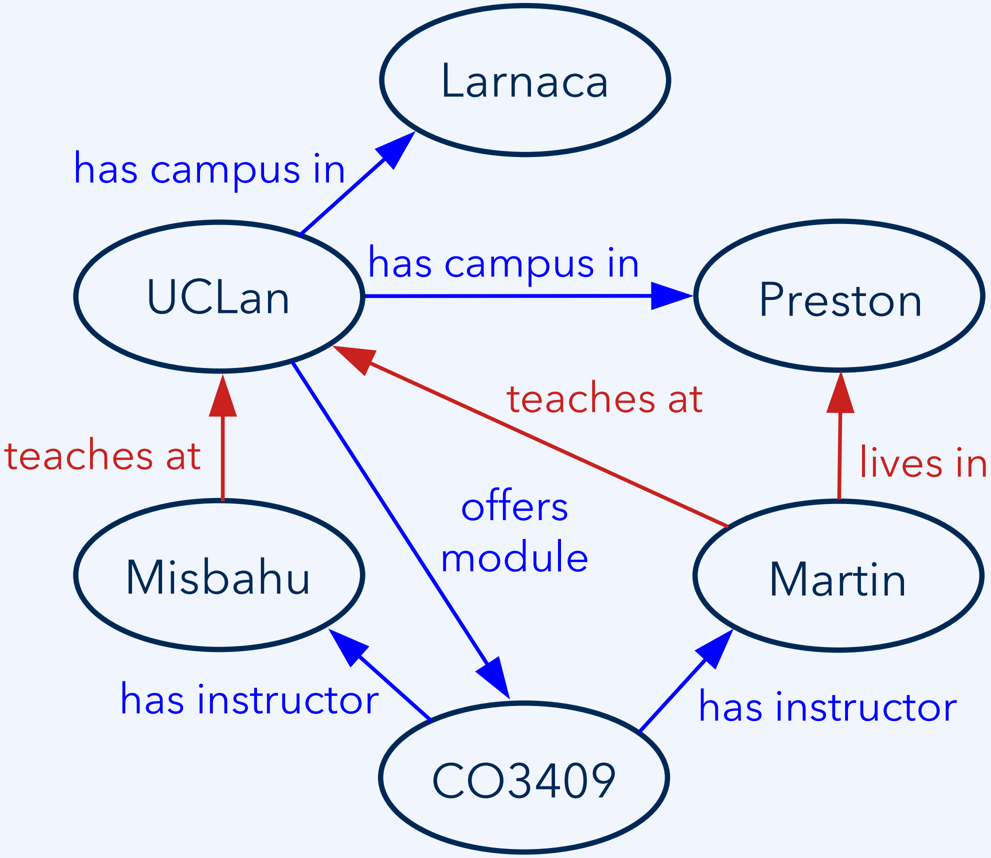
JSON is often used for data ingest & extraction into/from DBs via RESTful APIs:

- JSON is a hierarchical format, in which one element can *contain* other elements; in this sense it is equivalent to XML, but with less overhead.
- JSON can be used as a type in *relational databases* including MySQL,[1] *i.e.*, JSON formatted data can be ingested without transformation.
- It is also used in *non-relational DBs*; *e.g.*, MongoDB is based on JSON.[2]
- The hierarchical structure – *transitivity* of the containment relation – limits the way in which objects can be connected to each other (*trees* only).
- Data in a JSON file are self-contained, there is no standard way to include *external resources*; except via "JSON linked data" (JSON-LD).

[1] https://dev.mysql.com/doc/refman/8.0/en/json.html

[2] https://docs.mongodb.com/guides/server/introduction/

JSON representation of objects and properties

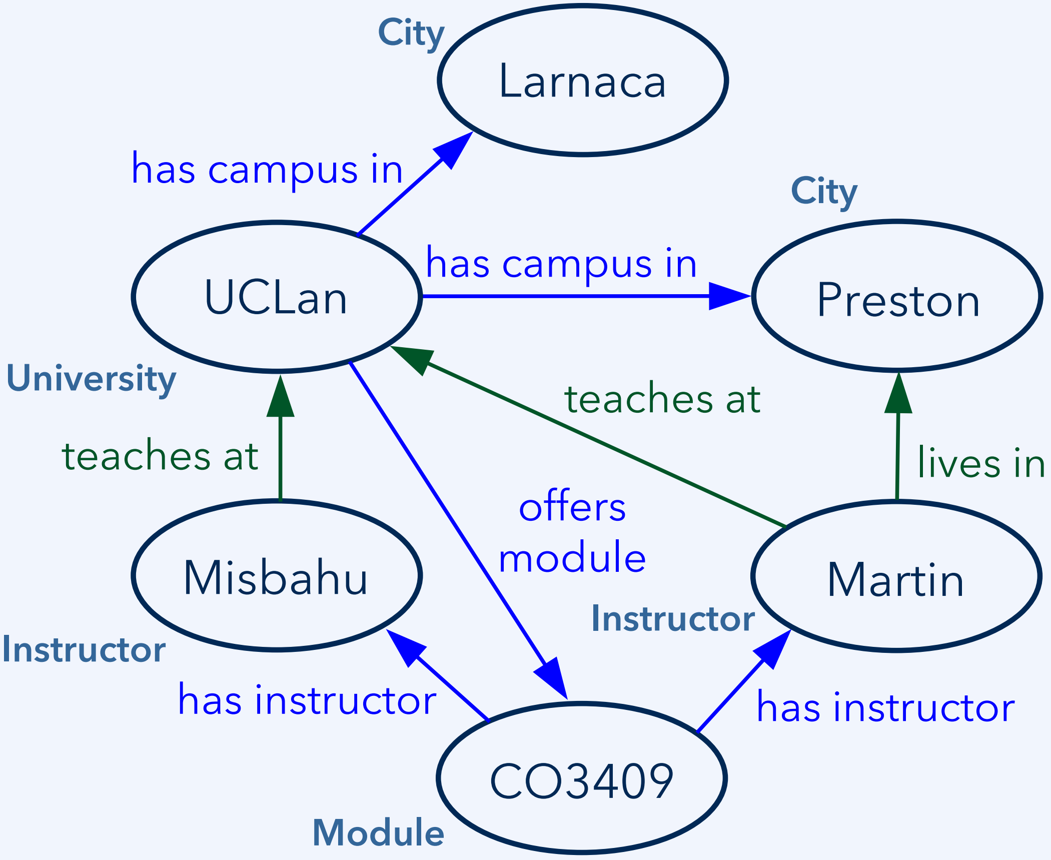


```
{
  "identifier": "UCLan",
  "hasCampusIn": [
    {
      "identifier": "Larnaca"
    }, {
      "identifier": "Preston"
    }
  ],
  "offersModule": {
    "identifier": "CO3409",
    "hasInstructor": [
      {
        "identifier": "Misbahu"
      }, {
        "identifier": "Martin"
      }
    ]
  }
}
```

tree structure required

no connection to semantic web

JSON for linked data: JSON-LD

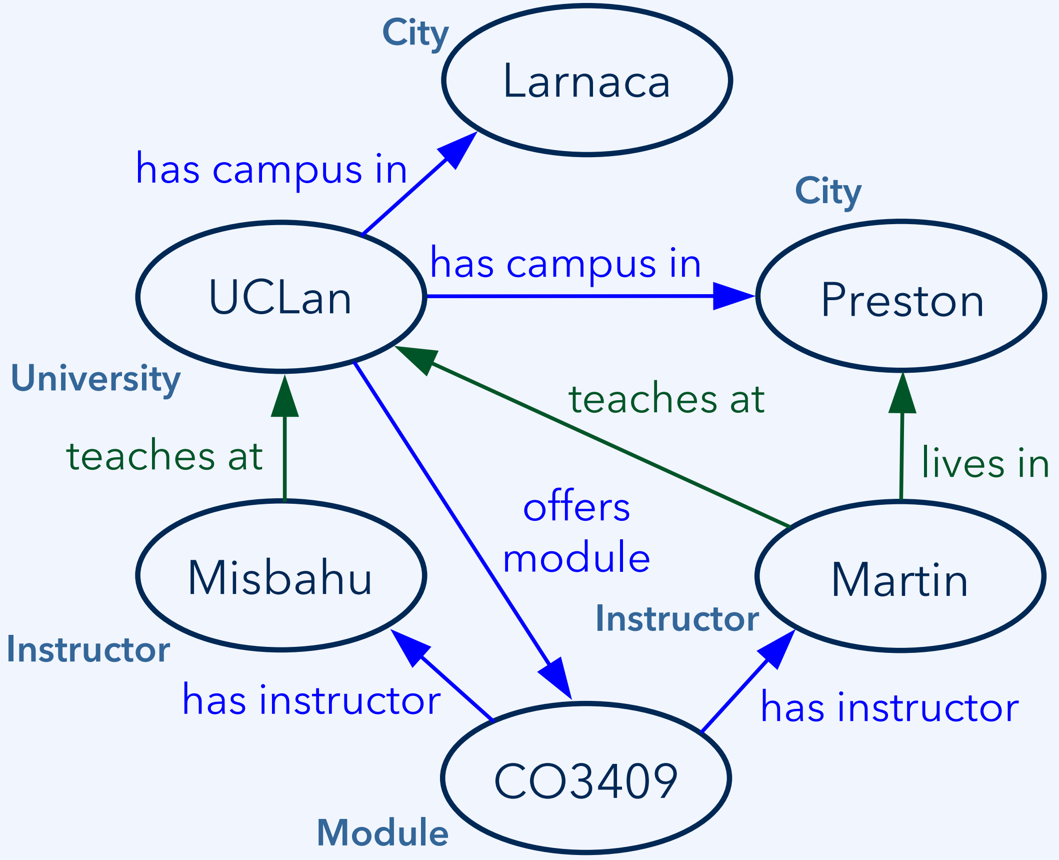


IRIs to reuse elements

Type (concept) IRIs for instantiation

```
{
  "@id": "scenario:uclan",
  "@type": "uni:University",
  "uni:hasCampusIn": [
    {
      "@id": "scenario:larnaca",
      "@type": "uni:City"
    }, {
      "@id": "scenario:preston",
      "@type": "uni:City"
    }
  ],
  "uni:offersModule": {
    "@id": "scenario:co3409",
    "@type": "uni:Module",
    "uni:hasInstructor": [
      ...
    ]
  }
}
```

JSON for linked data: JSON-LD



IRIs to reuse elements

IRIs to refer to external resources

hierarchical tree structure is retained from basic JSON

"@id" property to specify the object's IRI

"@type" property to specify the IRI of an instantiated concept

using object IRIs permits creating links beyond tree structure

JSON for linked data: JSON-LD

IRI prefixes/namespaces are declared through “@context”.

```
  "@context": {
     "uni": "http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#",
     "scenario": "http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#"
   }
```

```
"@id": "scenario:uclan",
"@type": "uni:University",
"uni:hasCampusIn": [
  ...
],
"uni:offersModule": {
  "@id": "scenario:co3409",
  "@type": "uni:Module",
  "uni:hasInstructor": [
    ...
  ]
}
```

“@id” and “@type” are used for IRIs of individuals and instantiated concepts.

IRIs are abbreviated using the declarations from “@context”.

```
"@id": "scenario:martin",
"@type": "uni:Instructor",
"uni:teachesAt": {
  "@id": "scenario:uclan"
},
"uni:livesIn": {
  "@id": "scenario:preston"
}
```

Use IRIs for referring to an object without containing it.

JSON-LD for distributed enterprise systems

Observations about JSON-LD:

- It is the most widespread format for communicating *semantically characterized content* via RESTful services on digital infrastructures.
- Normally, only the knowledge graph (content, *"assertions"*) is shared as JSON-LD; other formats are used for the *terminology* itself, if required.
- JSON-LD retains the hierarchical structure of JSON, but objects can be referenced multiple times through their IRI; a root node is still required.
- Like JSON, it is designed for easy parsing. While it is human readable, it is not optimized for that purpose. A feature that it shares with many such languages and formats is its reliance on nested parentheses.

Compare the JSON-LD file to the TTL file

```
@prefix uni: <http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#>.
@prefix scenario: <http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#>.

scenario:uclan a uni:University;                               # UCLan is a university;
  uni:hasCampusIn scenario:larnaca, scenario:preston;          # it has campuses in Larnaca and Preston;
  uni:offersModule scenario:co3409.                            # it offers the module CO3409.

scenario:larnaca a uni:City.     # Larnaca is a city.
scenario:preston a uni:City.     # Preston is a city.

scenario:co3409 a uni:Module;                                  # CO3409 is a module;
  uni:hasInstructor scenario:martin, scenario:misbahu.         # it has Martin and Misbahu as instructors.

scenario:martin a uni:Instructor;     # Martin is an instructor;
  uni:teachesAt scenario:uclan;       # he teaches at UCLan;
  uni:livesIn scenario:preston.       # he lives in Preston.

scenario:misbahu a uni:Instructor;    # Misbahu is an instructor;
  uni:teachesAt scenario:uclan.       # he teaches at UCLan.
```

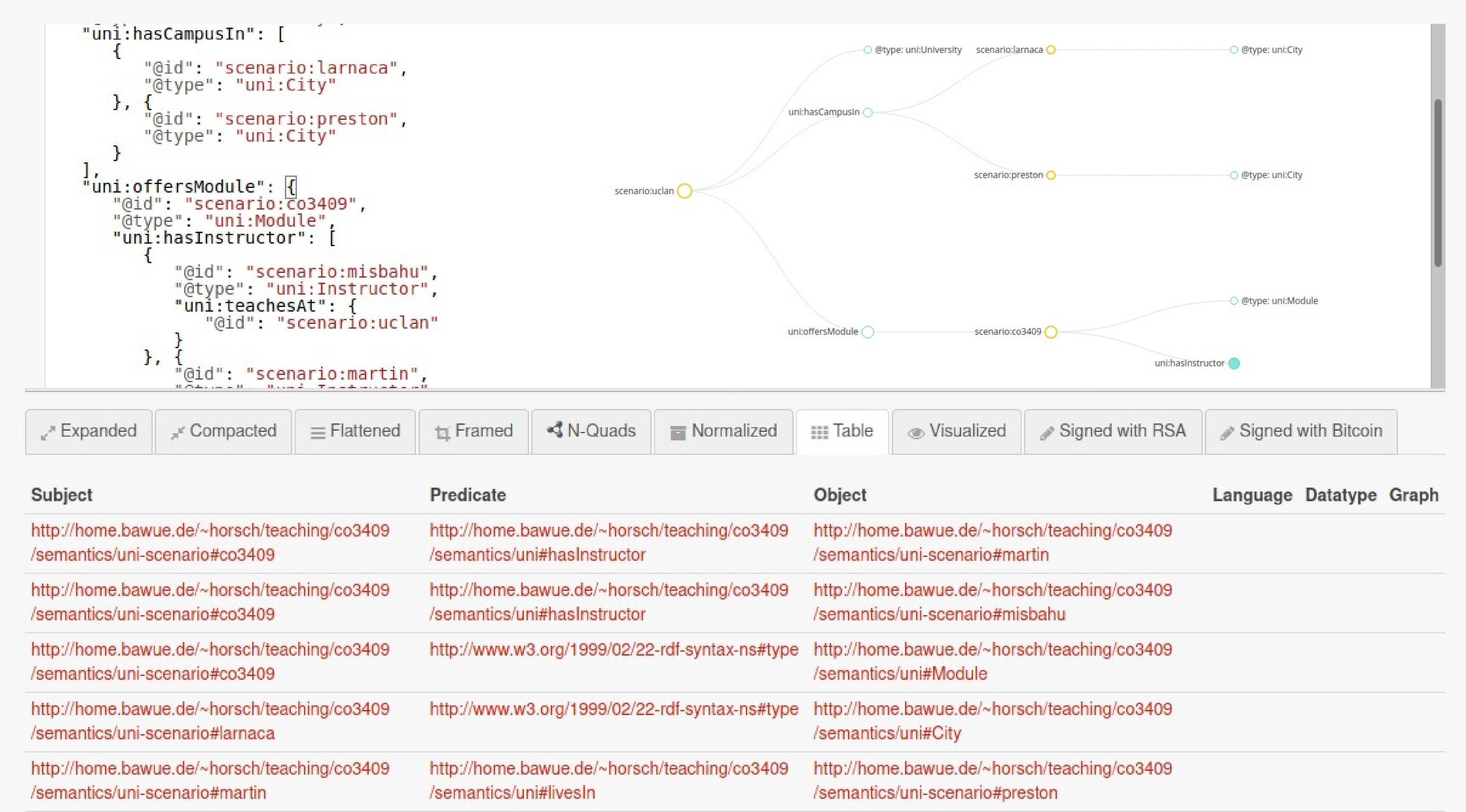
Discussion

CO3409

24th February 2022

Syntax and sanity check on JSON-LD files

JSON-LD Playground (https://json-ld.org/playground/)

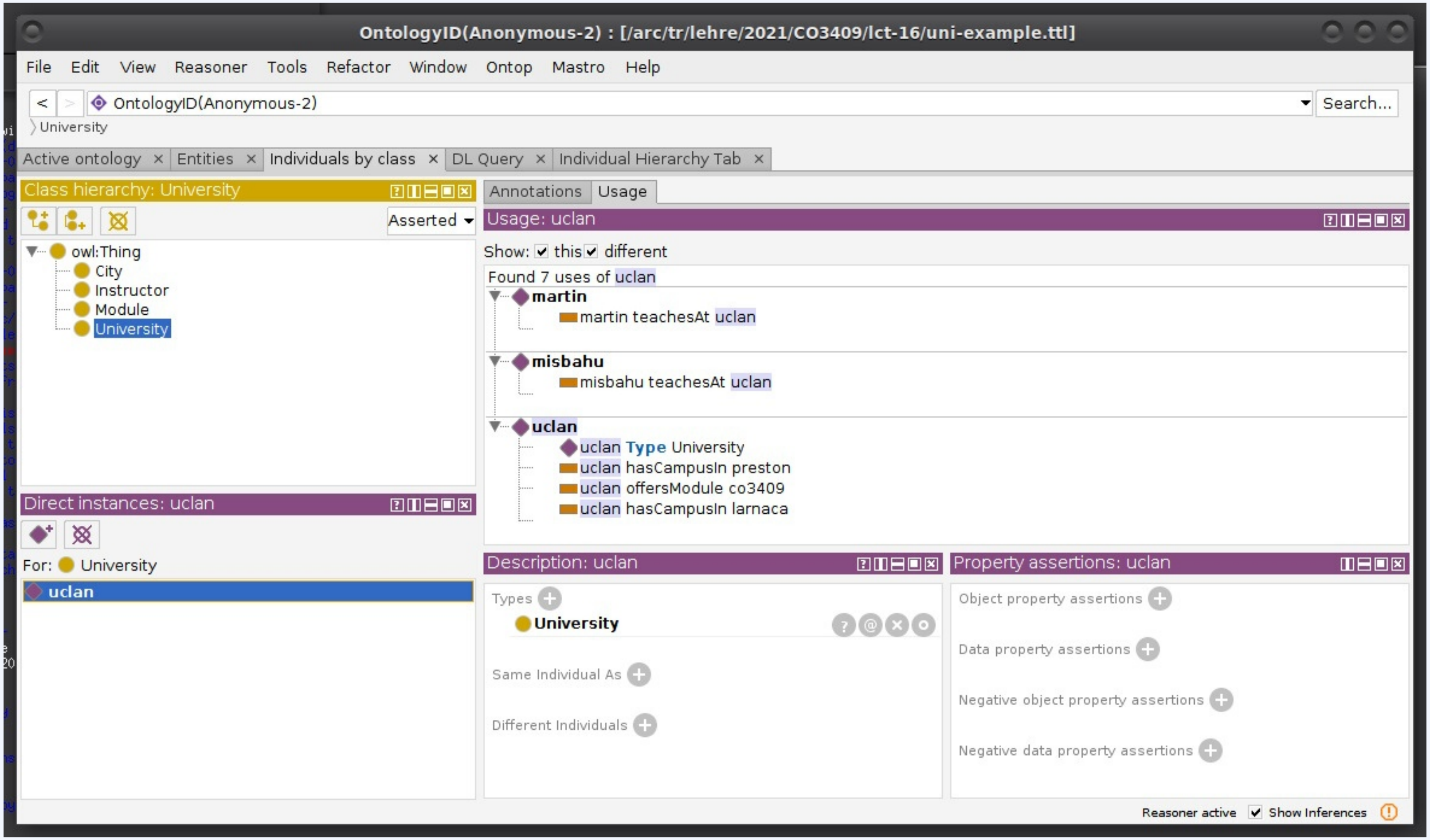


```
  "uni:hasCampusIn": [
      {
          "@id": "scenario:larnaca",
          "@type": "uni:City"
      }, {
          "@id": "scenario:preston",
          "@type": "uni:City"
      }
  ],
  "uni:offersModule": {
      "@id": "scenario:co3409",
      "@type": "uni:Module",
      "uni:hasInstructor": [
          {
              "@id": "scenario:misbahu",
              "@type": "uni:Instructor",
              "uni:teachesAt": {
                  "@id": "scenario:uclan"
              }
          }, {
              "@id": "scenario:martin",
```

Subject	Predicate	Object	Language	Datatype	Graph
http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#co3409	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#hasInstructor	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#martin			
http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#co3409	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#hasInstructor	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#misbahu			
http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#co3409	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#Module			
http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#larnaca	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#City			
http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#martin	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni#livesIn	http://home.bawue.de/~horsch/teaching/co3409/semantics/uni-scenario#preston			

Protégé tool for working with TTL, RDFS, etc.

Protégé (https://protege.stanford.edu/)



OntologyID(Anonymous-2) : [/arc/tr/lehre/2021/CO3409/lct-16/uni-example.ttl]

File Edit View Reasoner Tools Refactor Window Ontop Mastro Help
OntologyID(Anonymous-2)
Search...
University
Active ontology × Entities × Individuals by class × DL Query × Individual Hierarchy Tab ×
Class hierarchy: University
Asserted
owl:Thing
City
Instructor
Module
University
Annotations Usage
Usage: uclan
Show: ☑ this ☑ different
Direct instances: uclan
For: University
uclan
Description: uclan
Types
University
Same Individual As
Different Individuals
Property assertions: uclan
Object property assertions
Data property assertions
Negative object property assertions
Negative data property assertions
Reasoner active ☑ Show Inferences

Found 7 uses of uclan

martin
- martin teachesAt uclan

misbahu
- misbahu teachesAt uclan

uclan
- uclan Type University
- uclan hasCampusIn preston
- uclan offersModule co3409
- uclan hasCampusIn larnaca

Lab worksheet challenge

The plan for the lab is to get started working with **JSON-LD** and **TTL**.

Install Protégé and try out both **Protégé** and the **JSON-LD Playground**.

Challenge, formulated by S. Borgo and O. Kutz.[1, 2]

Create a knowledge graph for the following information content:

> *"A flower is red in the summer.*
> *As time passes, the colour changes.*
> *In autumn the flower is brown."*

Formalize it as JSON-LD or TTL, or both if possible.

[1] http://stl.mie.utoronto.ca/upper/FOUST-templateV2CleanedCases.pdf, problem 3a.
[2] Also discussed in doi:10.5281/zenodo.4679522 under heading 3a.

University of
Central Lancashire
UCLan

CO3409
Distributed Enterprise Systems

Distributed knowledge
The semantic web
JSON for linked data

Where opportunity creates success