Rheinische Friedrich-Wilhelms-Universität Bonn

Master's Thesis in Computer Science

Ontology Webform: Generating web forms using ontologies

Submitted by

Kunal Rout

Matriculation Number: 3190071

Examiner: Dr. Hajira Jabeen

March 9, 2021

ACKNOWLEDGMENT

I would like to thank Dr. Hajira Jabeen for granting me the opportunity to work on this topic and for her ideas, encouragement, and for her continuous effort to help me throughout the thesis.

I would also like to thank Prof. Dr. Elena Demidova for evalutating my work.

DECLARATION OF AUTHORSHIP

I Kunal Rout, declare that the thesis has been composed by myself and that the work has not been submitted for any other degree or professional qualification. I confirm that the work submitted is my own, except where work which has formed part of jointly-authored publications has been included.

Gossaigaon, March 9, 2021

Kunal Rout

CONTENTS

1 INTRODUCTION 1
- 1.1 Motivation 1
- 1.2 Research Questions 1
- 1.3 Contributions 1
- 1.4 Thesis Outline 2

2 BACKGROUND 5
- 2.1 Semantic Web 5
- 2.2 RDF 5
- 2.3 RDFS 6
- 2.4 OWL 8
  - 2.4.1 allValuesFrom 8
  - 2.4.2 someValuesFrom 8
  - 2.4.3 cardinality 8
  - 2.4.4 domain 8
  - 2.4.5 range 9
  - 2.4.6 Types of properties 9
- 2.5 SPARQL 9
- 2.6 ISA TAB 9
  - 2.6.1 ISA-TAB file structure example 10
  - 2.6.2 Equivalent Graph representation of ISA-TAB 11
  - 2.6.3 RDF Representation of ISA-TAB 11

3 ISA-TAB ENRICHMENT ANALYSIS 13
- 3.1 Related Work 13
  - 3.1.1 ISA Tools 13
  - 3.1.2 Bio Graphin system 13
  - 3.1.3 SEEK 14
  - 3.1.4 Ontology Webform 16

4 ISA TAB ONTOLOGY 19
- 4.1 Overview 19
- 4.2 Ordering 19
- 4.3 Relationships 22

5 ONTOLOGY WEBFORM 23
- 5.1 Idea 23
- 5.2 Software Architecture 23
- 5.3 Use Cases 24
- 5.4 ISA TAB ontology 25
- 5.5 Enrichment 27

5.5.1 Using Restrictions 27
5.5.2 Using Domain and Range Information 27
5.6 Data Storage 27
5.7 Validations 28
5.7.1 qualifiedCardinality 28
5.8 Performance 29
5.9 Technologies 29
5.9.1 Apache Jena 29
5.9.2 Java with Tomcat 29
5.9.3 Apache Jena Fuseki 29
6 CONCLUSION AND FUTURE WORK 31

LIST OF FIGURES

Figure 1 ISA Specs 2
Figure 2 Semantic Web Language Stack 6
Figure 3 Expressiveness of RDF formats 7
Figure 4 ISA-TAB file structure 10
Figure 5 ISA-TAB file example 11
Figure 6 Graph representation of an ISA-TAB file 11
Figure 7 RDF representation of an ISA-TAB file 12
Figure 8 BioGraphin Tool 14
Figure 9 SEEK Software Architecture 15
Figure 10 ISA Tools Software Suite 17
Figure 11 Core Entities 20
Figure 12 Object Properties 20
Figure 13 Data Properties 21
Figure 14 Relationships 22
Figure 15 Ontology Webform Software Architecture 24
Figure 16 Ontology Webform Upload Page 25
Figure 17 Ontology Webform Individuals Page 25
Figure 18 Ontology Webform Triples Page 26
Figure 19 Ontology Webform Fuseki Server 26
Figure 20 Qualified Cardinality Restriction Implementation 28

ABSTRACT

The Investigation/Study/Assay (ISA) tab-delimited (TAB) format is a general purpose framework with which to collect and communicate complex metadata (i.e. sample characteristics, technologies used, type of measurements made) from 'omics-based' experiments employing a combination of technologies.[21]

This thesis work aims to help domain experts enrich their data by suggesting existing ISA-TAB data already stored in Database. However, the code is written to be flexible enough to be used for other data types, and not just ISA-TAB. It dynamically reads the provided ontology, and generates a webform to present the ontology in a more human-friendly way. This webform is also capable of taking user inputs, which is then used to generate triples corresponding to the ontology. Along with these it also provides suggestions by pulling data that has been populated by other users from from Fuseki Triplestore.

1

INTRODUCTION

1.1 MOTIVATION

The ISA-TAB[28] framework provides a standard way in which scientists can record and share information such that it is easier to understand for everyone. It contains information about the investigation being conducted, like it's description, date, about it's author, and so on and these investigations are linked to one or multiple studies. A study file describes a unit of research, describing the subjects of study and how they are obtained. Those subjects are then used in one or more assay files, which in turn, describe analytical measurements.[14] Presently this is used to describe and share experiments by a number of institutes.[1]

We have created a semantic representation of ISA-TAB called isa_tab_ontology. This OWL ontology[2] can be used to store the ISA-TAB data as RDF triples[3]. It also helps users to use the community built semantic knowledge to help in better understanding and interoperability. Using SPARQL[4] we can also search this dataset to get insights which in turn can be used by domain experts to enrich their data without being concerned about the implementation details and technical details of the ontology.

1.2 RESEARCH QUESTIONS

The research questions that are being dealt with in this thesis work are the following:

1. Are there existing tools that helps domain experts to enrich their data without being concerned about the technical details?

2. Are there existing ontologies for ISA-TAB?

1.3 CONTRIBUTIONS

In this thesis we searched for existing tools that helps domain experts get suggestions based on existing data to enrich their own data. We analyzed BioGraphin[20] and SEEK[15] in regards to it. Apart from that we searched for

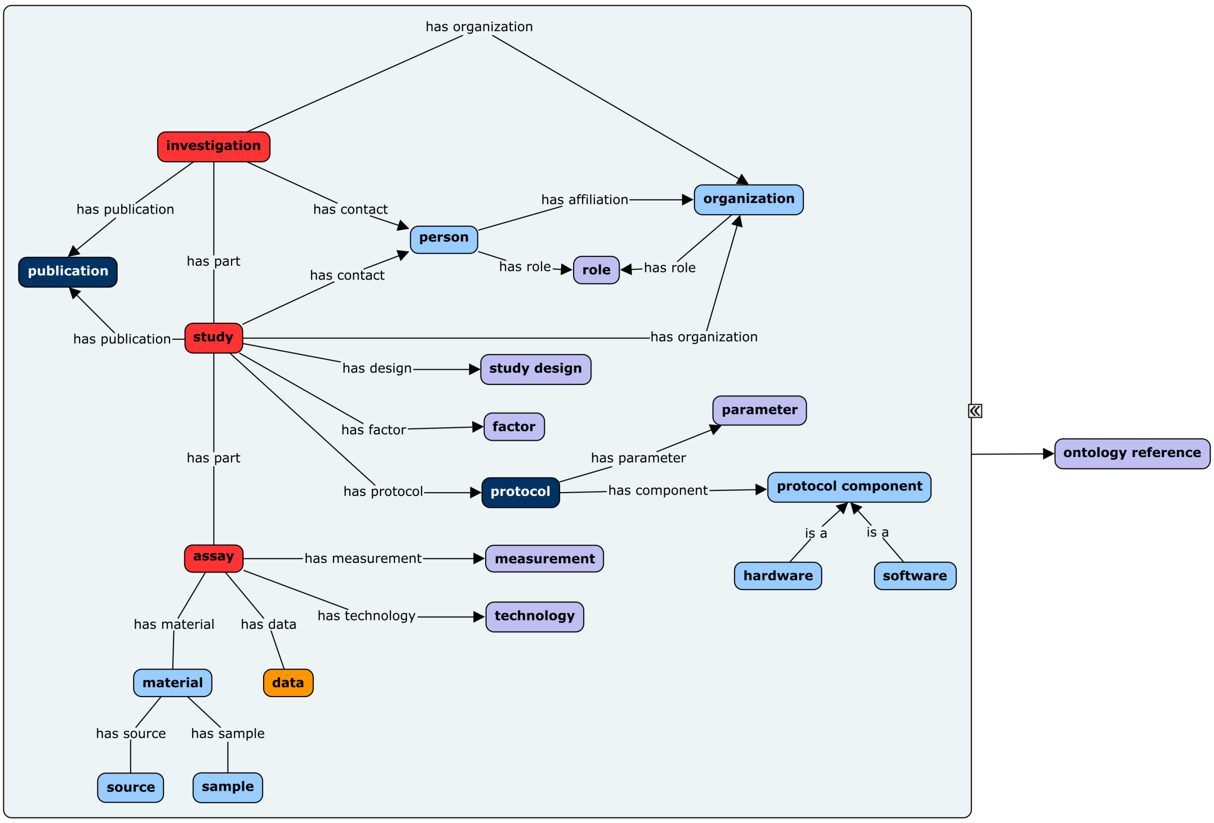


Figure 1: *ISA Specs. Adapted from [5]*

existing ontologies, if any, that exists as an equivalent for ISA-TAB. LinkedISA[14] was found to be an existing work on this but it did not quite match the structure mentioned in the ISA-specs as shown in Figure 1. Therefore, we created an ontology of our own named isa_tab_ontology.owl[1].

Furthermore, we developed a web tool called OntologyWebform based on this isa_tab_ontology.owl. This tool tries to present the ontology in a human-readable webform which helps domain experts understand the ontology. Along with the ontology, text fields are provided where the domain experts can insert their own data. Against these text fields we provide suggestions by reading existing data. These suggestions are meant to help the domain experts enrich their own data. We also tried to provide a number of validations to prevent users from providing wrong data.

1.4 THESIS OUTLINE

In the next chapter we provide the background information required for this thesis. In the following chapter (chapter 3) we discuss some related work that has

[1] https://github.com/2kunal6/OntologyWebform/blob/master/isa_tab_ontology.owl

already been done in this regard. Then in chapter 4 and 5 we provide detailed information of our analysis and the tool that we developed. The last chapter (chapter 6) concludes the discussion with some future possible work that can be done in this regard.

2

BACKGROUND

This chapter deals with discussing the background required for our work. It starts with introducing Semantic Web[29], it's definition and why it's used. Then it discusses RDF[3], which can be thought of as the building blocks of Semantic Web. Then we briefly discuss the various serializations available to express RDFs. The next section touches one of the most important concepts related to Semantic Web, which are RDFS and OWL. RDFS[16] provides schema knowledge to RDF triples, while OWL is used to check the consistency of the knowledge provided by RDF triples. The following part describes SPARQL[4], which is the query language for RDF data. The last part discusses ISA TAB, the domain of our work, in which we are working using the Semantic tools mentioned above.

2.1 SEMANTIC WEB

The term "Semantic Web" refers to W3C's vision of the Web of linked data. Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, and SKOS. [29]. The Semantic Web Language Stack is depicted in Figure 2.

The problem with Web is that machines are unable to interpret data in Web. Semantic Web helps in this regard by providing meaning to data such that machines can extract or understand more. This also helps machines to communicate among themselves.

2.2 RDF

RDF stands for "Resource Description Framework". It is a W3C recommendation since 1998. It can be thought of as a fundamental building block of Semantic Web. It is a universal machine-readable interchange format. The motivation to create RDF came from the fact that there is no unique way in XML to represent knowledge.[26]

The RDF data model comprises of URI, Triples, Resources, Literals, Blank Nodes, Lists, and so on. URIs are used to describe resources unambiguously,

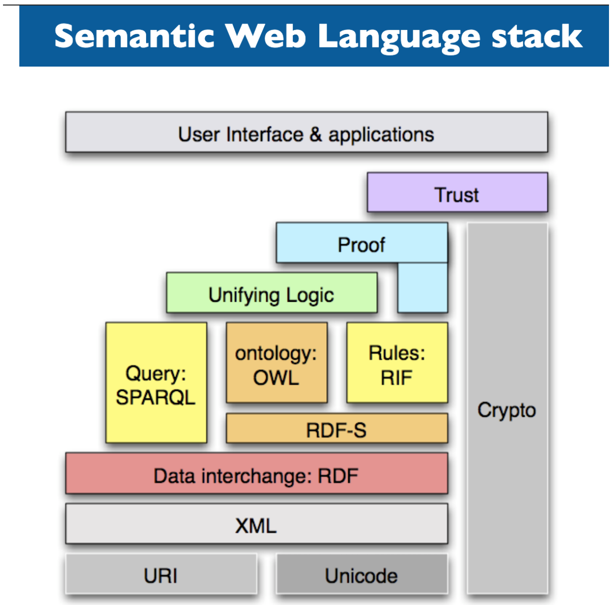


Figure 2: The Semantic Web Language Stack. Adapted from [30].

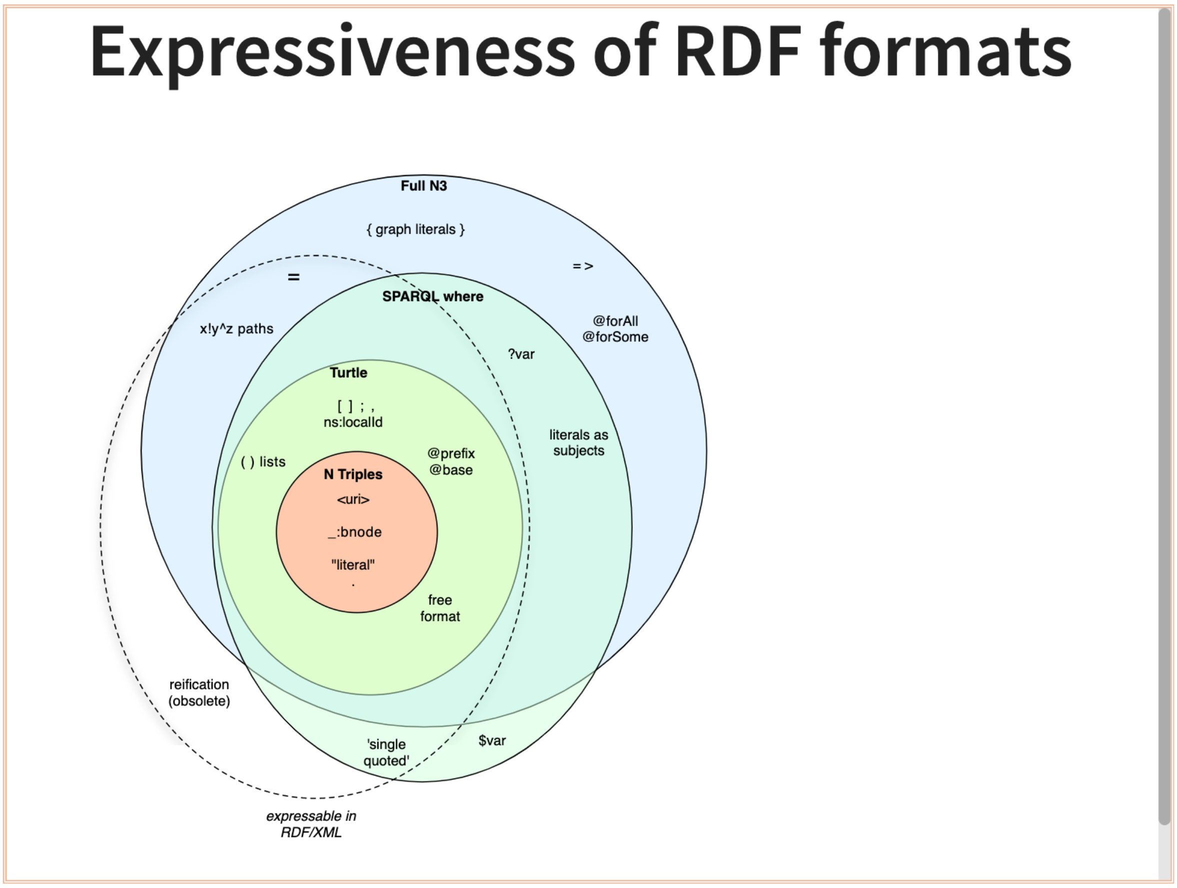
meaning resources can be represented using globally unique URIs. Literals describe data values, using data type. An RDF triple comprises of subject and object connected using a predicate.

There are different serialization formats to store the RDF triples mentioned above like Turtle, RDF/XML, RDFa, N3, JSON-LD, and so on, each with their own qualities. These qualities can be in regards to performance, compatibility, and so on[19]. The expressiveness of these formats is depicted in Figure 3.

2.3 RDFS

RDFS stands for Resource Description Framework Schema, and as the name suggests, it provides schema knowledge to RDF triples. An example of a schema knowledge for the triple "A isMarriedTo B" could be A and B are both Person. However, the schema knowledge could describe more complex things. RDFS uses concepts like Classes, Class hierarchy, Properties and their domain and ranges, lists, and so on to provide domain knowledge.

Although RDFS can express a lot of things, but it has some limitations too. For example, it cannot define the negation of an expression, to define cardinalities, define set of classes, define metadata to schema like version information, and so on. To overcome this we use OWL, which is described below.



Expressiveness of RDF formats

Figure 3: *Expressiveness of RDF formats. Adapted from [18].*

2.4 OWL

OWL stands for Web Ontology Language. OWL ontology consists of classes, properties, individuals (class instances), constraints on properties, and so on. An ontology describes a common vocabulary for shared understanding. In Computer Science it should render something which is machine interpretable. Following are few of the important concepts that we used in our tool:

2.4.1 *allValuesFrom*

The owl:allValuesFrom restriction requires that for every instance of the class that has instances of the specified property, the values of the property are all members of the class indicated by the owl:allValuesFrom clause.[24]

2.4.2 *someValuesFrom*

It is similar to owl:allValuesFrom restriction except that at least one value of the property must be of the class indicated by owl:someValuesFrom.

2.4.3 *cardinality*

owl:cardinality permits the specification of exactly the number of elements in a relation. owl:maxCardinality can be used to specify an upper bound. owl:minCardinality can be used to specify a lower bound. In combination, the two can be used to limit the property's cardinality to a numeric interval.[24]

Cardinality restrictions in OWL allow us to say how many distinct values a property can have for any given subject. Other restrictions tell us about the classes of which those values can or must be members. But these restrictions work independently of one another; we cannot say how many values from a particular class a particular subject can have. A simple example of qualified cardinality is a model of a hand: A hand has five fingers, one of which is a thumb. Qualified cardinalities may seem like a needless modeling detail, and, in fact, a large number of models get by quite fine without them. But models that want to take advantage of detailed cardinality information often find themselves in need of such detailed modeling. This happens especially when modeling the structure of complex objects.[17]

2.4.4 *domain*

For a property one can define (multiple) rdfs:domain axioms. Syntactically, rdfs:domain is a built-in property that links a property (some instance of the class rdf:Property) to a class description. An rdfs:domain axiom asserts that the subjects of such property statements must belong to the class extension of the indicated class description.[25]

2.4.5 *range*

For a property one can define (multiple) rdfs:range axioms. Syntactically, rdfs:range is a built-in property that links a property (some instance of the class rdf:Property) to to either a class description or a data range. An rdfs:range axiom asserts that the values of this property must belong to the class extension of the class description or to data values in the specified data range.[25]

2.4.6 *Types of properties*

There are two types of properties used in OWL, each serving a different purpose as mentioned below:

1. Object Property: used to link two different individuals.
2. Data Property: used to provide a data value corresponding to a relationship for an entity.

2.5 SPARQL

SPARQL stands for "SPARQL Protocol and RDF Query Language" and it is the Query Language for RDF triples. Along with queries it is also capable of manipulating the RDF data. SPARQL queries work on the concept of Graph patterns. Therefore it works well with the Graph formed from the RDF triples.

To store the RDF graph we use a Database. And these Databases are exposed via SPARQL endpoints. The query happens over HTTP using GET and POST.

2.6 ISA TAB

The Investigation/Study/Assay (ISA) tab-delimited (TAB) format is a general purpose framework with which to collect and communicate complex metadata (i.e. sample characteristics, technologies used, type of measurements made) from ’omics-based’ experiments employing a combination of technologies.[21]

ISA-TAB consists of 3 entities:

1. Investigation: The regex for investigation file name is i_.*.txt
2. Study: The regex for study file name is s_.*.txt

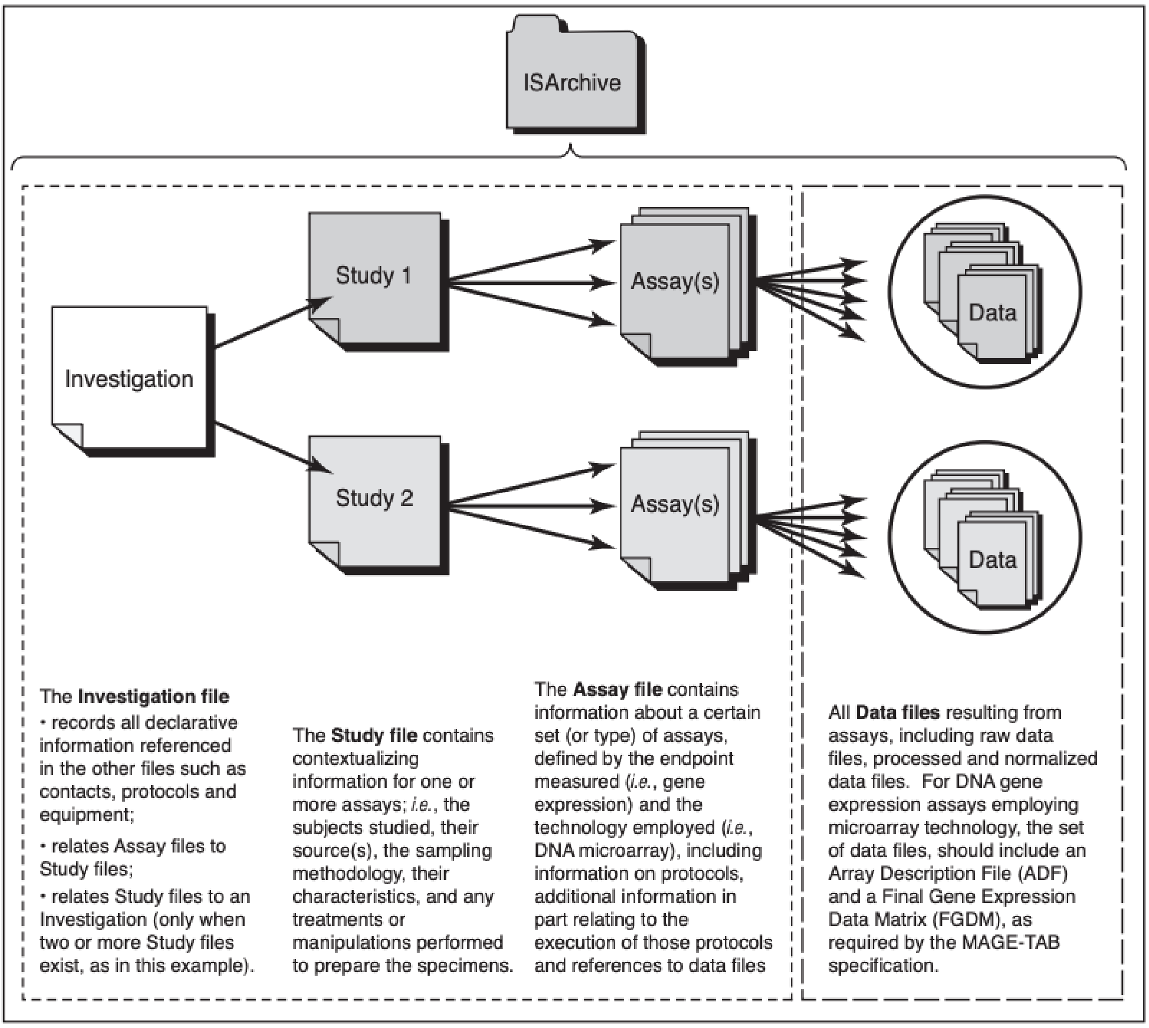

ISArchive
Investigation
Study 1
Study 2
Assay(s)
Assay(s)
Data
Data
The **Investigation file**
• records all declarative information referenced in the other files such as contacts, protocols and equipment;
• relates Assay files to Study files;
• relates Study files to an Investigation (only when two or more Study files exist, as in this example).
The **Study file** contains contextualizing information for one or more assays; *i.e.*, the subjects studied, their source(s), the sampling methodology, their characteristics, and any treatments or manipulations performed to prepare the specimens.
The **Assay file** contains information about a certain set (or type) of assays, defined by the endpoint measured (*i.e.*, gene expression) and the technology employed (*i.e.*, DNA microarray), including information on protocols, additional information in part relating to the execution of those protocols and references to data files
All **Data files** resulting from assays, including raw data files, processed and normalized data files. For DNA gene expression assays employing microarray technology, the set of data files, should include an Array Description File (ADF) and a Final Gene Expression Data Matrix (FGDM), as required by the MAGE-TAB specification.

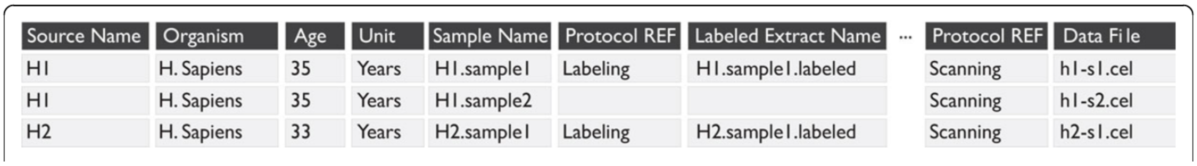
Figure 4: *ISA-TAB file structure. Adapted from [28].*

3. Assay: The regex for assay file name is a_.*.txt

Investigation contains all the information needed to understand the overall goals and means used in an experiment; Study is the central unit, containing information on the subject under study, its characteristics and any treatments applied. Each Study has associated Assay(s), producing qualitative or quantitative data, defined by the type of measurement (i.e. gene expression) and the technology employed (i.e. high-throughput sequencing).[27]

2.6.1 *ISA-TAB file structure example*

As mentioned earlier, ISA-TAB is a tab-delimineted file. This file is in the form of a table and an example is shown in Figure 4.



Source Name	Organism	Age	Unit	Sample Name	Protocol REF	Labeled Extract Name	···	Protocol REF	Data File
H1	H. Sapiens	35	Years	H1.sample1	Labeling	H1.sample1.labeled		Scanning	h1-s1.cel
H1	H. Sapiens	35	Years	H1.sample2				Scanning	h1-s2.cel
H2	H. Sapiens	33	Years	H2.sample1	Labeling	H2.sample1.labeled		Scanning	h2-s1.cel

Figure 5: *ISA-TAB file example. Adapted from [14].*

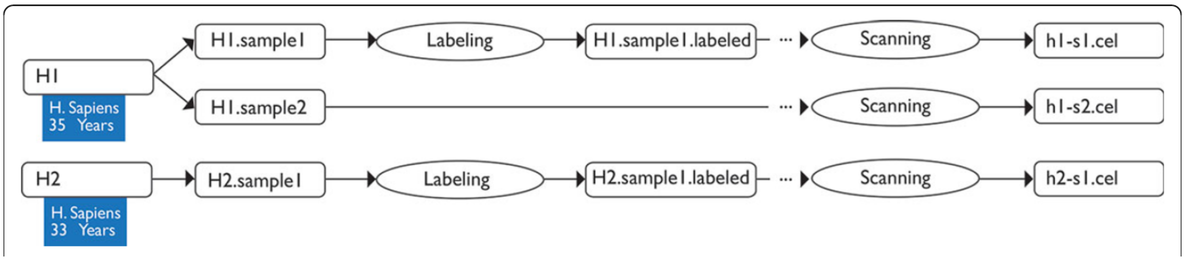


Figure 6: *Graph representation of an ISA-TAB file. Adapted from [14].*

2.6.2 *Equivalent Graph representation of ISA-TAB*

The underlying model of the ISA-Tab format is a direct acyclic graph with nodes representing material entities or data, and edges representing transformation between nodes as shown in Figure 5.[14]

2.6.3 *RDF Representation of ISA-TAB*

The linkedISA conversion exploits knowledge about the ISA-Tab specification to obtain a semantically-rich interpretation of ISA-Tab into RDF (see Figure 5). When available, the transformation exploits the ontological annotations present in ISA-Tab.[14] An example can be seen in Figure 7.

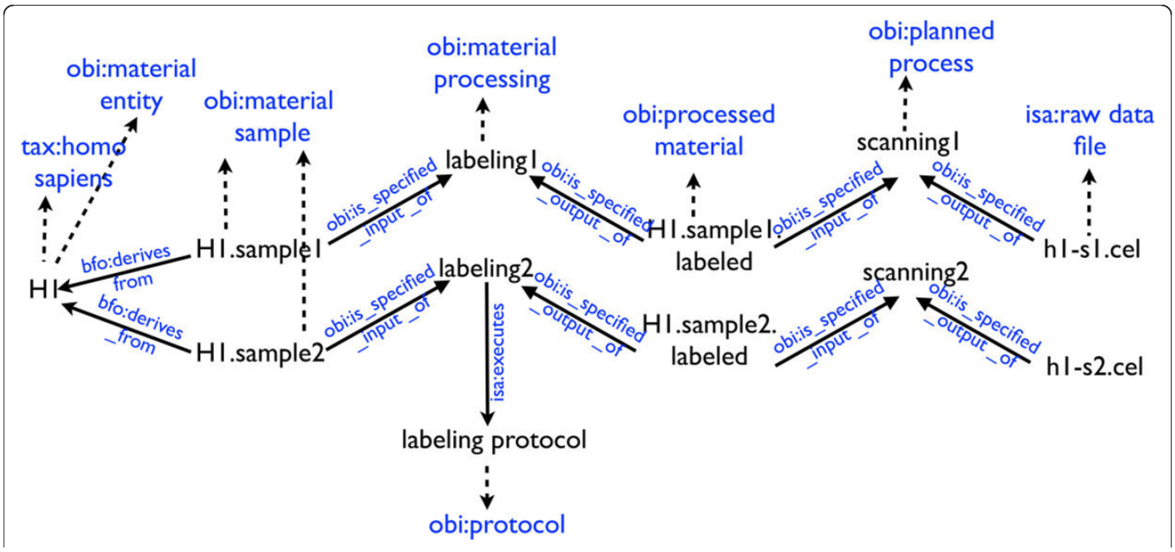


Figure 7: RDF representation of an ISA-TAB file. Adapted from [14].

3

ISA-TAB ENRICHMENT ANALYSIS

There exists software packages and at least one ontology related to converting ISA-TAB data to RDF. But we found the ontology and the tools, both to be less helpful and the reasons behind them are mentioned along with the related work in this chapter.

3.1 RELATED WORK

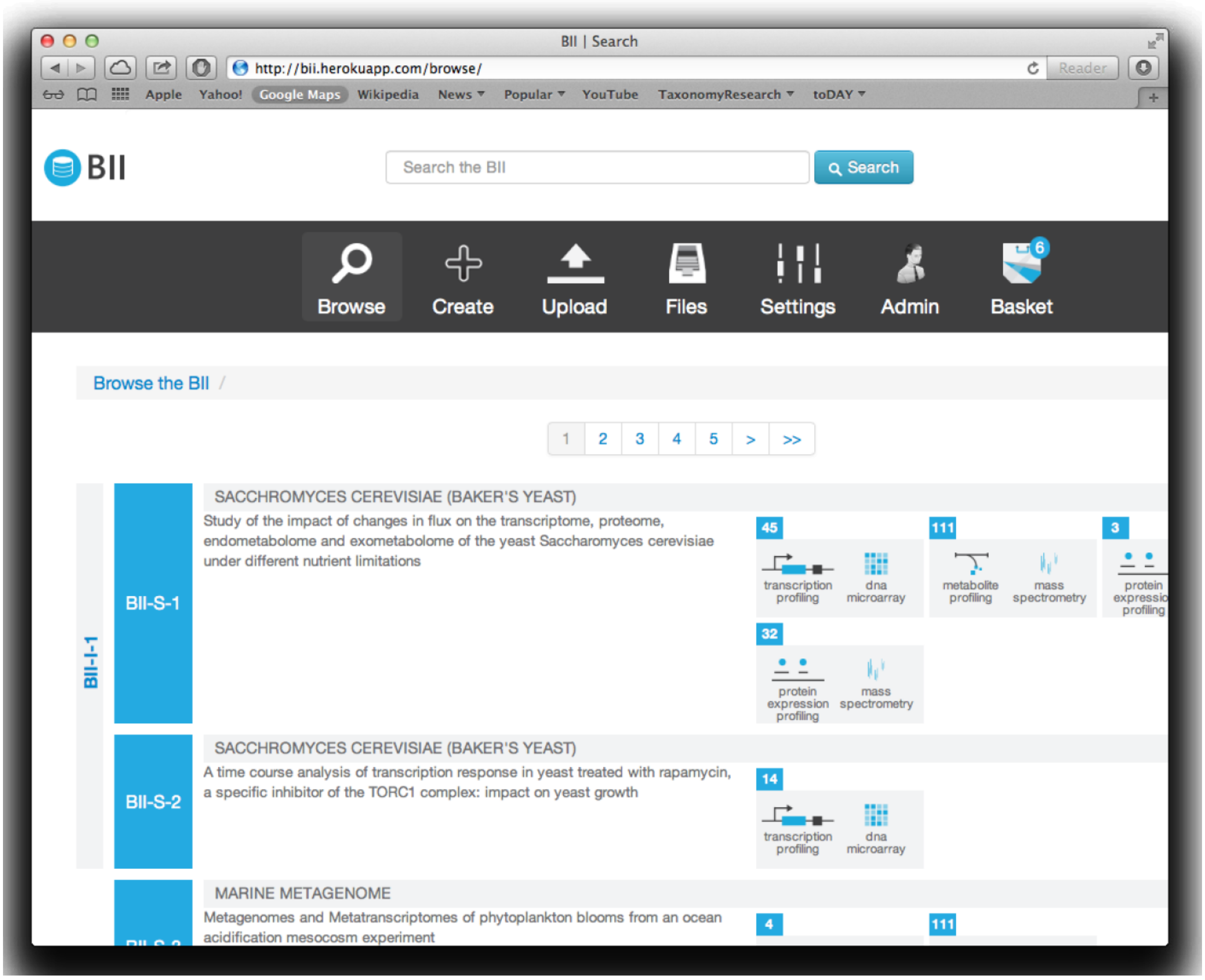
3.1.1 *ISA Tools*

ISA tools are a set of multiple tools that helps in converting ISA-TAB data to RDF. This is the first open source software-suite in this kind.[6] The data persistence layer in these set of tools is called the Bio Investigation Index[7]. These components have been successfully used in systems such as the Stem Cell Discovery Engine (Ho Sui et al., 2012). This tool takes experimental metadata, takes ontologies from the community, and uses these information to help in publishing to international public repositories.

However, Bio Investigation Index is 'read only' and does not exploit any semantic features, nor does it allow 'slicing and dicing' across datasets.[20] Apart from that it is not under active developement. The last commit was made in 2013. Inactive developement poses a bigger risk of security issues and this has been acknowledged by the developers of this tool themselves.[7]

3.1.2 *Bio Graphin system*

Bio-GraphIIn is the new generation of the BII and is designed to extend BII's functionality by allowing CRUD operations, providing semantically rich queries across experiments, among other things.[20] This is a web application and it provides a REST[8] endpoint to convert to RDF. It persists the data in a Graph database and uses SPARQL for information retrieval[23], using Tinkerpop[9] for GUI access.



SACCHROMYCES CEREVISIAE (BAKER'S YEAST)

Study of the impact of changes in flux on the transcriptome, proteome, endometabolome and exometabolome of the yeast Saccharomyces cerevisiae under different nutrient limitations

SACCHROMYCES CEREVISIAE (BAKER'S YEAST)

A time course analysis of transcription response in yeast treated with rapamycin, a specific inhibitor of the TORC1 complex: impact on yeast growth

MARINE METAGENOME

Metagenomes and Metatranscriptomes of phytoplankton blooms from an ocean acidification mesocosm experiment

Figure 8: BioGraphin Tool. Adapted from [20]

However not much information has been provided in the published paper[20] about the software architecture. The code could not be found also in the github page of one of the authors which contain the talk on this tool.[10] Lastly, the heroku[11] link mentioned in the research paper also does seem to exist anymore. The screenshot of the app is provided in Figure 8.

3.1.3 *SEEK*

The SEEK is a suite of tools to support the management, sharing and exploration of data and models in systems biology. The SEEK platform provides an access-controlled, web-based environment for scientists to share and exchange data and models for day-to-day collaboration and for public dissemination.[15]

However the Software Architecture as shown in Figure 9 is not described in detail, especially pertaining to RDF and the search functionality of the same.

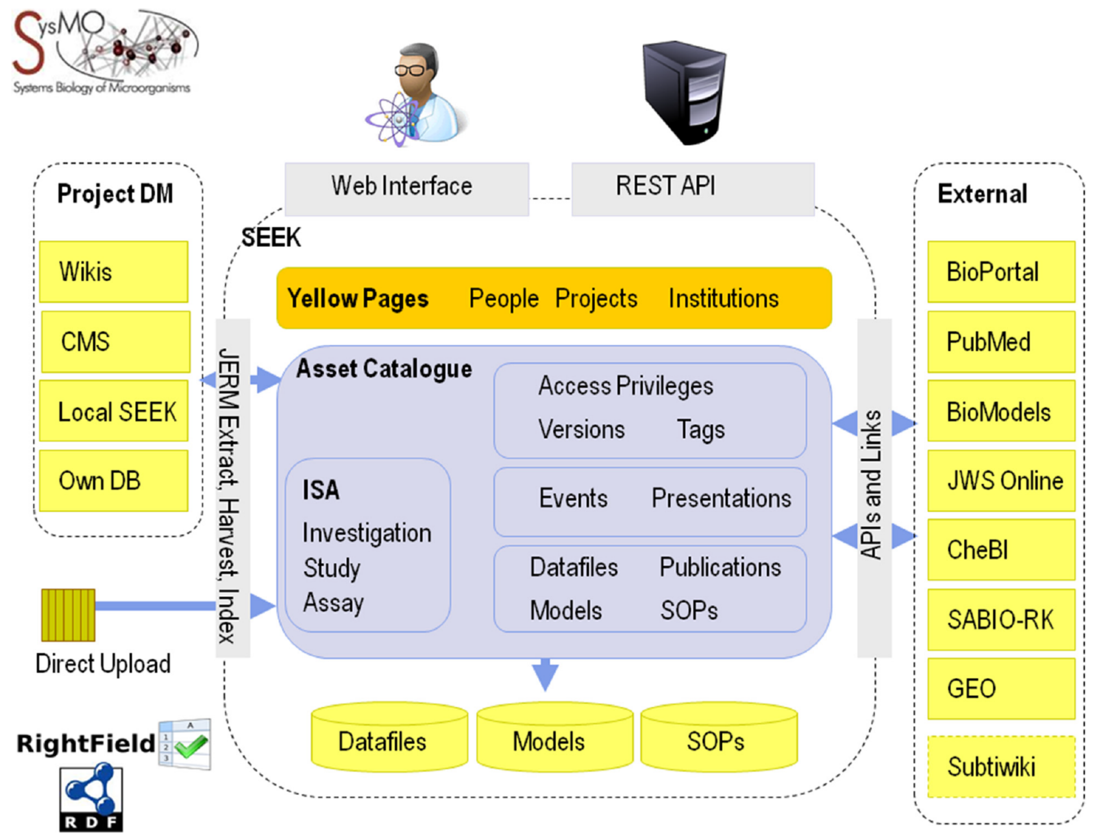


Fig. 1 A diagram of the SEEK components. The SEEK is an Assets Catalogue and repository, which links a number of external tools and services and provides a unified, structured interface to all SEEK assets by linking Assays, Studies and Investigations (the ISA Infrastructure)

Figure 9: SEEK Software Architecture. Adapted from [15]

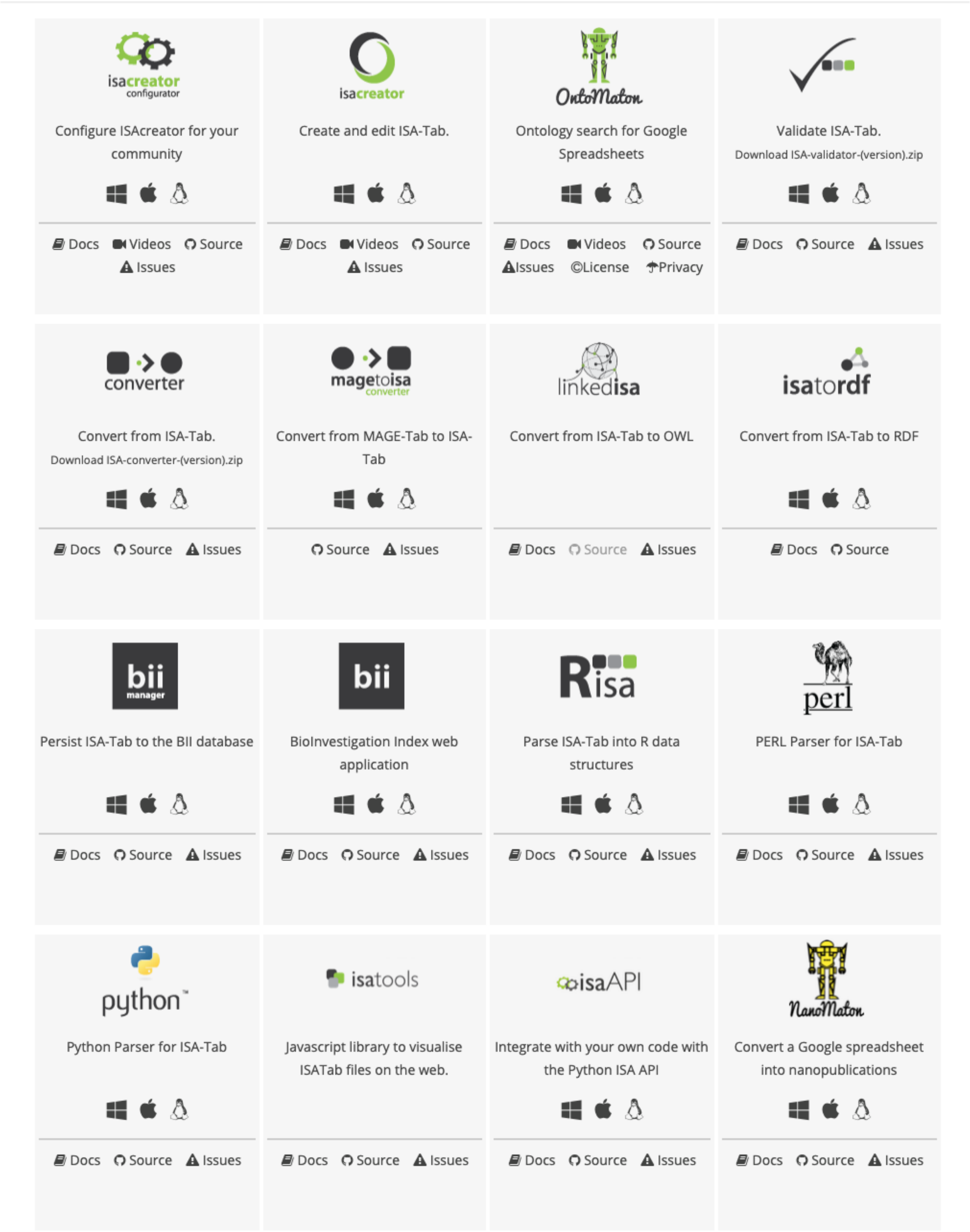
3.1.4 *Ontology Webform*

In comparison to the above mentioned models, the Ontology Webform tries to achieve the following goals:

1. Provide an easy to use tool to create RDF triples from data. This is particularly helpful in comparison to ISA-TOOLS with it's complex suite of many tools as shown in Figure 10.

2. Create an intuitive tool where existing data can be seen and used from readily.

3. Create a generic tool independent of an Ontology which parses the ontology on the fly, connects to a RDF TripleStore and provide suggestions based on retrieved data.



isacreator configurator

Configure ISAcreator for your community

Docs Videos Source Issues

isacreator

Create and edit ISA-Tab.

Docs Videos Source Issues

OntoMaton

Ontology search for Google Spreadsheets

Docs Videos Source Issues ©License Privacy

Validate ISA-Tab.
Download ISA-validator-(version).zip

Docs Source Issues

converter

Convert from ISA-Tab.
Download ISA-converter-(version).zip

Docs Source Issues

magetoisa converter

Convert from MAGE-Tab to ISA-Tab

Source Issues

linkedisa

Convert from ISA-Tab to OWL

Docs Source Issues

isatordf

Convert from ISA-Tab to RDF

Docs Source

bii manager

Persist ISA-Tab to the BII database

Docs Source Issues

bii

BioInvestigation Index web application

Docs Source Issues

Risa

Parse ISA-Tab into R data structures

Docs Source Issues

perl

PERL Parser for ISA-Tab

Docs Source Issues

python™

Python Parser for ISA-Tab

Docs Source Issues

isatools

Javascript library to visualise ISATab files on the web.

Docs Source Issues

isaAPI

Integrate with your own code with the Python ISA API

Docs Source Issues

NanoMaton

Convert a Google spreadsheet into nanopublications

Docs Source Issues

Figure 10: *ISA Tools Software Suite. Adapted from [21]*

4

ISA TAB ONTOLOGY

This chapter discuss the ontology that we created based on the ISA TAB format. This ontology is used to create and present the webform to the users for their inputs and for their reference. The ontology can be found in the github repository.[1]

4.1 OVERVIEW

The entities in the ontology are grouped into three sections for ease of use. Those sections are:

1. Core: It consists of the core entities that make up the ISA TAB i.e. Investigation, Study and Assay. A visualisation of the core entities is shown in Figure 11.
2. ISA details: Consisting of entities specific to ISA TAB like Study design, protocol, and so on.
3. General details: Consisting of general details like Person, Organisation, and so on.

The details about the entities are put into two buckets, namely, Object property and Data property. The difference between them is that in Object property we refer to another entity in the ontology whereas in Data property we use literal values which do not have meaning of it's own. A visualisation of object property is shown in Figure 12 and a visualisation for data property is shown in Figure 13.

4.2 ORDERING

To order the entities such that it matches with the display order shown generally in the ISA-TAB files mentioned in the background chapter, we used annotations. We created a new annotation named 'view_position' which we used in both entities and properties. In the code, we sort the respective items according to

[1] https://github.com/2kunal6/OntologyWebform/blob/master/isa_tab_ontology.owl

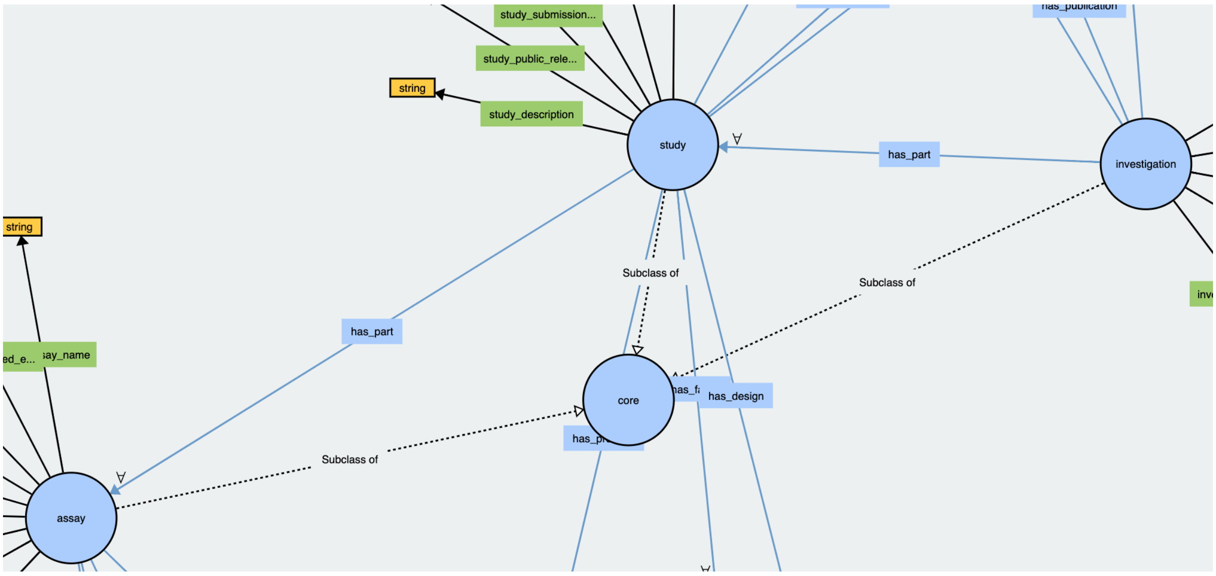


Figure 11: Core Entities

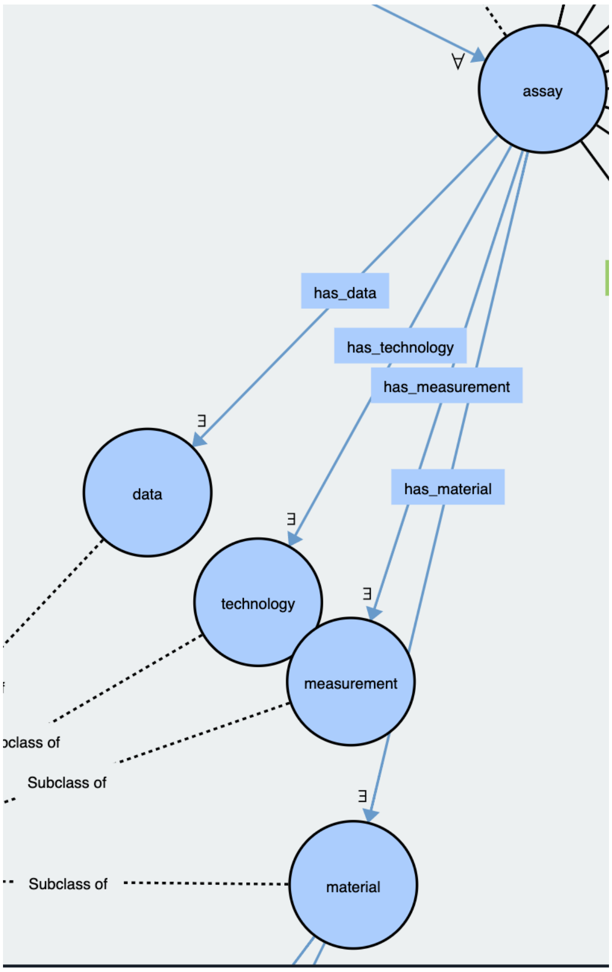


Figure 12: Object Properties

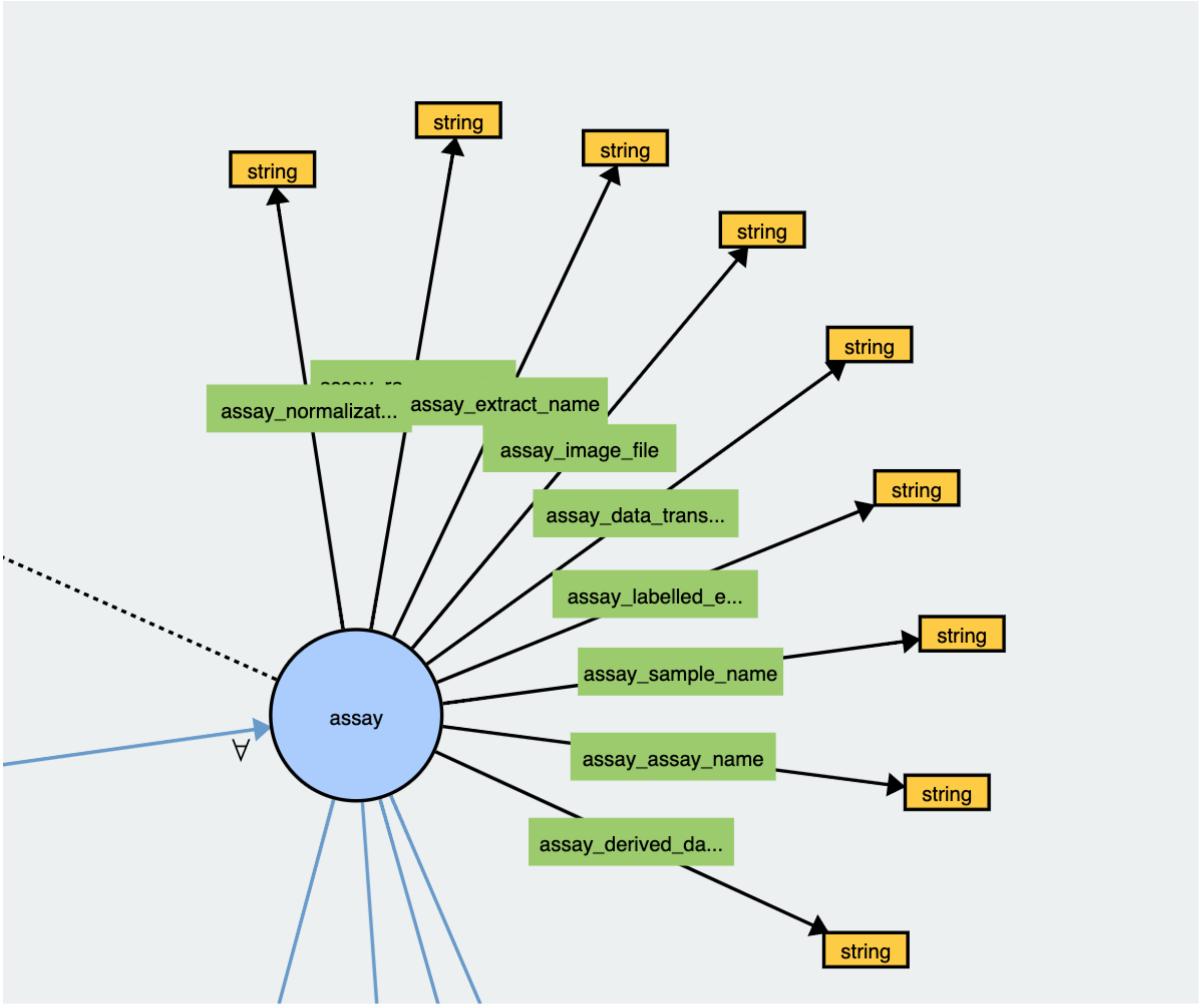
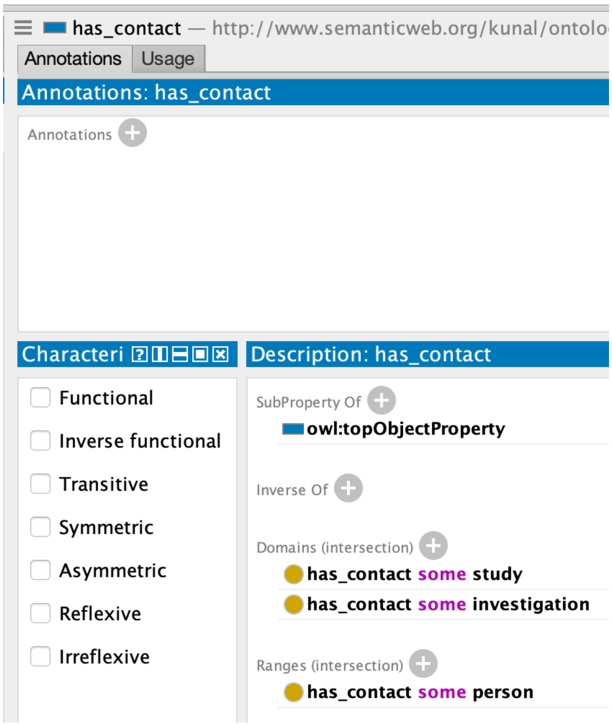


Figure 13: Data Properties



☐ Functional
☐ Inverse functional
☐ Transitive
☐ Symmetric
☐ Asymmetric
☐ Reflexive
☐ Irreflexive

SubProperty Of
owl:topObjectProperty

Inverse Of

Domains (intersection)
has_contact some study
has_contact some investigation

Ranges (intersection)
has_contact some person

Figure 14: Relationships

their values in the 'view_position' annotation, and display them in this same sorted order.

4.3 RELATIONSHIPS

The Domain and Range values of "Object Property" is used to define the relationships. For example, both the entities investigation and study can only have a person as a contact. So to achieve this, an object property named 'has_contact' was created having domain study and investigation, and since the range can only be a 'person', so the range is defined accordingly. An image from Protege for this is shown in Figure 14.

5 ONTOLOGY WEBFORM

This chapter deals with discussing the tool that we developed, namely Ontology Webform. Ontology Webform is a simple web application using which users (especially domain experts) can use to enrich their data. This can also be used by domain experts to create at least some portion of the Knowledge Graph from their data directly, without having to go via a technical person knowledgable in Knowledge Graphs, thus saving time. The platform is open source and the source code can be found here.[1]

5.1 IDEA

Ontology Webform tool is built upon a simple idea of a few webpages and few validations to begin with. The first page lists all the classes where users can define individuals corresponding to classes. The next page is the relationship page where the classes and their attributes list are displayed. The users can then fill out the attribute values corresponding to the attributes in a text field. Validations are also done in this page to help users avoid mistakes related to cardinality, someValuesFrom, allValuesFrom, and so on.

5.2 SOFTWARE ARCHITECTURE

The Software Architecture diagram is shown in Figure 15.
The Software Architecture is designed as follows:

1. This is an web application and the users can access it from a web browser.
2. Java[12] is used to work easily with Apache Jena, a nice open-source tool to work with RDF graphs.
3. The web application connects to Java Servlet over HTTP request-response architecture using javax-servlet.
4. Fuseki server is used as the triplestore to store RDF triples.

[1] https://github.com/2kunal6/OntologyWebform

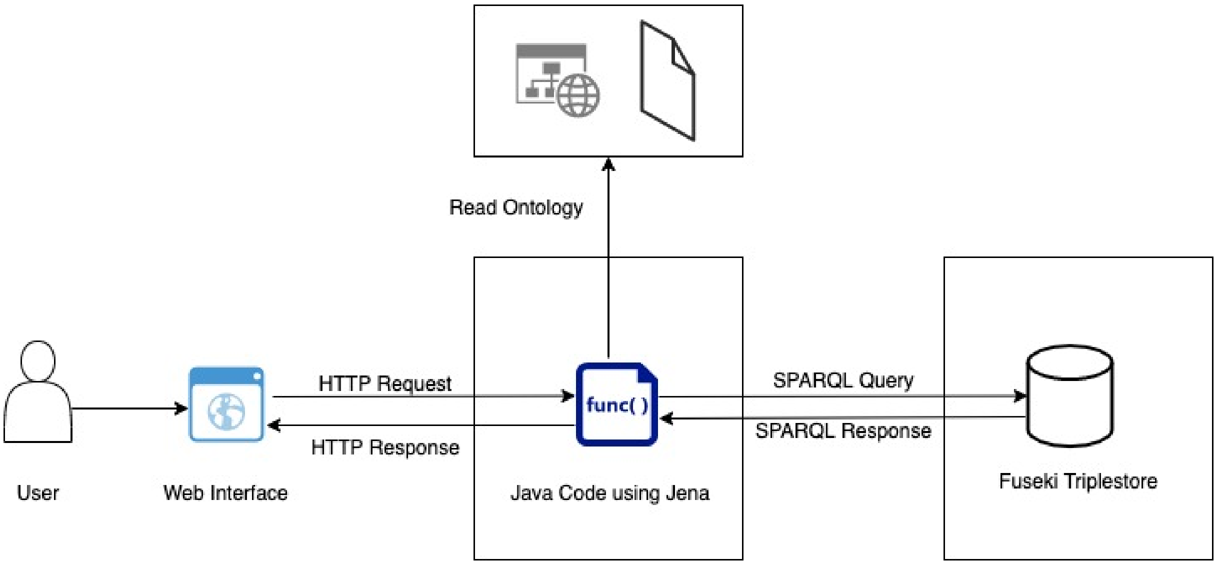


Figure 15: Ontology Webform Software Architecture

5. The Java code connects to Fuseki server using Apache Jena Framework, to store data as RDF triples.

6. The Ontology can be provided by the user either as a file or as an URL. Please note that if URL is provided then the raw content should be provided instead of the actual HTML page (eg. https://raw.githubusercontent.com/ISA-tools/linkedISA-ontologies/master/isaterms.owl instead of https://github.com/ISA-tools/linkedISA-ontologies/blob/master/isaterms.owl). Also please note that the file formats supported right now are RDF/XML, n3 and TTL. However more formats like JSON-LD can be supported easily with minor extensions.

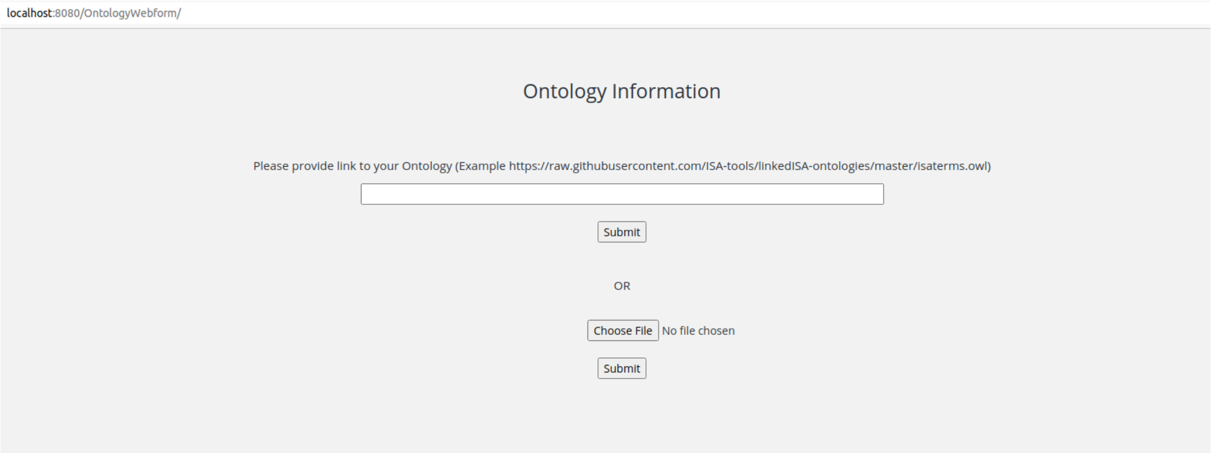
5.3 USE CASES

Ontology Webform can be used to do the following things:

1. The first page (shown in Figure 16) is used to upload the Ontology. The Ontology can be provided either as a file (supported formats are mentioned in the above section) or the raw URL can be provided.

2. The next page (shown in Figure 17) shows all the classes after the Ontology is parsed. Users can use this page to see what other individuals have been created for those classes by other users and also use this page to provide his/her own individuals.

3. The final page (shown in Figure 18) is the relationship page where the actual RDF triples concerning the attributes of individuals are created. Similar

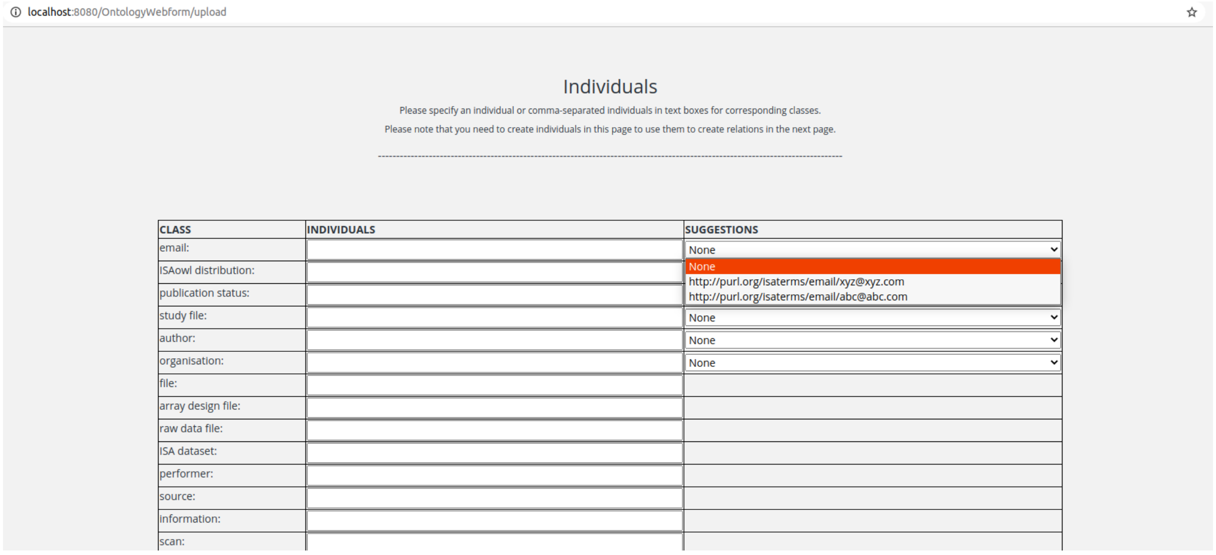


localhost:8080/OntologyWebform/

Ontology Information

Please provide link to your Ontology (Example https://raw.githubusercontent.com/ISA-tools/linkedISA-ontologies/master/isaterms.owl)

Figure 16: Ontology Webform Upload Page



localhost:8080/OntologyWebform/upload

Individuals

Please specify an individual or comma-separated individuals in text boxes for corresponding classes.

Please note that you need to create individuals in this page to use them to create relations in the next page.

CLASS	INDIVIDUALS	SUGGESTIONS
email:		None None http://purl.org/isaterms/email/xyz@xyz.com http://purl.org/isaterms/email/abc@abc.com
ISAowl distribution:		
publication status:		
study file:		None
author:		None
organisation:		None
file:		
array design file:		
raw data file:		
ISA dataset:		
performer:		
source:		
information:		
scan:		

Figure 17: Ontology Webform Individuals Page

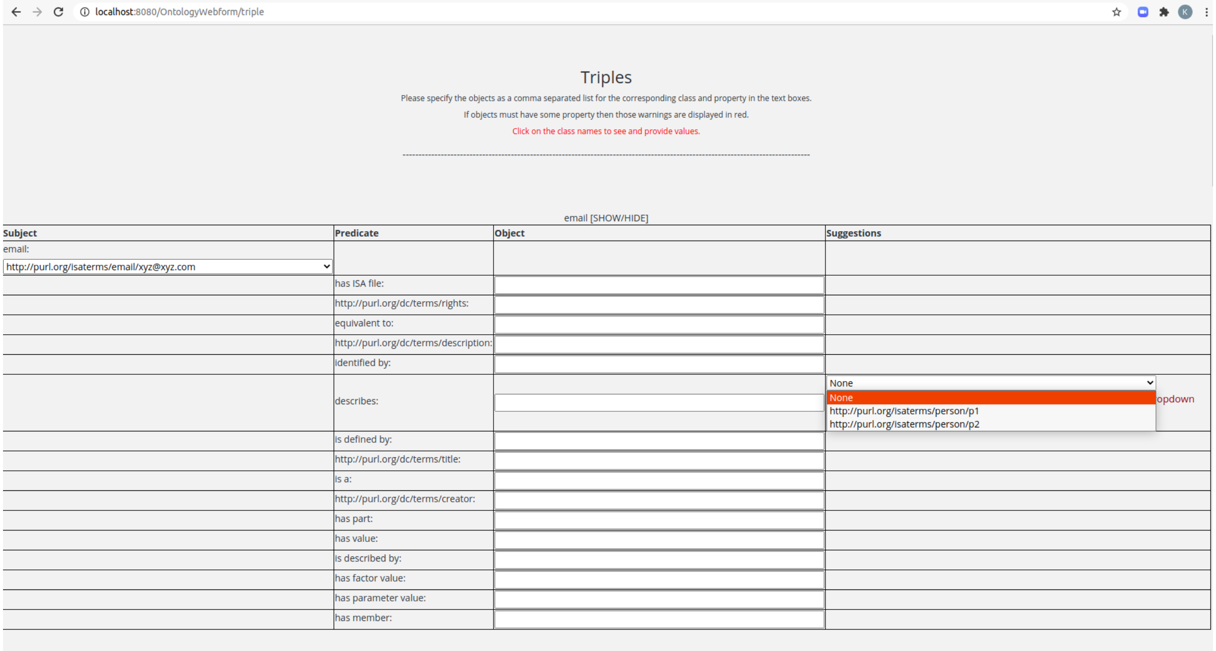
to above, this page also shown the existing relationships created by other users.

4. Apart from the above the tool can also be used to validate few things like cardinality, qualified cardinality, and so on.

5.4 ISA TAB ONTOLOGY

Since we did not find any ontology that fits our purpose, we decided to create an ontology of our own. This ontology is provided in the codebase.[2] A

[2] https://github.com/2kunal6/OntologyWebform



Triples

Please specify the objects as a comma separated list for the corresponding class and property in the text boxes.

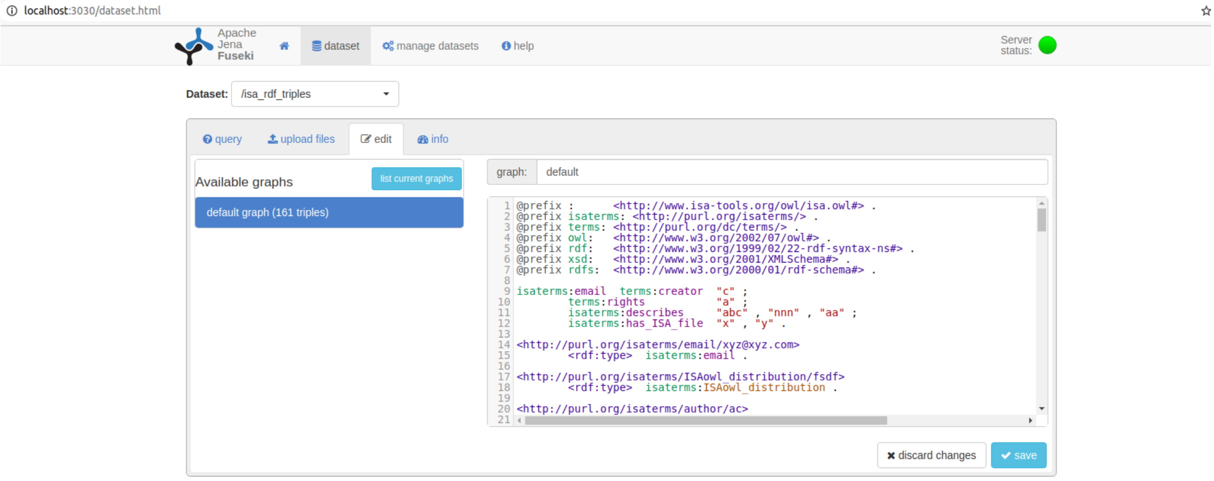
If objects must have some property then those warnings are displayed in red.

Click on the class names to see and provide values.

email [SHOW/HIDE]

Subject	Predicate	Object	Suggestions
email: http://purl.org/isaterms/email/xyz@xyz.com			
	has ISA file:		
	http://purl.org/dc/terms/rights:		
	equivalent to:		
	http://purl.org/dc/terms/description:		
	identified by:		
	describes:		None None http://purl.org/isaterms/person/p1 http://purl.org/isaterms/person/p2 opdown
	is defined by:		
	http://purl.org/dc/terms/title:		
	is a:		
	http://purl.org/dc/terms/creator:		
	has part:		
	has value:		
	is described by:		
	has factor value:		
	has parameter value:		
	has member:		

Figure 18: *Ontology Webform Triples Page*



Dataset: /isa_rdf_triples

Available graphs

default graph (161 triples)

graph: default

```
@prefix :      <http://www.isa-tools.org/owl/isa.owl#> .
@prefix isaterms: <http://purl.org/isaterms/> .
@prefix terms: <http://purl.org/dc/terms/> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .

isaterms:email  terms:creator  "c" ;
        terms:rights           "a" ;
        isaterms:describes     "abc" , "nnn" , "aa" ;
        isaterms:has_ISA_file  "x" , "y" .

<http://purl.org/isaterms/email/xyz@xyz.com>
        <rdf:type>  isaterms:email .

<http://purl.org/isaterms/ISAowl_distribution/fsdf>
        <rdf:type>  isaterms:ISAowl_distribution .

<http://purl.org/isaterms/author/ac>
```

Figure 19: *Ontology Webform Fuseki Server*

visualisation is also provided for the same using VOWL[31], which is a tool used to visualise ontologies.

5.5 enrichment

One of the important things we wanted to achieve is to help domain experts enrich their data using help from data created earlier by other users / researchers. To achieve this we viewed the Ontology from two perspectives described below.

5.5.1 *Using Restrictions*

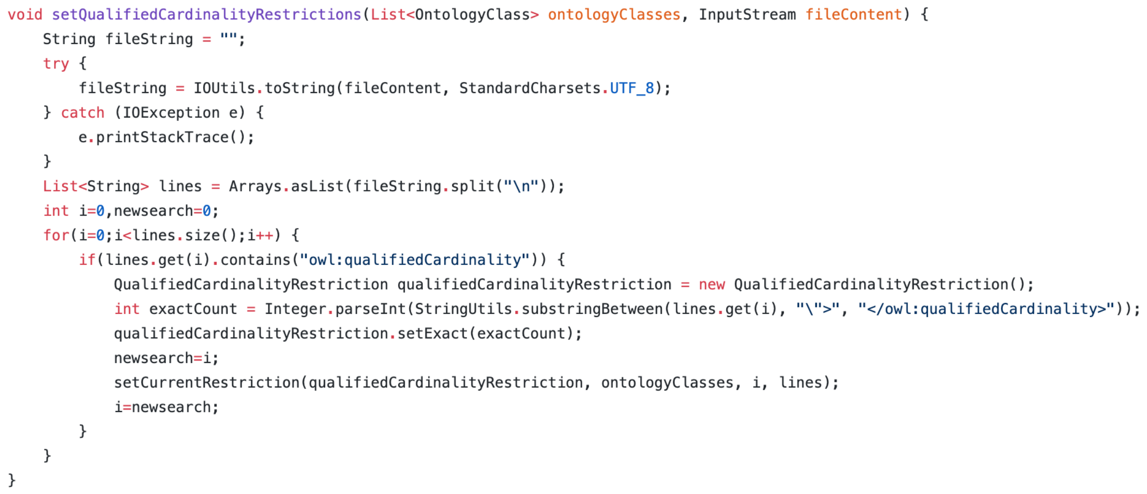
Restrictions help in identifying the values corresponding to a property that a class accepts. For example, in case of allValuesFrom restriction, we know that all the values for a property of a class must come from some particular class's individuals. Thus we can show them as suggestions and users can select from these suggested values. Similarly, in case of someValuesFrom restriction, we know that some values must belong to a particular class's individuals. We again show them as suggestions for users to choose from. In case of qualifiedCardinality, we can similarly use this granular information for suggestions.

5.5.2 *Using Domain and Range Information*

Another area from where we can use some information to help users in enrichment is using Domain and Range values of ObjectProperty. As mentioned in the Background section, an rdfs:range axiom asserts that the values of this property must belong to the class extension of the class description or to data values in the specified data range.[25] Similarly the rdfs:domain extension tells to which entity this ObjectProprty belongs to. Basically the Domain and Range of the ObjectProperties help form the relationship between two entities via this ObjectProperty. We can use this information to show the corresponding individuals similar to above.

5.6 data storage

After getting the values from the text values, the values are checked to see if it begins with the class URI, and if so then it is stored as is. If not then the class URI is prepended with the value from the text field to make it unique. For example, for http://purl.org/isaterms/email class if the value provided in the text field is a@a.com then it is prepended with the class URI and the acutal value stored is http://purl.org/isaterms/email/a@a.com.



```
void setQualifiedCardinalityRestrictions(List<OntologyClass> ontologyClasses, InputStream fileContent) {
    String fileString = "";
    try {
        fileString = IOUtils.toString(fileContent, StandardCharsets.UTF_8);
    } catch (IOException e) {
        e.printStackTrace();
    }
    List<String> lines = Arrays.asList(fileString.split("\n"));
    int i=0,newsearch=0;
    for(i=0;i<lines.size();i++) {
        if(lines.get(i).contains("owl:qualifiedCardinality")) {
            QualifiedCardinalityRestriction qualifiedCardinalityRestriction = new QualifiedCardinalityRestriction();
            int exactCount = Integer.parseInt(StringUtils.substringBetween(lines.get(i), "\">", "</owl:qualifiedCardinality>"));
            qualifiedCardinalityRestriction.setExact(exactCount);
            newsearch=i;
            setCurrentRestriction(qualifiedCardinalityRestriction, ontologyClasses, i, lines);
            i=newsearch;
        }
    }
}
```

Figure 20: *Qualified Cardinality Restriction Implementation*

For now, all the data is stored in the default graph. If data grows then we can have different graphs to store different kind of data to handle the size.

5.7 VALIDATIONS

The allValuesFrom, someValuesFrom and the cardinality restrictions were straightforward to implement once we had all the individual values pre-populated in the cache of the corresponding classes needing them. The pre-population of the cache is done in the OntologyProcessor class and the validation is done in the TripleValidator class of the source code.[3]

5.7.1 *qualifiedCardinality*

Implementing the qualifiedCardinality restriction was not as straightforward as the above mentioned ones because this functionality is not implemented yet (as on 07.01.2021) in the Apache Jena library that we are using.[22]

Additional code had to be written to handle the QualifiedCardinality feature. The code for the same is shown in Figure 20.

This implementation as of now is only for ontologies in owl format and makes use of the owl file structure. It parses the whole file line by line and on getting the keyword "owl:qualifiedCardinality" it gets the integer and traverses back to find the class which uses this qualifiedCardinality and the class on which it is applied.

[3] https://github.com/2kunal6/OntologyWebform

5.8 PERFORMANCE

To optimize on performance point of view everything is kept in cache so that minimum calls to Fuseki is needed. Now, currently we are dealing with small data size and thus it is currently possible to use the RAM as cache but when the data grows beyond the RAM's capabilities then we might have to use some distributed external in-memory cache like REDIS[4] or MEMCACHED[5], such that the performance is still not sacrificed.

5.9 TECHNOLOGIES

The Technologies used and the arguments for their choices are described in the following subsections.

5.9.1 *Apache Jena*

As described above, it is an open-source technology to read and work with ontologies.

5.9.2 *Java with Tomcat*

Since Apache Jena works nicely with Java[22], we used Java to write the entire application to make full use of Apache Jena. And for the web application part, we used another open-source technology called Apache Tomcat, which also works nicely with Java.

5.9.3 *Apache Jena Fuseki*

Apache Jena Fuseki[13], another open-source tool is used for storing the triples because it is simple to deploy and use. It also provides an UI for easy access.

[4] https://redis.io

[5] https://memcached.org/

6

CONCLUSION AND FUTURE WORK

In this thesis work we started with exploring the following research questions:

1. Does an Ontology exist for ISA-TAB?

2. Does a tool exists which enriches ISA-TAB data (or any data in general)?

The motivation to build this tool is to help scientists and researchers exchange experimental data by storing it in semantic form as RDF triples. This also helps in better understanding by giving semantic meaning to data.

We could not find the answer to the first question above, so we created an ontology for ISA TAB on our own. We then utilised this ontology to enrich ISA-TAB data. However the converter tools or the data enrichment tools mentioned below were not very helpful:

1. ISA Tools: Although it was used successfully in few systems but it is 'read only'.

2. Bio Graphin: The proposed tool couldn't be found in the address mentioned in Figure 8 (screenshot taken from [20])

3. SEEK: Although RDF files were generated for entities, the search was not obvious, if it exists at all.

Therefore we started off with building a simple web application for this purpose using a very simple idea. After an user uploads an ontology to our tool, we parse it dynamically. Using this parsed information we show two pages. The first page is the "Individual" page wherein users can see all the classes of the ontology and create individuals in the corresponding text fields. Users can also see corresponding individuals created by other users and get ideas from it. These suggestions are derived using Restriction and Range information. We then store these individuals in Fuseki Triplestore. The next page is the Relationships page where RDF triples concerning relations are created. The individuals created in the previous page, along with all the individuals created previously by other users are suggested to the user who can then choose to create relationships accordingly. These values are again stored in the Fuseki Triplestore.

The tool as of now relies on user to provide data in text fields. However, in future we can try to parse the data files directly using code and populate the text fields accordingly. We might have to use some kind of metadata information about the template of the data to do this. Some knowledge of Natural Language Processing might also help in this process.

Another improvement that can be made to the tool is related to performance. Right now, two calls are made to the Fuseki Server to get the triples, once after the upload action and once after new individuals are created. This can be restricted to one call by using the values from first call and merging the newly created individuals with the existing ones and place them properly based on restrictions. And those new triples can just be stored to Fuseki Server using an asynchronous call. Caution should be taken to avoid data corruption in case the program crashes or if the Fuseki Server goes down.

An improvement can also be made to make the implementation of finding qualifiedCardinality of a class more robust. Additionally it can be integrated to the open source framework Apache Jena.

BIBLIOGRAPHY

[1] URL: https://www.isacommons.org/. (accessed: 01.03.2021).

[2] URL: https://www.w3.org/OWL/. (accessed: 01.03.2021).

[3] URL: https://www.w3.org/RDF/. (accessed: 01.03.2021).

[4] URL: https://www.w3.org/TR/rdf-sparql-query/. (accessed: 01.03.2021).

[5] URL: https://isa-specs.readthedocs.io/en/latest/isamodel.html. (accessed: 01.03.2021).

[6] URL: https://isa-tools.org/. (accessed: 03.03.2021).

[7] URL: https://github.com/ISA-tools/BioInvIndex. (accessed: 03.03.2021).

[8] URL: https://www.redhat.com/en/topics/api/what-is-a-rest-api. (accessed: 04.03.2021).

[9] URL: https://tinkerpop.apache.org/. (accessed: 04.03.2021).

[10] URL: https://agbeltran.github.io/talks/2013-10-16-nettab. (accessed: 04.03.2021).

[11] URL: https://www.heroku.com/platform. (accessed: 04.03.2021).

[12] URL: https://www.java.com/en/. (accessed: 01.03.2021).

[13] URL: https://jena.apache.org/. (accessed: 01.03.2021).

[14] González-Beltrán et al. "linkedISA: semantic representation of ISA-Tab experimental metadata". In: *BMC Bioinformatics* (2014).

[15] Wolstencroft et al. "SEEK: a systems biology data and model management platform". In: *BMC Systems Biology* (2015). DOI: 10.1186/s12918-015-0174-y.

[16] Vassilis Christophides. *Resource Description Framework (RDF) Schema (RDFS)*. Ed. by LING LIU and M. TAMER OZSU. Boston, MA: Springer US, 2009, pp. 2425–2428. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_1319. URL: https://doi.org/10.1007/978-0-387-39940-9_1319.

[17] Jim Hendler Dean Allemang. "Semantic Web for the Working Ontologist". In: *Semantic Web for the Working Ontologist*. sciencedirect, 2011. Chap. 12, pp. 249–278. DOI: 10.1016/B978-0-12-385965-5.10012-3.

[18] *Expressiveness of RDF formats*. URL: https://slidewiki.org/print/90781/_/90781/. (accessed: 24.12.2020).

[19] Javier D. Fernández and Miguel A. Martínez-Prieto. “RDF Serialization and Archival”. In: *Encyclopedia of Big Data Technologies*. Ed. by Sherif Sakr and Albert Zomaya. Cham: Springer International Publishing, 2018, pp. 1–11. ISBN: 978-3-319-63962-8. DOI: 10.1007/978-3-319-63962-8_286-1. URL: https://doi.org/10.1007/978-3-319-63962-8_286-1.

[20] Alejandra Gonzalez-Beltran et al. “Bio-GraphIIn: a graph-based, integrative and semantically-enabled repository for life science experimental data”. In: *EMBnet.journal* 19.B (2013), pp. 46–50. ISSN: 2226-6089. DOI: 10.14806/ej.19.B.728. URL: https://journal.embnet.org/index.php/embnetjournal/article/view/728.

[21] *ISA - TAB*. URL: https://www.dcc.ac.uk/resources/metadata-standards/isa-tab#:~:text=The%20Investigation%2FStudy%2FAssay%20(,employing%20a%20combination%20of%20technologies.. (accessed: 28.11.2020).

[22] *Jena Ontology API*. URL: https://jena.apache.org/documentation/ontology/. (accessed: 07.01.2021).

[23] Chris Kamphuis. “Graph Databases for Information Retrieval”. In: *Advances in Information Retrieval*. Ed. by Joemon M. Jose et al. Cham: Springer International Publishing, 2020, pp. 608–612. ISBN: 978-3-030-45442-5.

[24] *OWL Web Ontology Language Guide*. URL: https://www.w3.org/TR/owl-guide/. (accessed: 02.01.2021).

[25] *OWL Web Ontology Language Reference*. URL: https://www.w3.org/TR/owl-ref/. (accessed: 09.01.2021).

[26] *RDF*. URL: https://slidewiki.org/print/90789/_/90789/. (accessed: 24.12.2020).

[27] Rocca-Serra et al. “ISA software suite: supporting standards-compliant experimental annotation and enabling curation at the community level.” In: *Bioinformatics* (2010), 26(18):2354–2356. DOI: 10.1093/bioinformatics/btq415.

[28] et al Sansone S-A. “The First RSBI (ISA-TAB) Workshop: “Can a Simple Format Work for Complex Studies?”” In: *OMICS A Journal of Integrative Biology* (2008). DOI: 10.1089/omi.2008.0019. URL: https://www.liebertpub.com/doi/pdf/10.1089/omi.2008.0019.

[29] *SEMANTIC WEB*. URL: https://www.w3.org/standards/semanticweb/#:~:text=The%20term%20%E2%80%9CSemantic%20Web%E2%80%9D%20refers,SPARQL%2C%20OWL%2C%20and%20SKOS.. (accessed: 23.12.2020).

[30] *Semantic Web Language Stack*. URL: https://www.w3.org/2009/Talks/0120-campus-party-tbl/#(14). (accessed: 23.12.2020).

[31] Vincent LinkEduard MarbachStefan Negru Steffen Lohmann. “WebVOWL: Web-based Visualization of Ontologies”. In: *Springer, Cham* (2015). DOI: https://doi.org/10.1007/978-3-319-17966-7_21.