# Relazione Assignments Web Semantico

## Studio Libreria RDFLib e Definizione di una Ontologia RDF/OWL per Wireless Sensor Network e Multi-Agent System

•••

A.A. 2020/2021
filippo.paganelli3@studio.unibo.it
0000926989

Assignment 1: Studio Libreria Python RDFLib

- Features Libreria
- Struttura Libreria
- Criticità Libreria

Assignment 2: Definizione RDF

- Analisi Dominio WSN-MAS
- Analisi Ontologie Riusabili
- Binding Namespaces
- Enumerazione Termini
- Definizione Tassonomia
- Definizione Classi e Proprietà

Assignment 3: Definizione OWL

- Estensione RDF
- Definizione Istanze
- Serializzazione e Validazione
- Query SPARQL

# RDFLib

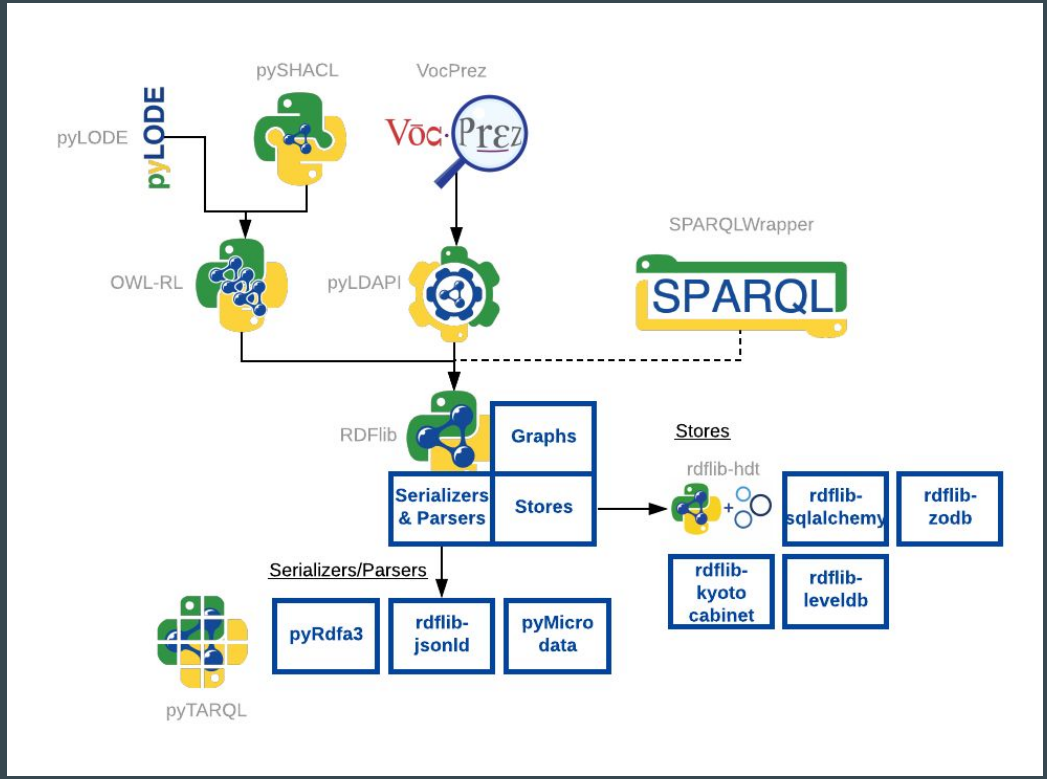
- libreria Python open-source per la manipolazione di RDF
- mantenuta su Github ( https://github.com/RDFLib/rdflib/ ) dal 2005
- disponibile tramite Python Package Index ( *pip install rdflib* )
- catalogata dal W3C sotto le categorie Triple Store[1] e Programming Environment[2] ( https://www.w3.org/2001/sw/wiki/RDFLib )

1 - Tools che possono essere usati come RDF Database.  Altri esempi di tools appartenenti a questa categoria sono RDFox, Oracle Spatial, Apache Jena e GraphDB.

2 - Tools che forniscono un ambiente funzionante per lo sviluppo di applicazioni riguardanti il Web Semantico nella forma di librerie o moduli. Altri esempi di tools appartenenti a questa categoria sono Apache Jena, Mobi, Sesame e Redland RDF.

La grande community che supporta il progetto mantiene anche altri progetti correlati al Web Semantico come librerie e plugin:

- **sparqlwrapper** - Wrapper SPARQL in Python per eseguire query remote.
- **pyLODE** - Tool Python per lo sviluppo di ontologie OWL basato su LODE.
- **rdflib-jsonld** - Plugin per RDFLib, implementazione di JSON-LD.



# Principali Features

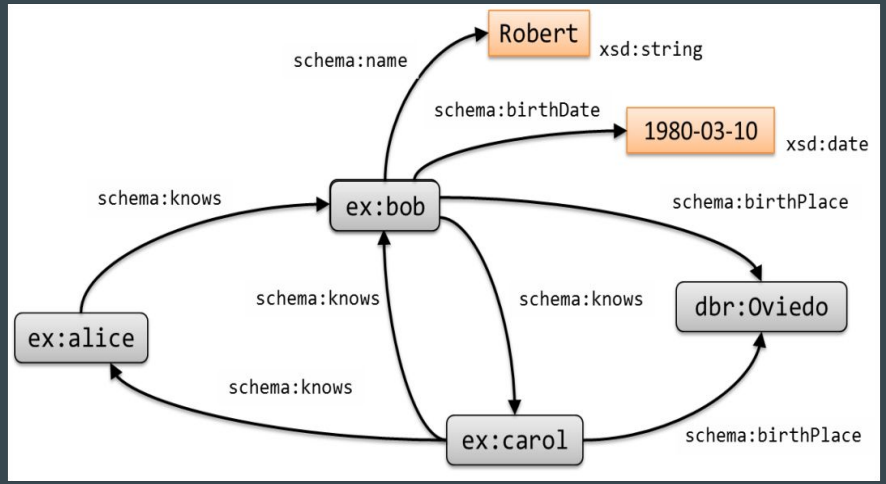
Le principali funzionalità fornite dalla libreria RDFLib sono:

- parsers e serializers per RDF/XML, N3, NTriples, N-Quads, Turtle, TriX, Trig e JSON-LD
- una interfaccia Graph che supporta diverse implementazioni di Store (single graph, conjunctive graph o dataset graph)
- implementazioni di store per memorizzazioni persistenti o in-memory basate su BerkeleyDB
- supporto a query e update SPARQL 1.1

# Struttura Libreria

- **RDF Terms** - Oggetti che possono comparire all'interno di una tripla RDF. Ognuna delle seguenti principali tipologie di RDF Terms è sottoclasse della classe Identifier:
  - **Blank Node** - Nodo in un RDF Graph che rappresenta una risorsa **anonima**, per il quale non è presente un URI o un literal. Può essere usato solamente come subject o object in una tripla RDF.
  - **URI Reference** - Stringa Unicode rappresentante un URI valido.
  - **Literal** - Rappresenta il valore degli attributi RDF e può avere un data-type oun tag language.
- **RDF Nodes** - I nodi del grafo sono un sottoinsieme dei termini che lo store memorizza e consistono nei subject e object delle triple del grafo.
- **RDF Namespaces** - Con namespace si intende una collezione di nomi, identificati da URI e utilizzati nei documenti
- come nomi per gli attributi e tipi per gli elementi.  Tra i namespace già disponibili all'interno della libreria è possibile trovare  RDFS, OWL, XSD, FOAF e DC ma è possibile anche definire un proprio namespace tramite l'apposito NamespaceManager ed effettuare il binding al documento RDF.

- **RDF Graph** - Grafo orientato a etichette ed identificato da un Identifier o BNode, dove gli archi rappresentano le relazioni tra le risorse, queste ultime rappresentate dai nodi. La libreria RDFLib permette la definizione delle seguenti tipologie di grafo:
  - **Graph** - Insieme di triple RDF, modellate con tuple a 3 terms nella forma <subject, predicate, object>.
  - **Quoted Graph** - Ad ogni grafo è associato un identificatore fornito dal parser N3 e permette la definizione di formule N3.
  - **Conjunctive Graph** - Aggregazione (unnamed) di tutti i named graphs contenuti in uno store.
  - **Dataset** - Estensione del grafo di tipo Conjunctive Graph che non può essere identificato da un BNode. Fornisce metodi differenti per manipolare il grafo.



- **RDF Store** - Chiamati anche Triple Store o Quad Store e sono ideati per memorizzare triple RDF e altre informazioni sul contesto e sul grafo. Dovrebbe fornire sia interfacce standard per la gestione delle connessioni a DBMS che interfacce standard per la manipolazione, gestione, creazione o rimozione (operazioni CRUD) di triple RDF.
  Uno store può essere:
  - Formula-aware - Store in grado di distinguere statements asserted da statements quoted. (Notation3)
  - Context-aware - Store in grado di memorizzare statements insieme al contesto, permettendo la partizione del modello RDF che rappresenta in named sub-graphs.

# Criticità Libreria

- **Problemi di upgrade della libreria** - Rilascio della nuova versione 6.0.0 programmata entro fine 2021 con supporto a versioni Python >= 3.6 .
- **Problemi di compatibilità tra le dipendenze della libreria** - Non per tutte le dipendenze della libreria è definita una specifica versione: questo comporta una risoluzione automatica delle dipendenze da parte di PyPI che potrebbe portare a conflitti/breaking changes.
- **Problemi di performance** - Scarse performance di alcuni algoritmi riguardanti principalmente l'esecuzione di query SPARQL e parsing.
- Conflitti con alcune specifiche dello standard W3C.
- >200 Issue e >25 Pull Request aperte[1]

1 - Dati al 13/09/2021 ( https://github.com/RDFLib/rdflib/issues , https://github.com/RDFLib/rdflib/pulls )

# Progettazione Ontologia

Il processo di sviluppo adottato per la progettazione della ontologia, segue a grandi linee quello indicato da Noy e McGuinness in ”Ontology Development 101:  A Guide to Create Your First Ontology”[1]:

- Analisi del dominio
- Analisi ontologie riusabili
- Enumerazione termini
- Definizione tassonomia
- Definizione schema (Classi e Proprietà)
- Definizione istanze
- Serializzazione e validazione

1 - Natalya F. Noy, Deborah L. McGuinness, Ontology Development 101:  A Guide to Creating Your First Ontology, Stanford University

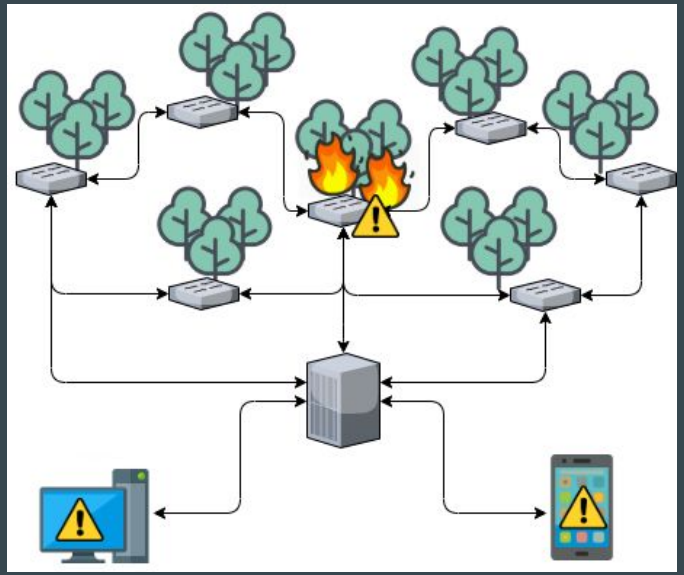
# Analisi del dominio - Enumerazione dei termini

Il dominio scelto è quello delle Wireless Sensor Networks e Multi-Agent System, applicate allo scenario di rilevazione degli incendi.

Il primo passo utile per la definizione di una ontologia è la scrittura di una lista di tutti i termini rilevanti del dominio che ci si aspetta di modellare con la ontologia.

- Wireless Sensor Network
  - Device
  - Sensor
  - Router
  - Node
  - Environment

- Multi-Agent System
  - Agent
  - Platform
  - JID
  - Contact



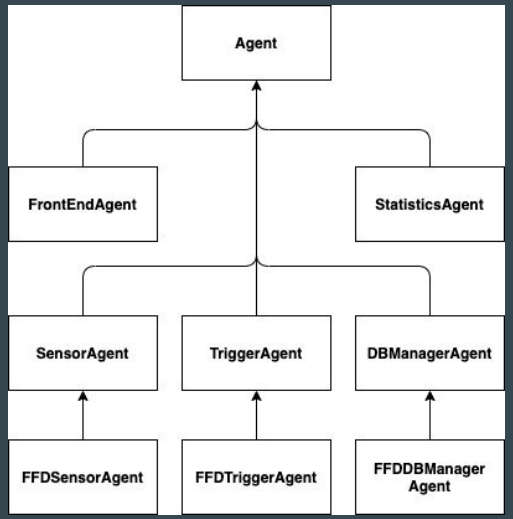
# Analisi Ontologie Riusabili

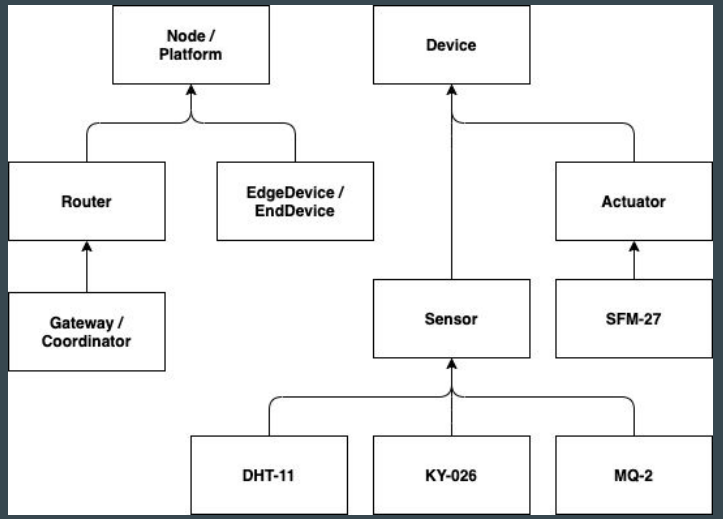
Raramente è necessario partire da zero nella definizione di una nuova ontologia: quasi sempre è disponibile una ontologia di terze parti che fornisce un buon punto di partenza. Le ontologie esistenti individuate come base di sviluppo sono:

- **SSN (Semantic Sensor Network Ontology)** - Ontologia che descrive sensori, attuatori, osservazioni e concetti relativi.
- **SOSA (Sensor, Observation, Sample, and Actuator Ontology)** - Ontologia basata sull'ontologia SSN sviluppata dal W3C Semantic Sensor Networks Incubator Group (SSN-XG) con considerazioni dal W3C/OGC Spatial Data on the Web Working Group.
- **QUDT** - Ontologia che descrive specifiche riguardanti unita` di misura, tipo di quantita`, dimensioni e tipi di dato.
- **GEO** - Ontologia per rappresentare latitudine, longitudine e altre informazioni relative a oggetti spatially-located.

# Definizione Tassonomie e Proprietà

Con tassonomia si intende la disciplina che si occupa della classificazione gerarchica di elementi. Dopo aver identificato i termini rilevanti `e necessario organizzarli in una ben precisa tassonomia gerarchica che permetta il rispetto della semantica dei costrutti primitivi forniti dalle tecnologie adottate.



# Definizione Schema RDF

Lo schema RDF permette la definizione del vocabolario utilizzato nel data model RDF. In particolare consente di descrivere quali proprietà sono applicate a quali tipologie di oggetti, quali valori esse possono assumere e le relazioni che intercorrono tra gli oggetti.

In RDFLib è possibile definire sia lo schema RDFS che le istanze RDF tramite l'inserimento di triple in un grafo aventi la forma standard (subject, predicate, object).

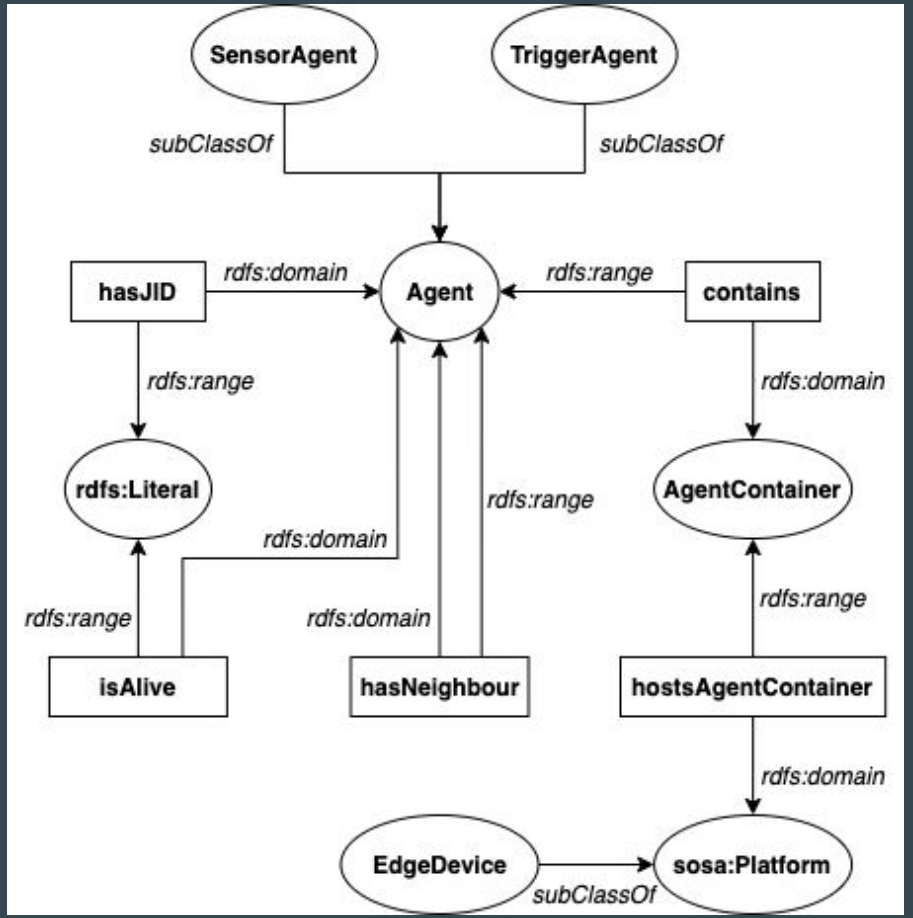
```
wsd ffd rdf : Graph = Graph()
base ontology: str = ’http://wsn=ffd.org/’
dht11 = URIRef( value=’DHT11’ ,base=base_ontology )
wsd_ffd_rdf.add((dht11, RDF.type, RDFS.Class))
wsd_ffd_rdf.add((dht11, RDFS.comment,
        Literal ( ’DHT11 = humidity and temperature sensor ’)))
wsd_ffd_rdf.add((dht11, RDFS.seeAlso,
        Literal ( ’ https ://www. adafruit .com/product /386 ’)))
wsd_ffd_rdf.add((dht11, RDFS.subClassOf, SSN.Sensor))
```

- Definizione della **classe DHT11**: sensore presente su ogni nodo della rete che permette di rilevare temperatura e umidità dell’ambiente in cui si trova.
- Estende la classe Sensor definita nell’ontologia SSN.

- Definizione **proprietà TempAccuracy** della classe DHT11: rappresenta l’accuratezza del sensore nel misurare la temperatura.
- Estende la proprietà Accuracy definita nell’ontologia SSN_SYSTEM

```
temp accuracy = URIRef( value=’TempAccuracy ’ , base=base_ontology )
wsd_ffd_rdf.add((temp accuracy, RDF.type, RDF.Property))
wsd_ffd_rdf.add((temp accuracy , RDFS.comment,
        Literal ( ’Temperature accuracy ’)))
wsd_ffd_rdf.add((temp accuracy, RDFS.domain, dht11))
wsd_ffd_rdf.add((temp accuracy, RDFS.range, RDFS.Literal))
wsd_ffd_rdf.add((temp accuracy , RDFS.subPropertyOf ,
        SSN SYSTEM. Accuracy ))
```

La proprietà ***hostsAgentContainer***, con cui `e stata estesa la classe Platform dell'ontologia SOSA, rappresenta il punto di connessione tra i due domini Wireless Sensor Network e Multi-Agent System. Essa infatti definisce la relazione tra la piattaforma hardware della rete di sensori, cioè il nodo, e l'insieme/container di agenti, modellato tramite la classe AgentContainer, che utilizza tale piattaforma come ambiente di esecuzione.



# Definizione RDF

Una ontologia, per essere definita tale deve rispettare alcune proprietà, tra cui:

- **Linguaggio formale** - Il formalismo è una condizione necessaria per permettere alle macchine l'elaborazione delle informazioni.
- **Assunzioni su un dominio** - Definisce/modella uno specifico dominio concettualizzando aspetti reali.
- **Condivisa** - La comunità ritiene adeguata la modellazione del dominio fatta, cioè diversi utilizzatori hanno la stessa visione del dominio modellato.

```
edge_device_instance = URIRef( value='ED0000001 ' , base=base_ontology)
wsd_ffd_rdf.add((edge_device_instance, RDF.type, edge_device))
wsd_ffd_rdf.add(( edge_device_instance , GEO. lat , Literal (55.701)))
wsd_ffd_rdf.add((edge_device_instance , GEO.long, Literal (12.552)))
wsd_ffd_rdf.add(( edge_device_instance , SOSA. hosts , Bag( wsd_ffd_rdf, BNode() ,
        [ dht11_instance , mq2_instance , ky026_instance , sfm27_instance ]).uri ))
wsd_ffd_rdf.add(( edge_device_instance , is_routed_by , Bag( wsd_ffd_rdf , BNode() , [ router_instance ]). uri ))
```

- Esempio di definizione di una istanza della classe **EdgeDevice** utilizzando RDFLib
- Tramite l'ontologia **GEO** è definita la sua posizione nello spazio.
- **Elemento container Bag** (fornito da RDF) per indicare una collezione di sensori e attuatori presenti fisicamente sul nodo.
- Elemento container Bag per indicare una collezione di nodi Router della rete in grado di instradare i dati verso questo nodo.

# Estensione OWL

Con **OWL** (Ontology Web Language) si intende un semantic markup language standard W3C per la rappresentazione della conoscenza tramite la definizione e la condivisione di ontologie sul web, fornendo una maggiore espressività rispetto a RDF/RDFS.

RDF/RDFS soffrono la mancanza di alcune features importanti (presenti in OWL), ad esempio:

- Definizione di range delle proprietà che si applicano solamente ad alcune classi.
- Disgiunzione tra le classi. In RDFS è possibile definire solo relazioni del tipo subClassOf.
- Definizione di nuove classi tramite la combinazione di altre classi utilizzando operatori come unione, intersezione e complemento.
- Definizione di vincoli di cardinalità delle proprietà.

```
OWL: Namespace = Namespace('http://www.w3.org/2002/07/owl#')
wsd ffd owl.bind('owl', OWL)
```

Import e binding namespace OWL in RDFLib

Tramite il linguaggio OWL è stato possibile definire associazioni/vincoli tra le classi e le proprietà che compongono il sistema. Alcuni dei costrutti utilizzati sono:

- **Restriction** - Utilizzato per modellare i vincoli del dominio.
- **onProperty** - Utilizzato per associare i vincoli alle specifiche proprietà.
- **inverseOf** - Utilizzato per esplicitare proprietà che sono una l'inverso dell'altra.
- **disjointWith** - Utilizzato per esplicitare classi disgiunte fra loro.

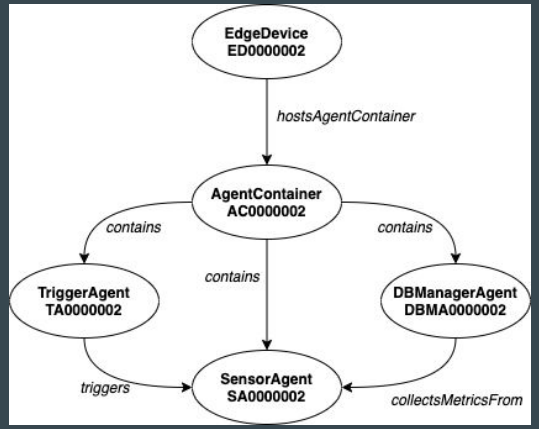
```
base_rdf_ontology_uri:str =
'https://gitlab.com/paganelli.f/ws-project-paganelli-1920/-/raw/develop/wsd-ffd-pretty-xml.rdf'

RDF_ONTOLOGY: Namespace = Namespace(base_rdf_ontology_uri)
wsd_ffd_owl.bind('wsn-ffd-rdf', MY_RDF_ONTOLOGY)
```

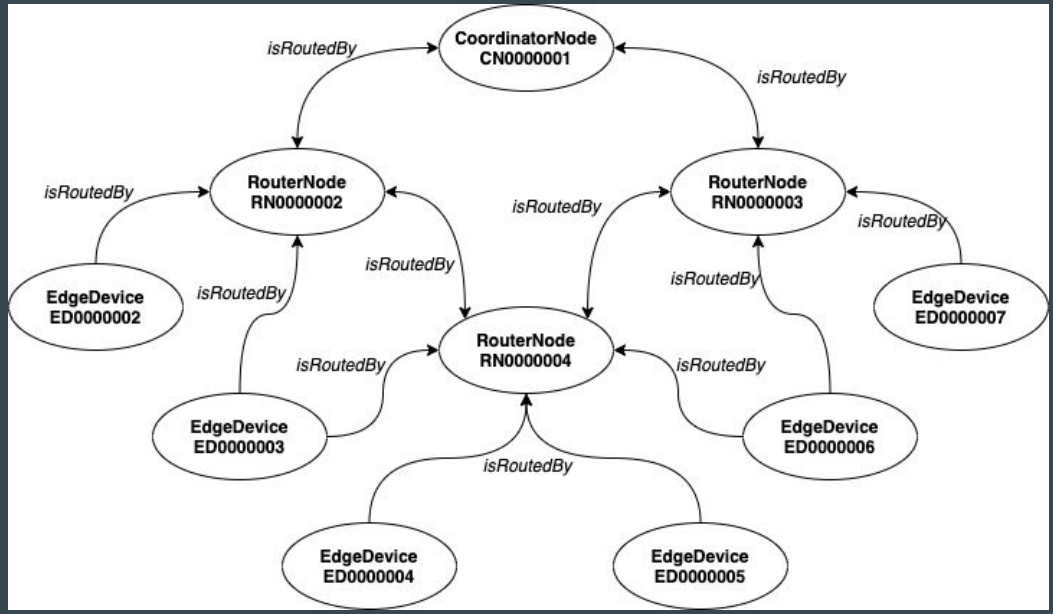
Import ontologia RDF definita nel secondo assignment per estensione OWL del terzo assignment con RDFLib. Dopo aver effettuato il binding dell'ontologia al nome *wsn-ffd-rdf* è possibile utilizzare classi e proprietà in essa definite con il prefisso **wsn-ffd-rdf:** (i.e. wsn-ffd-rdf:SerialNum).

La WSN d'esempio modellata in OWL tramite la definizione delle istanze è composta da:

- un nodo coordinator
- tre nodi router
- sei nodi edge

Le proprietà **canRoute** e **isRoutedBy** permettono di definire la topologia WSN.




Per ogni nodo edge della rete è definito un relativo **AgentContainer** che rappresenta l'ambiente di esecuzione degli agenti che compongono il MAS.

# Serializzazione Grafo

Una delle importanti features della libreria RDFLib è la serializzazione del grafo RDF definito in diversi formati e diversi encoding. Per il progetto il grafo è stato serializzato nei formati

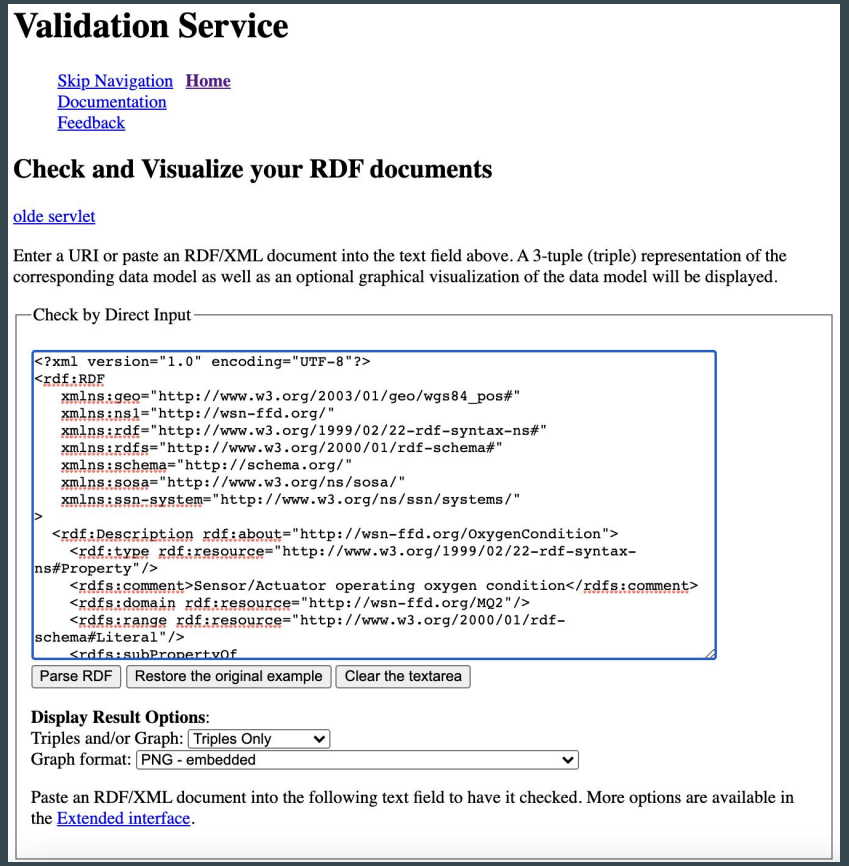
- xml
- pretty-xml
- turtle

```
wsd_ffd_rdf.serialize(
destination=os.getcwd()+'/wsd-ffd-xml.rdf ',
format='xml ')

wsd_ffd_rdf.serialize(
destination=os.getcwd()+'/wsd-ffd-pretty-xml.rdf ',
format='pretty=xml ')

wsd_ffd_rdf.serialize(
destination=os.getcwd()+'/wsd-ffd-turtle.ttl ',
format='turtle ')
```

# Validazione

- Ogni documento serializzato tramite RDFLib è stato validato utilizzando il tool ufficiale W3C (https://www.w3.org/RDF/Validator/).

- La validazione ha avuto esito positivo per tutti e tre i formati serializzati (xml, pretty-xml e turtle), sia nel secondo assignment che nel terzo assignment indicando quindi una buona qualità di serializzazione dei documenti da parte della libreria RDFLib.



**Validation Service**

Skip Navigation **Home**
Documentation
Feedback

**Check and Visualize your RDF documents**

olde servlet

Enter a URI or paste an RDF/XML document into the text field above. A 3-tuple (triple) representation of the corresponding data model as well as an optional graphical visualization of the data model will be displayed.

Check by Direct Input

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
   xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#"
   xmlns:ns1="http://wsn-ffd.org/"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:schema="http://schema.org/"
   xmlns:sosa="http://www.w3.org/ns/sosa/"
   xmlns:ssn-system="http://www.w3.org/ns/ssn/systems/"
>
  <rdf:Description rdf:about="http://wsn-ffd.org/OxygenCondition">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property"/>
    <rdfs:comment>Sensor/Actuator operating oxygen condition</rdfs:comment>
    <rdfs:domain rdf:resource="http://wsn-ffd.org/MQ2"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Literal"/>
    <rdfs:subPropertyOf
```

Parse RDF | Restore the original example | Clear the textarea

**Display Result Options**:
Triples and/or Graph: Triples Only
Graph format: PNG - embedded

Paste an RDF/XML document into the following text field to have it checked. More options are available in the Extended interface.

# SPARQL

- Recommendation W3C RDF Query Language
- La versione latest è la 1.1, rilasciata il 21 Marzo 2013
- Supportato dalla maggior parte dei tool per il Web Semantico, incluso RDFLib
- Nel Semantic Web Stack si posiziona al livello di RDF e non a quello più basso del formato/sintassi specifico (i.e. XML Query Language). Questo permette di utilizzare la stessa sintassi per fare query sia su documenti xml che pretty-xml che turtle perchè SPARQL conosce il data model RDF.

```
prefix sosa: <http://www.w3.org/ns/sosa/>
prefix ssn: <http://www.w3.org/ns/ssn/>
prefix ssn-system: <http://www.w3.org/ns/ssn/systems/>
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
prefix qudt-1-1: <http://qudt.org/1.1/schema/qudt#>
prefix qudt-unit-1-1: <http://qudt.org/1.1/vocab/unit#>
prefix schema: <http://schema.org/>
prefix wsn-ffd-rdf: <https://gitlab.com/paganelli.f/ws-project-paganelli-1920/-/raw/develop/wsd-ffd-pretty-xml.rdf>
select ?n ?serial_num ?lat ?long
where {
   { ?n rdf:type <http://www.w3.org/2002/07/owl#EdgeDevice>}
      union { ?n rdf:type <http://wsn-ffd.org/EdgeDevice>}
      union { ?n rdf:type <http://www.w3.org/2002/07/owl#RouterNode>}
      union { ?n rdf:type <http://www.w3.org/2002/07/owl#CoordinatorNode>} .
   ?n geo:lat ?lat .
   ?n geo:long ?long .
   ?n wsn-ffd-rdf:SerialNum ?serial_num .
}
order by asc(?serial_num)
```

Esempio di query SPARQL: restituisce la posizione geografica (latitudine e longitudine) di tutti i nodi della Wireless Sensor Network, ordinati in modo crescente per numero seriale

```
prefix sosa: <http://www.w3.org/ns/sosa/>
prefix ssn: <http://www.w3.org/ns/ssn/>
prefix ssn-system: <http://www.w3.org/ns/ssn/systems/>
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
prefix qudt-1-1: <http://qudt.org/1.1/schema/qudt#>
prefix qudt-unit-1-1: <http://qudt.org/1.1/vocab/unit#>
prefix schema: <http://schema.org/>
prefix wsn-ffd-rdf: <https://gitlab.com/paganelli.f/ws-project-paganelli-1920/-/raw/develop/wsd-ffd-pretty-xml.rdf>
select ?n ?is_routed_by ?serial_num
where {
    ?n wsn-ffd-rdf:isRoutedBy ?is_routed_by .
    filter (?serial_num != 'IT234GJ55500001') .
}
```

Esempio di query SPARQL: restituisce la routing table della Wireless Sensor Network, ovvero tutti i percorsi di rete (di lunghezza 1) escluso il nodo con numero seriale 'IT234GJ55500001' (nodo coordinator)

# Risorse/Link:

- Grigoris Antoniou and Frank van Harmelen, *A Semantic Web Primer*, The MIT Press
- Toby Segaran and Colin Evans, Jamie Taylor, *Programming the Semantic Web*, O'Reilly
- https://gitlab.com/paganelli.f/ws-project-paganelli-2021
- https://gitlab.com/paganelli.f/sd-project-paganelli-1920
- https://gitlab.com/paganelli.f/sctm-project-paganelli-2021
- https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/
- https://neo4j.com/blog/other-graph-database-technologies/?ref=blog
- https://github.com/RDFLib/rdflib
- https://www.w3.org/2001/sw/wiki/RDFLib