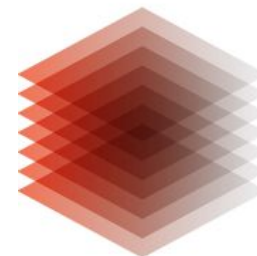




NFDI₄Chem

ENHANCE
YOUR
DATA.



TIB

dct:title

Breakout Session II: Hands on Data Annotation using Ontologies

Creating a prototype knowledge graph from NMR spectroscopy research data

dct:identifier

[DOI:10.5281/zenodo.7050763](https://doi.org/10.5281/zenodo.7050763)

dct:creator

[ORCID:0000-0002-1595-3213](https://orcid.org/0000-0002-1595-3213)

fo:'participates in' 1st Ontologies4Chem Workshop 2022 – Ontologies for chemistry (day 2)

dct:identifier

[DOI:10.25798/frnp-sn04](https://doi.org/10.25798/frnp-sn04)

sdo:duration

2022-09-08T16:50:00+02:00/PT10M

Creating knowledge graphs from research data

the preconditions - research data formats



- research data is stored in multiple file formats
 - domain specific formats
 - vendor specific or open file formats

Format	Data type	Maintainer	Parent Format	Specification
mzML	Mass spectrometry	HUPO/ PSI	XML	open
NMReDATA	NMR	NMReDATA Initiative	SDF	open
JCAMP-DX	multiple	IUPAC	ASCII, Text	open

Creating knowledge graphs from research data

the preconditions - metadata schemata



- there are different metadata schemata
 - e.g. DataCite's schema, schema.org, Dublin Core Terms and repository specific schemata
 - often not domain specific / fine grained enough
 - if domain specific usually repository specific

typeClass:	"compound"
▼ value:	
▼ 0:	
▶ topicClassValue:	{...}
▶ topicClassVocab:	{...}
▼ topicClassVocabURI:	
typeName:	"topicClassVocabURI"
multiple:	false
typeClass:	"primitive"
value:	"https://pubchem.ncbi.nlm.nih.gov/compound/5280805"

▼ tag:	
id:	574047
taggable_type:	"Molecule"
taggable_id:	3366
▼ taggable_data:	
▼ chemotion:	
doi:	"10.14272/QXXCRBSWGPRILJ-UHFFFAOYSA-N.1"
chemotion_first:	"2022-01-10T14:48:05.327+01:00"
last_published_at:	"2022-01-10T14:48:13.202+01:00"
pubchem_cid:	162394348

Creating knowledge graphs from research data

the preconditions - domain specific metadata



- it is hard to query & aggregate data from different sources
 - need to know provider specific metadata schemata
 - or, have a true domain specific standard schema
 - like IUPAC FAIRspec finding aid ?

Creating knowledge graphs from research data

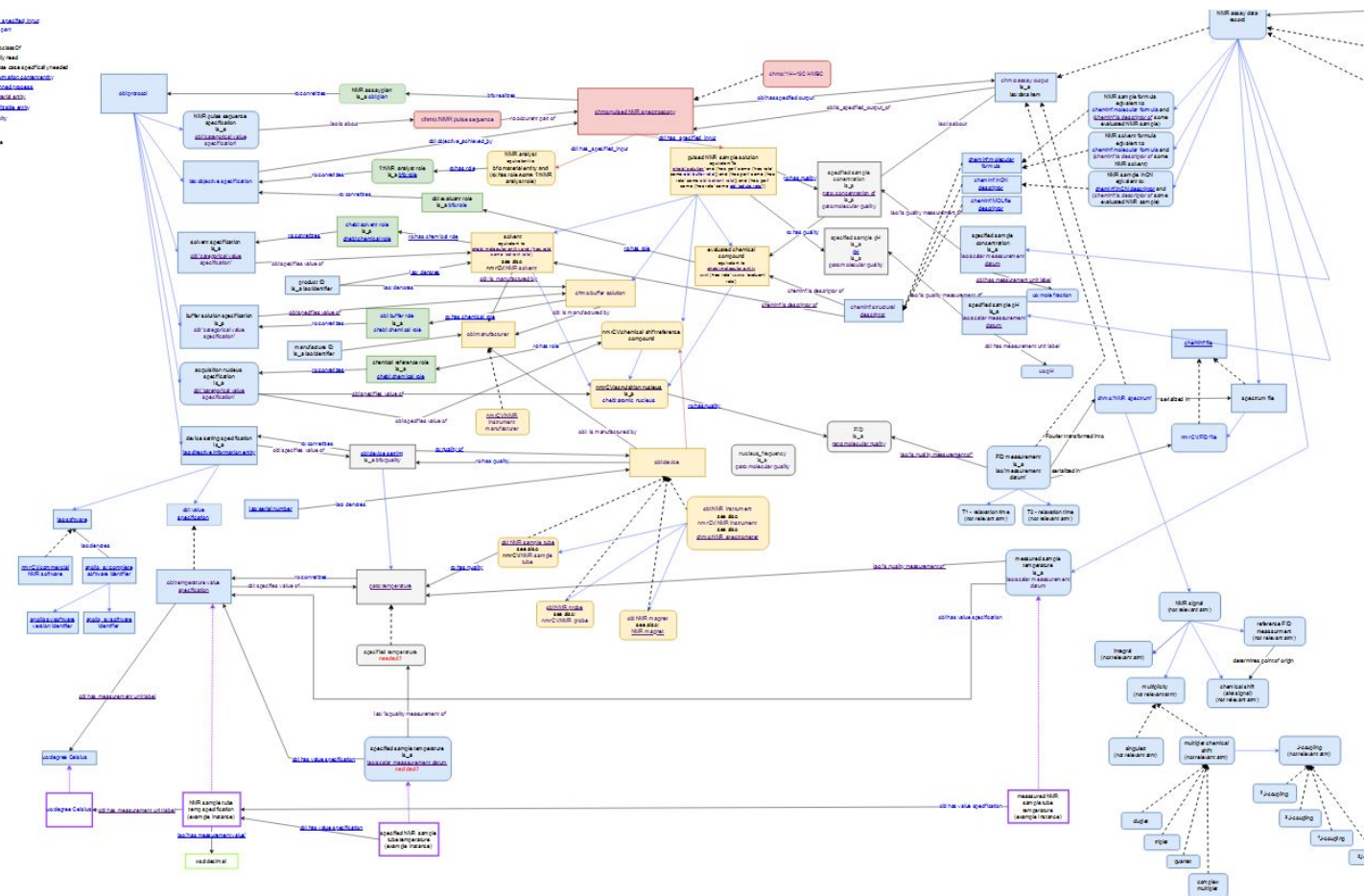
the preconditions - suitable ontologies already exist



- knowledge graphs could be an alternative
 - make federated queries with SPARQL
 - only need to know about the used ontology terms
 - many of which are already commonly used
 - fosters interdisciplinary work
- proof of concept
 - make knowledge graphs from existing datasets
 - make federated queries on them with SPARQL

Creating knowledge graphs from research data

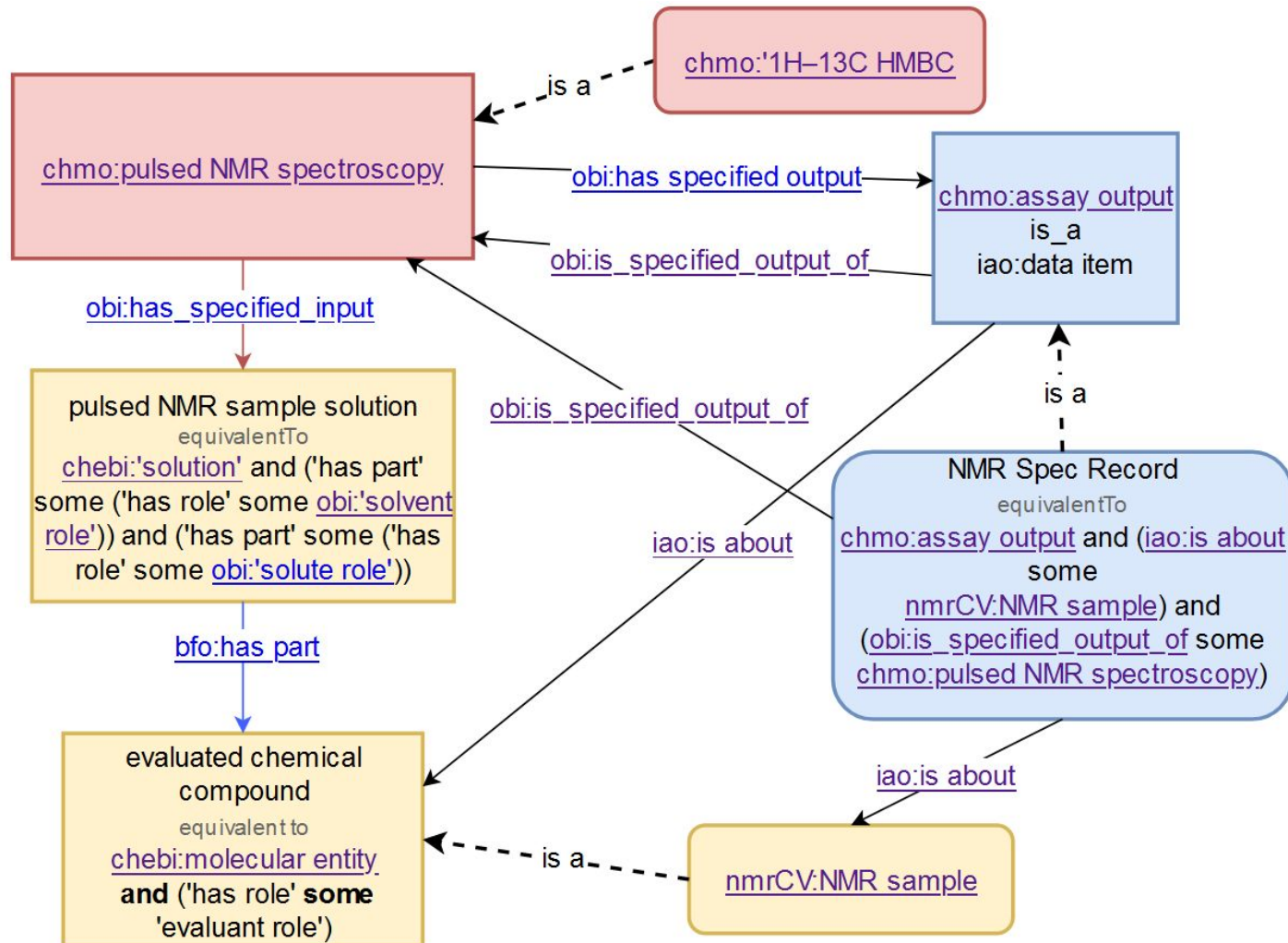
Step 1: define a terminology box (TBox)



- finding the right terms to describe NMR spectroscopy
- reusing as many terms as possible for interoperability
- we found most in:
 - CHMO
 - nmrCV
 - CHEBI
 - CHEMINF
 - OBI
 - IAO

Creating knowledge graphs from research data

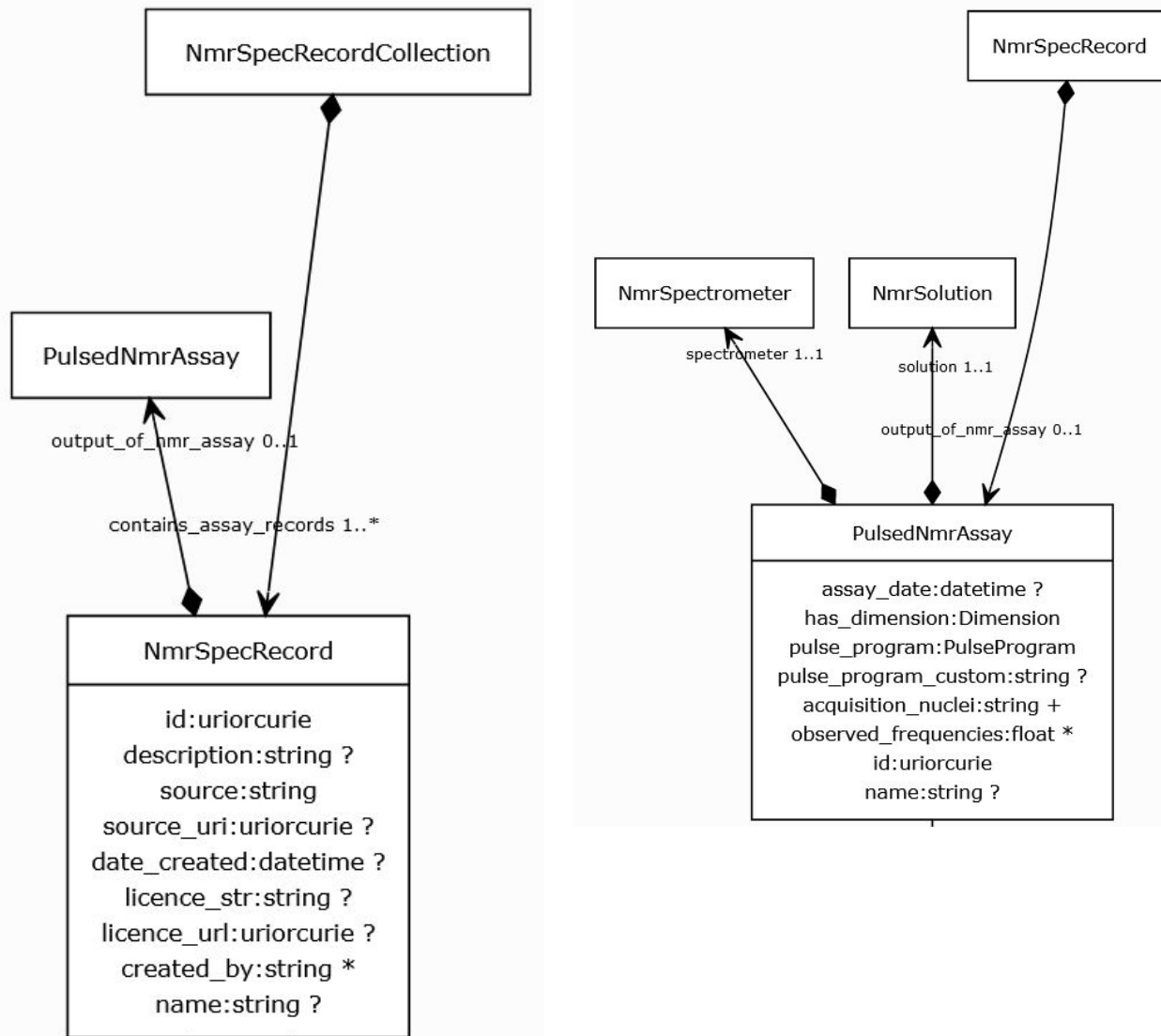
Step 1: define a terminology box (TBox)




- need to define new entities
 - in existing ontologies
 - one of our later TODOs
 - involves discussion with domain experts
 - semantically more precise

Creating knowledge graphs from research data

Step 2: define the shape of the knowledge graph



- <https://stroemphi.github.io/NMRspec/>
- using <https://linkml.io> framework 
- prototypical → not complete
- focus on pulsed NMR spectroscopy
 - only links to broader assay context

Creating knowledge graphs from research data

Step 2: define the shape of the knowledge graph



```
NMRspec.yaml
40 - semweb_context
41 imports:
42 - linkml:types
43 - ./Provenance
44 emit_prefixes:
45   nmrSPARQL
46 #####
47 classes:
48 # Mixins
49   ChemicalDescriptor:
82   NamedThing:
89
90 # processes
91 PulsedNmrAssay:
92   mixins:
93     - NamedThing
94   class_uri: chmo:0000613
95   slot_usage:
96     id:
97       ifabsent: uri(nmrSPARQL:Assay)
98   attributes:
99     assay_date:
105    solution:
112    spectrometer:
113      slot_uri: obi:0000293
114      inlined: true
115      required: true
116      domain: PulsedNmrAssay
117      range: NmrSpectrometer
118   has_dimension:
119     required: true
120     ifabsent: string(10)
121     range: Dimension
122   pulse_program:
123     description: The pulse program of a
      PulsedNmrAssay is a required property and
      must be conform to one of the values
      defined in the PulseProgram enum class.
```

The diagram illustrates the workflow for defining the shape of a knowledge graph. It shows three main components: the NMRspec.yaml file, the NMRspec web interface, and the NMRspec.py Python file.

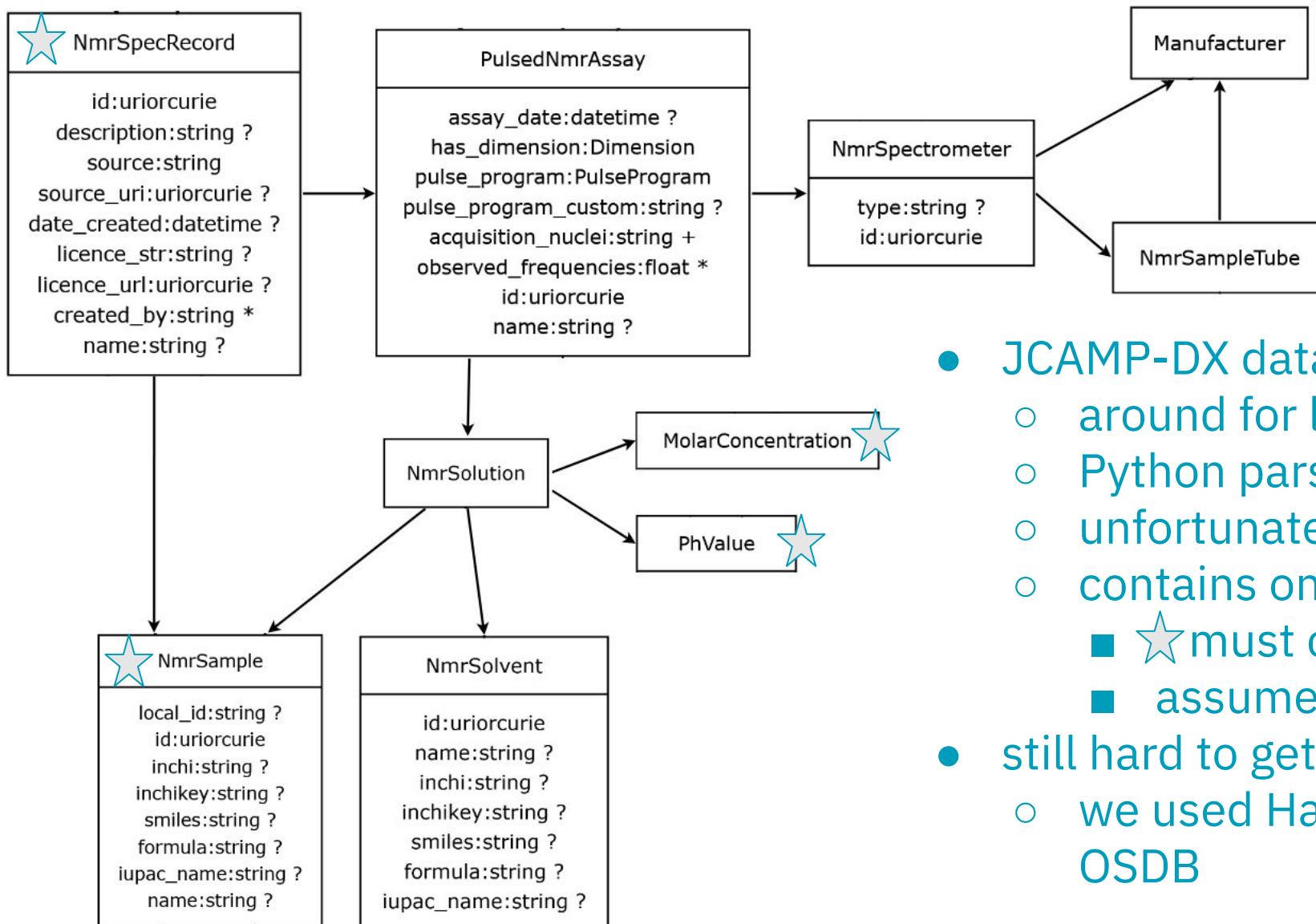
NMRspec.yaml: This file defines the classes and mixins for the knowledge graph. It includes a class `PulsedNmrAssay` which inherits from `NamedThing` and has attributes like `assay_date`, `solution`, `spectrometer`, `has_dimension`, and `pulse_program`.

NMRspec web interface: This interface provides a user-friendly way to interact with the knowledge graph. It shows the `NMRspec` model version (1.7.0) and version (0.0.1). It lists the classes defined in the model, including `MOLfile`, `Manufacturer`, `MolarConcentration`, `MolarityMeasurementDatum`, `NmrBuffer`, `NmrSample`, `NmrSampleTube`, `NmrSolution`, and `NmrSolvent`.

NMRspec.py: This Python file defines the `PulsedNmrAssay` class using the `@dataclass` decorator. It includes attributes like `class_uri`, `class_curie`, `class_name`, `class_model_uri`, `solution`, `spectrometer`, `acquisition_nuclei`, `has_dimension`, `pulse_program`, `id`, `assay_date`, `pulse_program_custom`, `observed_frequencies`, and `name`.

Creating knowledge graphs from research data

Step 3: parsing values from JCAMP-DX



- JCAMP-DX datasets as source for NMR
 - around for long and open
 - Python parser already exists
 - unfortunately many dialects :(
 - contains only limited metadata
 - ★ must come from other sources
 - assumed to be auto-generated in ELNs
- still hard to get many .jdx files
 - we used Havard Dataverse, Chemotion, OSDB

Creating knowledge graphs from research data

Step 3: parsing values from JCAMP-DX



```
dataset_info.yaml
Typ: YAML-Datei

Limonene_7020ug200uL_CDCI3_13CNMR_400MHz_JDX.jdx
Typ: JDX-Datei

Limonene_7020ug200uL_CDCI3_COSY_400MHz_JDX.jdx
Typ: JDX-Datei

Limonene_7020ug200uL_CDCI3_HMBC_400MHz_JDX.jdx
Typ: JDX-Datei

dataset_info.yaml
9  licence_url: https://creativecommons.org/share-your-work/public-domain/cc0
10 date_created: "2019-10-28"
11 description: "NMR data of D-limonene in DMSOd6. The dataset contains 1D 1H 13C as
    well as 2D COSY, HSQC, HMBC, all acquired at 400 MHz (Jeol 400 MHz spectrometer
    with SuperCOOL Probe) (2019-10-07). Related Publication Can Invalid Bioactives
    Undermine Natural Product-Based Drug Discovery? J. Med. Chem. doi: 10.1021/
    acs.jmedchem.5b01009 https://doi.org/10.1021/acs.jmedchem.5b01009. Related
    Material Indofine 5989-27-5 Lot #025082s 7.02 mg 3mm Tube Cambridge Isotope
    DML-10-10X1 Lot #10E-645 Solvent volume 200 µL."
12 id: doi:10.7910/DVN/2UEA9M
13 name: D-Limonene 400 MHz in DMSOd6 NMR data
14 created_by:
15   - "Kim, Seon Beom (University of Illinois at Chicago) - ORCID: 0000-0001-8015-1404"
16   - "Simmler, Charlotte (University of Illinois at Chicago) - ORCID:
    0000-0002-6923-2630"
17   - "Bisson, Jonathan (University of Illinois at Chicago) - ORCID: 0000-0003-1640-9989"
18   - "Pauli, Guido (University of Illinois at Chicago) - ORCID: 0000-0003-1022-4326"
19
20 #####
21 # sample metadata #
22 #####
23 assays_sample:
24   # you MUST provide at least one of the following metadata on the assayed sample
25   name: D-Limonene
26   id: pubchemCID:440917
27   formula: C10H16
28   iupac_name: (4R)-1-methyl-4-prop-1-en-2-ylcyclohexene
29   inchi: InChI=1S/C10H16/c1-8(2)10-6-4-9(3)5-7-10/h4,10H,1,5-7H2,2-3H3/t10-/m0/s1
30   inchikey: XMGOYMMWDOXHJM-JTQLQIEISA-N
31   smiles: CC1=CCC(CC1)C=C
32   mol_file:
33     id: doi:10.7910/DVN/2UEA9M/JOCQYF
34     name: D-Limonene.mol
35
36 #####
37 # records in dataset #
38 #####
39 contains_assay_records:
40   # you MUST provide at least an URI as the id of each assay record (.jdx) in this
    dataset. If the records do not have an URI, you must use "nmrSPARQL:" as prefix and
    the filename (e.g. "id: nmrSPARQL:filename1.jdx")
41   - id: nmrSPARQL:Limonene_7020ug200uL_CDCI3_13CNMR_400MHz_JDX
42     source: Limonene_7020ug200uL_CDCI3_13CNMR_400MHz_JDX.jdx
```

- dataset_info.yaml with metadata not in JDX files
 - dataset provenance (source, creators, PID, ...)
 - metadata of the assayed sample (InChI, InChIKey, SMILES, ...)
- dict2yaml.py
 - parse JDX files in batch
 - create one NMR record collection as YAML
 - each JDX is one NMRSpecRecord
- yaml2rdf.bat
 - covert the YAML to TTL

Creating knowledge graphs from research data

Step 4: load the knowledge graphs into a triple store



- NMR research metadata expressed in triple statements

```
<https://doi.org/10.7910/DVN/F34GVS/KZOQZU/1HNMR_jdx> a ns1:NmrSpecRecord ;  
  ns2:OBI_0000312 <https://doi.org/10.7910/DVN/F34GVS/KZOQZU/1HNMR_jdx/Assay> ;  
  dc:source "1HNMR_jdx.jdx" .  
  
<https://doi.org/10.7910/DVN/F34GVS/KZOQZU/1HNMR_jdx/Assay> a ns2:CHMO_0000613 ;  
  ns2:OBI_0000293 <https://doi.org/10.7910/DVN/F34GVS/KZOQZU/1HNMR_jdx/Solution> ,  
    <https://doi.org/10.7910/DVN/F34GVS/KZOQZU/1HNMR_jdx/Spectrometer> ;  
  ns1:acquisition_nuclei "1H" ;  
  ns1:has_dimension "1D" ;  
  ns1:observed_frequencies "399.78219837825"^^xsd:float ;  
  ns1:pulse_program "NMR" .
```

SPARQL Queries



Show me only ¹H NMR Spectra with a resolution higher 400 MHz

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX OBO: <http://purl.obolibrary.org/obo/>

PREFIX NMR: <https://raw.githubusercontent.com/StroemPhi/NMRspec/main/model/schema/>

SELECT ?dataset ?frequency

WHERE {

?dataset NMR:pulse_program "NMR".

?dataset NMR:acquisition_nuclei "1H".

?dataset NMR:observed_frequencies ?frequency

FILTER(?frequency >= 400)

}

The screenshot shows a web-based SPARQL query interface. At the top, there are tabs for 'query', 'add data', 'edit', and 'info'. Below the tabs, the title 'SPARQL Query' is followed by the instruction 'To try out some SPARQL queries against the selected dataset, enter your query here.' There are two buttons for 'Example Queries': 'Selection of triples' and 'Selection of classes'. To the right, there are 'Prefixes' buttons for 'rdf', 'rdfs', 'owl', and 'xsd'. Below these, there are two dropdown menus for 'Content Type (SELECT)' and 'Content Type (GRAPH)', with 'JSON' and 'Turtle' selected respectively. The main area contains the SPARQL query code, which is the same as the one provided in the text. At the bottom, there is a table with 6 results, showing the 'dataset' and 'frequency' for each result. The table has a 'Table' button and a 'Response' button. The results are displayed in a table with 2 columns: 'dataset' and 'frequency'. The results are numbered 1 to 6. The 'dataset' column contains URIs, and the 'frequency' column contains numerical values in scientific notation.

SPARQL Query

To try out some SPARQL queries against the selected dataset, enter your query here.

Example Queries

Selection of triples Selection of classes

Prefixes

rdf rdfs owl xsd

Content Type (SELECT) JSON Content Type (GRAPH) Turtle

```
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX OBO: <http://purl.obolibrary.org/obo/>
6 PREFIX NMR: <https://raw.githubusercontent.com/StroemPhi/NMRspec/main/model/schema/>
7
8 SELECT ?dataset ?frequency
9 WHERE {
10   ?dataset NMR:pulse_program "NMR".
11   ?dataset NMR:acquisition_nuclei "1H".
12   ?dataset NMR:observed_frequencies ?frequency
13   FILTER( ?frequency >= 400)
14 }
15
16
17
```

Table Response 6 results in 0.039 seconds

Simple view Ellipse Filter query results Page size: 50

dataset	frequency
1 <https://doi.org/10.7910/DVN/KUEDTG/Linalool_100000ug200uL_CDCI3_1HNMR_900MHz_JDX/Assay>	"900.077600296"^^<http://www.w3.org/2001/XMLSchema#float>
2 <https://doi.org/10.7910/DVN/M8QR20/VEIWX8/Lupeol_LUPE01_q1H/Assay>	"900.077600296"^^<http://www.w3.org/2001/XMLSchema#float>
3 <https://doi.org/10.7910/DVN/SAC0TQ/SNCCAF/Tetrandrine_3090ug200uL_DMSOd6_1HNMR_600MHz_JDX/Assay>	"600.1536009"^^<http://www.w3.org/2001/XMLSchema#float>
4 <https://doi.org/10.14272/QXXCRBSWGPRILJ-UHFFFAOYSA-N/CHMO0000595/Assay>	"400.13160052"^^<http://www.w3.org/2001/XMLSchema#float>
5 <https://doi.org/10.14272/QXXCRBSWGPRILJ-UHFFFAOYSA-N/CHMO0000593/Assay>	"400.132470802"^^<http://www.w3.org/2001/XMLSchema#float>

Federated SPARQL Queries



Federated query across Research Data Triplestore and Wikidata based on PubChemId

PREFIX wdt: <http://www.wikidata.org/prop/direct/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX NMR: <https://raw.githubusercontent.com/StroemPhi/NMRspec/main/model/schema/>

SELECT ?NMRSample ?pubchemId ?item WHERE {

 ?NMRSample rdf:type NMR:NmrSample.

 BIND(REPLACE(str(?NMRSample),

 "https://pubchem.ncbi.nlm.nih.gov/compound/", ""))

 AS ?pubchemId).

 SERVICE <<https://query.wikidata.org/bigdata/namespace/wdq/sparql>>

{

 ?item wdt:P662 ?pubchemId.

}

}

The screenshot shows a web interface for running SPARQL queries. At the top, there are buttons for 'query', 'add data', 'edit', and 'info'. Below these is a section titled 'SPARQL Query' with a placeholder text 'To try out some SPARQL queries against the selected dataset, enter your query here.' To the right of this section are 'Prefixes' (rdf, rdfs, owl, xsd) and a 'Content Type (SELECT)' dropdown menu set to 'JSON'. The main area contains the SPARQL query code, which is the same as the one shown in the text blocks. Below the query, there is a 'Table' view showing 9 results in 2.932 seconds. The table has three columns: 'NMRSample', 'pubchemId', and 'item'. The results are as follows:

NMRSample	pubchemId	item
<https://pubchem.ncbi.nlm.nih.gov/compound/2758>	2758	<http://www.wikidata.org/entity/Q161572>
<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	1548943	<http://www.wikidata.org/entity/Q273169>
<https://pubchem.ncbi.nlm.nih.gov/compound/5280805>	5280805	<http://www.wikidata.org/entity/Q407857>
<https://pubchem.ncbi.nlm.nih.gov/compound/64945>	64945	<http://www.wikidata.org/entity/Q416260>
<https://pubchem.ncbi.nlm.nih.gov/compound/222284>	222284	<http://www.wikidata.org/entity/Q121802>

Federated SPARQL Queries



Find matching entries in further data repository: Like finding a corresponding Mass Spectra in MassBank

PREFIX wdt: <http://www.wikidata.org/prop/direct/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX NMR: <https://raw.githubusercontent.com/StroemPhi/NMRspec/main/

SELECT ?NMRSample ?pubchemId ?item WHERE {

?NMRSample rdf:type NMR:NmrSample.

BIND(REPLACE(str(?NMRSample),

"https://pubchem.ncbi.nlm.nih.gov/compound/", ""))

AS ?pubchemId).

SERVICE <<https://query.wikidata.org/bigdata/namespace/wdq/sparql>>

{

?item wdt:P662 ?pubchemId.

?item wdt:P6689 ?massbank.

}

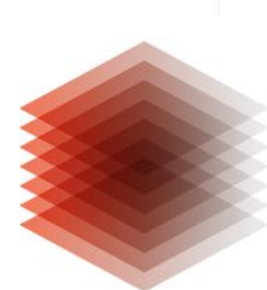
}

The screenshot shows a web interface for running SPARQL queries. At the top, there are buttons for 'query', 'add data', 'edit', and 'info'. Below these is a section titled 'SPARQL Query' with a description: 'To try out some SPARQL queries against the selected dataset, enter your query here.' There are also 'Example Queries' and 'Prefixes' sections. The 'Prefixes' section shows 'rdf', 'rdfs', 'owl', and 'xsd'. The 'Content Type (SELECT)' dropdown is set to 'JSON'. The 'Content Type (GRAPH)' is also visible. The query text is as follows:

```
1 PREFIX wdt: <http://www.wikidata.org/prop/direct/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX owl: <http://www.w3.org/2002/07/owl#>
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX OBO: <http://purl.obolibrary.org/obo/>
6 PREFIX NMR: <https://raw.githubusercontent.com/StroemPhi/NMRspec/main/model/schema/>
7 SELECT ?NMRSample ?massbank WHERE {
8   ?NMRSample rdf:type NMR:NmrSample.
9   BIND(REPLACE( str(?NMRSample), "https://pubchem.ncbi.nlm.nih.gov/compound/", "")) AS ?pubchemId.
10  SERVICE <https://query.wikidata.org/bigdata/namespace/wdq/sparql> {
11    ?item wdt:P662 ?pubchemId.
12    ?item wdt:P6689 ?massbank.
13  }
14 }
```

Below the query, there are tabs for 'Table', 'Response', and '54 results in 3.451 seconds'. The 'Table' tab is selected, showing a table with two columns: 'NMRSample' and 'massbank'. The table contains 5 rows of results:

	NMRSample	massbank
1	<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	PR100259
2	<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	PR100683
3	<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	TY000093
4	<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	TY000245
5	<https://pubchem.ncbi.nlm.nih.gov/compound/1548943>	WA001601



TIB

Thank you