Sviluppo di un'applicazione Java ontology-based per la catalogazione e classificazione di esopianeti

ST1414 Modellazione e Gestione della Conoscenza

Alessio Rubicini

Corso di laurea triennale in Informatica per la comunicazione digitale
Università degli Studi di Camerino
Anno Accademico 2022/23

1 Dominio e ambito applicativo

L'avanzamento della tecnologia negli ultimi decenni ha portato a un rapido salto in avanti in campo scientifico. In particolare, nel campo dell'astronomia, è stata posta molta attenzione sulla ricerca degli esopianeti, pianeti che orbitano attorno a stelle diverse dal nostro Sole. L'individuazione e lo studio di tali corpi celesti rivestono un'importanza cruciale per comprendere l'origine, l'evoluzione e la diversità del nostro universo. La scoperta di mondi che possono ospitare vita, o che presentano caratteristiche ambientali simili alla Terra, alimenta non solo la nostra curiosità sull'esistenza di forme di vita extraterrestri, ma anche la nostra prospettiva di diventare una specie multi-planetaria.

Tuttavia, con l'aumento esponenziale del numero di scoperte, e della conseguente mole di dati da gestire, sorge la necessità di strumenti e metodologie efficaci per la catalogazione e la classificazione di questi corpi celesti. Questo progetto si propone di sviluppare un'applicazione basata su ontologia per agevolare il processo di catalogazione degli esopianeti.

L'obiettivo principale è quello di fornire ai ricercatori un'interfaccia semplice e intuitiva che consenta di catalogare e consultare gli esopianeti di proprio interesse. L'utilizzo di un'ontologia come fondamento per il modello dei dati permette di

- rappresentare in modo strutturato e formale le conoscenze relative agli esopianeti, consentendo la categorizzazione e l'interrogazione dei dati in modo coerente e organizzato;
- effettuare reasoning sui dati, cioè applicare un ragionamento logico e deduttivo per estrarre informazioni implicite e inferire nuove conoscenze dalla struttura e dai dati presenti nell'ontologia stessa. In questo contesto, in particolare, il reasoning viene impiegato per la classificazione ad esempio delle stelle nei vari tipi spettrali e dei pianeti nelle varie sotto-categorie, tra cui quella dei pianeti potenzialmente abitabili, ambito di ricerca fondamentale come espresso in precedenza.

L'ontologia si concentra strettamente sul dominio degli esopianeti, rimanendo però aperta ad estensioni. Questo dominio include le caratteristiche fisiche, orbitali e atmosferiche dei pianeti, nonché informazioni relative alle stelle ospiti, come il tipo spettrale, la massa e la temperatura, poiché queste proprietà influenzano direttamente le caratteristiche dei pianeti in orbita. Sono inoltre considerate informazioni riguardanti i metodi di rilevamento impiegati per la scoperta e l'osservazione degli esopianeti, nonché gli strumenti di osservazione utilizzati e le relative missioni spaziali di appartenenza.

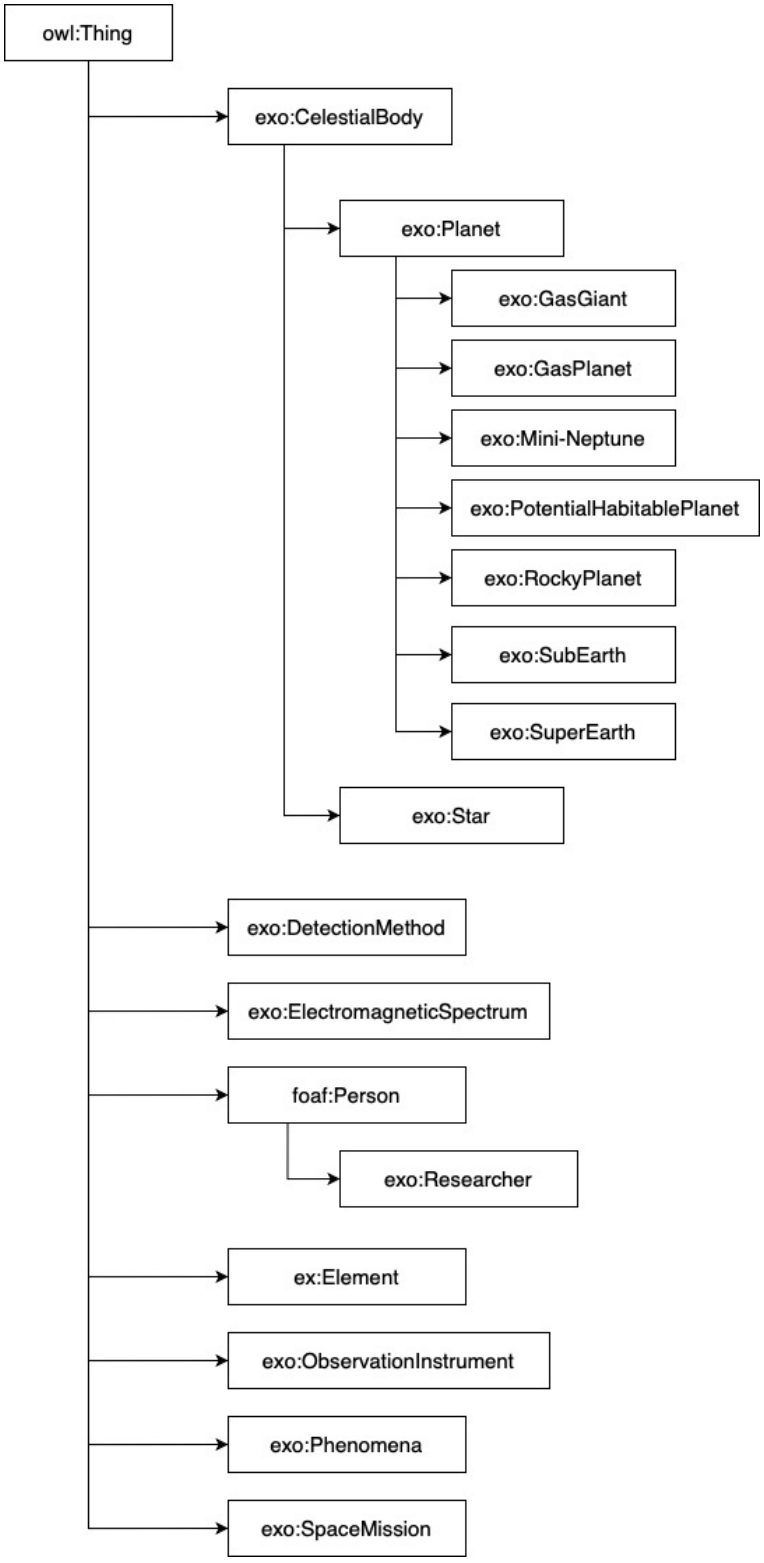
Per integrare l'ontologia creata con i vocabolari presenti online, per la realizzazione della stessa sono stati importati e utilizzati i seguenti vocabolari:

- **FOAF (Friend of a Friend)** [1]: una specifica semantica che fornisce un vocabolario per descrivere informazioni personali e relazioni sociali su di sé e sulle altre persone.
- **OM (Ontology of units of Measure)** [2]: si occupa di definire e rappresentare concetti e relazioni legate alle unità di misura, alle quantità, alle misurazioni e alle dimensioni nel contesto della ricerca scientifica.

Per una migliore integrazione sarebbe opportuno importare anche il vocabolario **SWEET (Semantic Web for Earth and Environment Technology Ontology)** [3], un'ontologia originariamente sviluppata dal *NASA Jet Propulsion Laboratory* che contiene oltre 6000 concetti organizzati in 200 ontologie rappresentate in OWL che includono rappresentazioni scientifiche, fenomeni, processi e attività umane. Date però le grandi dimensioni di questo vocabolario e lo scopo didattico del presente progetto, è stato deciso di non importare l'ontologia SWEET per mantenere chiarezza e pulizia nell'ontologia creata.

Di seguito viene allegata la struttura complessiva dell'ontologia realizzata, visualizzabile in modo più chiaro e dettagliato sul portale OWLGrEd o consultando il file `exoplanet-ontology.rdf` presente

nella directory di progetto tramite apposito software.



2 Ontologia e inferenza

Come affermato in precedenza, uno dei vantaggi principali nell'utilizzo di una ontologia come base di dati è la possibilità di fare inferenza sui dati. Per inferenza si intende il processo di fare deduzioni e trarre conclusioni logiche basate su conoscenze o informazioni già esistenti, utilizzando regole, assiomi e relazioni prestabiliti per ricavare nuove informazioni che potrebbero non essere esplicitamente dichiarate o rappresentate nei dati forniti. Rappresentando la conoscenza del dominio in modo formale, le ontologie consentono ai motori di ragionamento automatizzati (reasoner) di trarre conclusioni, fare deduzioni e rispondere a domande complesse sulla base delle informazioni disponibili.

2.1 Obiettivi

Nell'ambito di questo progetto, l'utilizzo del ragionamento logico e dell'inferenza sull'ontologia realizzata ha come obiettivo quello di classificare i corpi celesti descritti.

Più nello specifico, grazie alle relazioni e alle proprietà descritte nell'ontologia, è possibile

- classificare i pianeti in base alla composizione (gassoso o terrestre), alla massa (gigante, mini-Nettuno, super Terra e sub Terra) e alla loro eventuale e potenziale abitabilità. Riguardo quest'ultimo punto è bene specificare che dal punto di vista scientifico non esistono parametri ben definiti per dedurre l'abitabilità di un esopianeta in quanto essa dipende da vari fattori e dalle definizioni e modelli utilizzati dagli scienziati. Ai fini del progetto, sono stati definiti dei range di valori limite in cui i pianeti sono soliti presentare caratteristiche adatte alla presenza della vita;
- classificare le stelle sulla base del loro spettro, e più nello specifico della loro temperatura superficiale. A tal scopo è stata presa come riferimento la classificazione Harvard che distingue sette classi spettrali contrassegnate nell'ordine dalle lettere O, B, A, F, G, K, M;
- classificare gli strumenti di osservazione sulla base del loro spettro elettromagnetico di osservazione.

2.2 Strumenti impiegati

Per implementare la gestione dell'ontologia e l'inferenza descritta precedentemente nell'applicazione, sono state impiegate le seguenti API:

- **Apache Jena** [5]: un framework open source per lo sviluppo di applicazioni che gestiscono linked data. Fornisce varie funzionalità tra cui creazione e lettura di grafi RDF, esecuzione di query SPARQL, interazione con ontologie RDFS e OWL e motori di inferenza.
- **Pellet (Openllet)** [4]: un reasoner OWL-DL open source, compatibile con Apache Jena, che fornisce funzionalità di verifica della coerenza delle ontologie, computazione della gerarchia delle classi, spiegazione delle inferenze e risposte alle query SPARQL.

L'utilizzo di un reasoner esterno si è reso necessario dal momento che, come spiegato nella documentazione ufficiale di Apache Jena, nessuno dei reasoner OWL forniti dalla versione corrente di Jena è completo in senso tecnico e le prestazioni (in particolare l'uso della memoria) del reasoner più completo lasciano ancora a desiderare.

2.3 Note sulle prestazioni

Non essendo fondamentale per i concetti descritti e per la parte di inferenza, ai fini di dimostrazione del progetto, nell'applicazione realizzata è stato omesso il caricamento dell'ontologia esterna OM2 per le unità di misura, in quanto, viste le sue notevoli dimensioni, incrementa notevolmente i tempi di caricamento e inferenza sull'ontologia.

3 Scenari di utilizzo

Di seguito sono descritti i vari scenari di utilizzo dell'applicazione rilevanti per il dominio considerato, con le rispettive query di interrogazione dell'ontologia in linguaggio SPARQL.

Si noti che il formato di output è uniforme tra tutte le query, in modo da unificare il più possibile il formato dei risultati e facilitare la loro interpretazione lato applicazione.

I prefissi utilizzati sono i seguenti:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX om: <http://www.ontology-of-units-of-measure.org/resource/om-2/>
PREFIX exo: <https://www.unicam.it/cs/alessiorubicini/exoplanet-ontology#>
```

3.1 Consultazione completa del catalogo

All'avvio dell'applicazione viene caricato il catalogo completo degli esopianeti presenti nell'ontologia. La query utilizzata seleziona tutti gli individui di tipo `Planet`, restituendo per ognuno il suo il suo URI e il suo nome.

```
SELECT  ?label ?value
WHERE {
    ?planet rdf:type exo:Planet .
    BIND(?planet AS ?label) .
    ?planet rdfs:label ?value .
}
```

3.2 Visualizzazione dei dettagli di un pianeta

Quando l'utente seleziona un pianeta dal catalogo, l'applicazione carica le informazioni relative a quel pianeta nella schermata di dettaglio. La query utilizzata seleziona tutte le triple che hanno come soggetto un individuo di tipo `Planet` con il nome specificato e le varie proprietà associate ad esso.

```
SELECT ?label ?value
 WHERE {
  ?planet rdf:type exo:Planet .
  ?planet rdfs:label "NOME_PIANETA" .
  ?planet ?label ?value .
 }
```

3.3 Ricerca dei pianeti per nome

Quando l'utente immette il nome di un pianeta nell'apposita barra di ricerca, l'applicazione cerca nel catalogo i pianeti con il nome corrispondente visualizzandoli. La query utilizzata seleziona tutti gli individui di tipo pianeta il cui nome (label) inizia con la stringa specificata.

```
SELECT  ?label ?value
WHERE {
 ?planet rdf:type exo:Planet .
```

```
 BIND(?planet AS ?label) .
 ?planet rdfs:label ?value .
 FILTER(STRSTARTS(?value, "RICERCA"))
}
```

3.4 Inserimento di un nuovo pianeta

Inserisce nuove triple nel grafo RDF di default, con l'obiettivo di aggiungere un nuovo individuo di tipo `Planet` con le relative proprietà.

```
INSERT DATA {
    exo:NOME_INDIVIDUO rdf:type exo:Planet ;
        rdfs:label "VALUE" ;
        exo:planetMass VALUE ;
        exo:gravity VALUE ;
        exo:surfaceTemperature VALUE ;
        exo:surfacePressure VALUE ;
        exo:imageURL "VALUE" ;
        exo:discoveredBy exo:RESARCHER_NAME ;
        ... .
}
```

3.5 Inserimento di una nuova stella

Inserisce nuove triple nel grafo RDF di default, con l'obiettivo di aggiungere un nuovo individuo di tipo `Star` con le relative proprietà.

```
INSERT DATA {
        exo:NOME_STELLA rdf:type exo:Star ;
                rdfs:label "VALUE" ;
                exo:solarMasses VALUE ;
                exo:solarRadius VALUE ;
                exo:distanceFromEarth VALUE ;
                exo:rotationalPeriod VALUE ;
                exo:surfaceTemperature VALUE ;
                exo:surfacePressure VALUE ;
                exo:imageURL "VALUE" ;
                exo:hasSatellite CELESTIAL_BODY ;
                ...
}
```

4 Descrizione delle responsabilità

Nella fase di progettazione del software sono state individuate le seguenti responsabilità:

- **Costruire il modello dell'ontologia**: costruzione del modello RDF, caricamento dell'ontologia realizzata e creazione del modello inferito.
- **Mantenere le query SPARQL necessarie**: memorizzazione delle query SPARQL da utilizzare nei vari scenari d'utilizzo con le rispettive modalità di accesso e parametrizzazione.
- **Eseguire le query SPARQL sul modello**: preparazione ed esecuzione delle query SPARQL definite sul modello dell'ontologia.
- **Intrepretare (parsing) i risultati delle query effettuate**: decodifica dei dati risultanti dalla query eseguita in un formato conosciuto e facilmente accessibile.
- **Strutturare ed esporre i dati interpretati**: rappresentazione dei dati interpretati e delle rispettive operazioni di accesso alle singole informazioni.
- **Gestire il modello dell'ontologia**: esecuzione di tutte le operazioni di interrogazione, aggiornamento e verifica sull'ontologia.
- **Gestire lo stato dell'applicazione**: gestione delle interazioni tra l'interfaccia utente e il modello sottostante rappresentato dall'ontologia.
- **Presentare i dati interpretati sull'interfaccia**: stampa dei singoli dati nei rispettivi elementi dell'interfaccia utente.
- **Gestire l'interfaccia utente**: gestione degli elementi dell'interfaccia e dei rispetti eventi.

5 Implementazione delle responsabilità

5.1 Costruzione del modello dell'ontologia

Le modalità di costruzione del modello dell'ontologia sono descritte dall'interfaccia `ModelBuilder` che definisce il contratto comune per la costruzione di modelli RDF.

Nel dettaglio, l'interfaccia è implementata dalla classe `DefaultModelBuilder` che si occupa di creare e caricare modelli RDF standard, a sua volta estesa dalla classe `InferredModelBuilder` che aggiunge funzionalità di costruzione di modelli inferiti, utili nel contesto di questo progetto.

5.2 Mantenimento delle query SPARQL

Dal momento che le singole query SPARQL definite precedentemente per i vari scenari di utilizzo (3) sono ben definite e costanti, è possibile implementare questa responsabilità in modi diversi. In questo contesto, è stato scelto come approccio l'utilizzo delle enumerazioni.

L'interfaccia `SparqlQueries` definisce il contratto comune per la definizione, il mantenimento e le operazioni di accesso e parametrizzazione di query SPARQL. Essa è poi implementata da due enumerazioni `SelectionQueries` e `UpdateQueries` che si occupano di mantenere, rispettivamente, le query di selezione dei dati e le query di aggiornamento dei dati.

5.3 Esecuzione delle query SPARQL

L'interfaccia `QueryExecutor` definisce il contratto d'uso comune per l'esecuzione di query SPARQL tramite le API Jena ed è implementata dalla classe `OntologyQueryExecutor`.

5.4 Parsing dei risultati delle query

Il contratto comune per l'interpretazione (parsing) dei risultati delle query SPARQL è definito dall'interfaccia `DataParser`. La classe `JSONParser` fornisce una implementazione dell'interfaccia per il parsing dei dati verso una struttura dati JSON-like con coppie chiave-valore. In aggiunta la classe implementa anche le modalità di parsing dei singoli nodi e dei label RDF delle risorse ottenute dall'ontologia.

5.5 Strutturazione ed esposizione dei dati

L'interfaccia `ParsedData` definisce le modalità di presentazione e accesso ai dati interpretati dai risultati di una query. La classe `JSONData` implementa l'interfaccia ed espone i dati sempre in un formato JSON-like, agendo quindi da wrapper.

5.6 Gestione del modello dell'ontologia

La responsabilità di gestire il modello dell'ontologia sottostante all'applicazione mettendolo in comunicazione il controller dell'applicazione è stata affidata ad una apposita classe `OntologyController`.

Questa classe mantiene il modello istanziato dell'ontologia, il costruttore del modello RDF e l'esecutore delle query SPARQL.

5.7 Gestione dello stato dell'applicazione

La classe `Controller` si occupa di mettere in comunicazione l'interfaccia utente (view) con il controllore dell'ontologia, e di conseguenza con il modello sottostante (model), seguendo così il pattern architetturale del Model-view-controller (MVC).

5.8 Presentazione dei dati sull'interfaccia

Il contratto comune per il rendering dei dati sull'interfaccia utente è definito dall'interfaccia `DataRenderer`. Per ogni funzionalità dell'applicativo l'interfaccia viene poi implementata da una apposita classe di rendering.

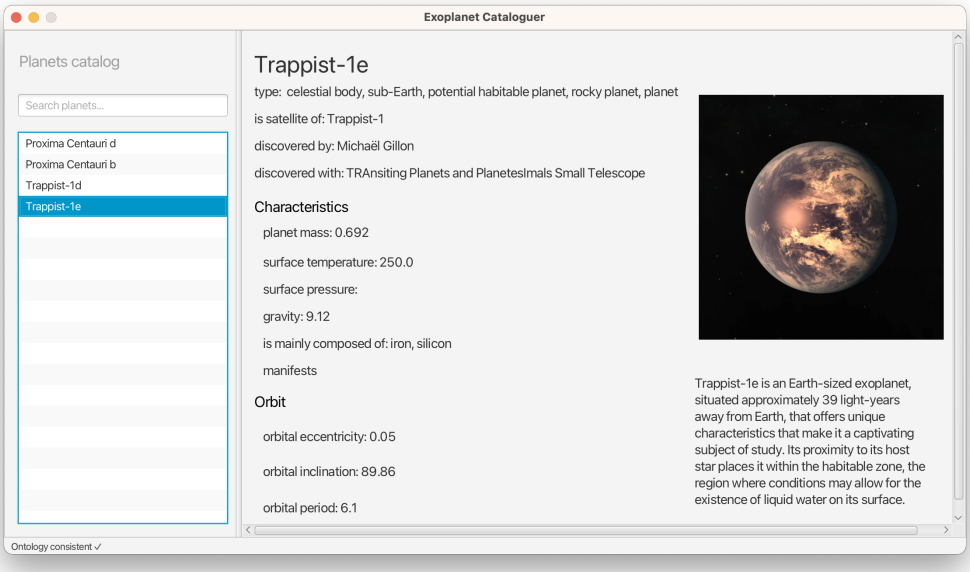
In questo contesto, sono state realizzate due classi che la implementano. `PlanetListRenderer` che si occupa di stampare la lista dei pianeti presenti nell'ontologia, e `PlanetDetailsRenderer` che si occupa di stampare tutti i dettagli di un pianeta selezionato nell'apposita finestra di dettaglio.

5.9 Gestione dell'interfaccia utente

Infine, la finestra principale dell'interfaccia utente è controllata dalla classe `AppFXController` che si occupa della gestione degli elementi dell'interfaccia grafica e della gestione degli eventi.

6 Uso dell'applicazione nel dominio

Grazie all'utilizzo delle ontologie, l'applicazione realizzata potrebbe svolgere un ruolo cruciale nel monitoraggio e nell'aggiornamento dei dati relativi agli esopianeti, diventando uno strumento centrale per la raccolta e l'integrazione di informazioni provenienti da diverse fonti come archivi scientifici, pubblicazioni, cataloghi e altri database, consentendo ai ricercatori di mantenere un catalogo accurato, aggiornato e soprattutto condiviso delle scoperte.



Facendo già affidamento sulle API di Jena e su una ontologia definita tramite standard (RDF/OWL), sarebbe facile collegare ulteriormente l'applicazione a vocabolari e dataset presenti online, portando conseguentemente tutti i benifici derivati dall'utilizzo dei **linked data** quali accesso pubblico ai dati, riusabilità degli stessi, possibilità di aggregarli dinamicamente, utilizzo di API condivise e uniche e aggiornamento continuo dei dati senza necessità di alcuna azione.

Grazie alle capacità di inferenza sarebbe possibile non solo inferire nuove informazioni su un largo dataset di dati, ma anche effettuare una validazione, verificando la conformità dei dati importati e identificando eventuali inconsistenze o errori. Inoltre, l'applicazione può consentire agli utenti di contribuire con nuovi dati o correzioni, facilitando la collaborazione e il miglioramento continuo del catalogo.

L'applicazione potrebbe poi essere eventualmente estesa con nuove funzionalità, come ad esempio la comparazione fianco a fianco di vari esopianeti presenti nel catalogo, in modo da poter confrontare le loro caratteristiche e individuare pattern, o l'analisi statistica dei dati con framework come Apache Commons Math. L'approccio modulare ed estendibile dell'applicazione permette di aggiungere nuovi moduli o estendere quelli esistenti senza dover apportare modifiche significative all'intera struttura dell'applicazione. Questo offre una maggiore flessibilità e facilità di manutenzione, consentendo di aggiornare l'applicazione con nuove funzionalità in risposta ai progressi scientifici.

Infine l'applicazione può svolgere un ruolo significativo nell'ambito dell'educazione e della divulgazione scientifica, venendo utilizzata come strumento didattico per insegnare agli studenti i concetti fondamentali legati agli esopianeti, facilitando anche il coinvolgimento degli stessi in progetti di ricerca.

Bibliografia

[1] http://xmlns.com/foaf/0.1/. *FOAF (friend of a friend)*. 2004.

[2] https://bioportal.bioontology.org/ontologies/OM. *Ontology of units of Measure*. 2023.

[3] https://bioportal.bioontology.org/ontologies/SWEET. *Semantic Web for Earth and Environment Technology Ontology*. 2022.

[4] https://github.com/Galigator/openllet. *Openllet: An Open Source OWL DL reasoner for Java.*

[5] https://jena.apache.org. *Apache Jena. A free and open source Java framework for building Semantic Web and Linked Data applications.*