Challenges of Trustable AI and Added-Value on Health
B. Séroussi et al. (Eds.)

doi:10.3233/SHTI220477

An Ontology and Data Converter from RDF to the i2b2 Data Model

Jules FASQUELLE-LOPEZ [a], Jean Louis RAISARO [a]
[a]*Lausanne University Hospital*

Abstract. In a national effort aiming at cross-hospitals data interoperability, the Swiss Personalized Health Network elected RDF as preferred data and meta-data representation format. Yet, most clinical research software solutions are not designed to interact with RDF databases. We present a modular Python toolkit allowing easy conversion from RDF graphs to i2b2, adaptable to other common data models (CDM) with reasonable efforts. The tool was designed with feedback from clinicians in both oncology and laboratory research.

Keywords. RDF, i2b2, SPHN, interoperability, semantic web

1. Introduction

The Swiss Personalized Health Network (SPHN) defined its three-pillar strategy for medical data interoperability [1]. As the first and second pillars aim at, respectively, developing a common semantic framework and formally representing meta-data and data using the Resource Description Framework (RDF)[2], the third pillar encompasses metadata and data transformation in a use-case dependent manner for cross-institutional projects. The present work provides a first prototype for a modular and versatile implementation of the third pillar objective.

We propose a RDF converter that can extract both metadata and data samples from RDF graphs and transform them into the destination common data model (CDM) while minimizing information loss. As a first instantiation of this framework, we focus on the i2b2 common data model [3]. i2b2 is currently in use in several Swiss university hospitals and can count on a large user base both in the US as well as in Europe. Prior work on this subject draw equivalences between RDF and i2b2 [4], or detail a graph pruning algorithm [5] which inspired one of our modules.

Created by Murphy et al. [3], Informatics for Integrating Biology and the Bedside (i2b2) offers a modular database system for storing clinical data divided in service cells. The ontology cell (ONT) compiles all ontology information as unique codes and relies on file-system-like paths to encapsulate concept hierarchies. Another cell, named data repository cell (CRC), holds the patient observations as instances of elements referenced in the ontology cell. Patient, data provider, encounter and other details are stored using additional dimension tables available in the i2b2 “star schema”. Our converter outputs CSV files ready to push into the i2b2 database.

2. Methods

By definition, the main challenge in building any type of connector is the diversity of scenarios and how generic the tool should be. This work was initiated in 2020 as an ad-hoc development task for the MedCo[6] project. Indeed, being funded by the SPHN consortium, the MedCo pilot deployments make use of the nominal SPHN semantics and ontologies available as RDF knowledge graphs. At the same time, MedCo provides a network privacy-preserving analytics layer on top of distributed i2b2 databases.

We aimed at developing a tool that would be robust to both minor architecture changes in the RDF graphs and representation in the output format. This made a priority of getting the framework easily configurable and expandable. To achieve this, we designed a modular architecture where much of the CDM-related dependencies are pushed to external configurable profiles as much as possible. The Converter consists of two main modules: an **Ontology converter**, and a **Data samples converter**. While complementary, their use cases might differ in practice: collaborative studies typically feature a common ontology and local data samples.

The Ontology converter is itself designed as two components interfacing on a very simple data structure, as explained in the next subsections.

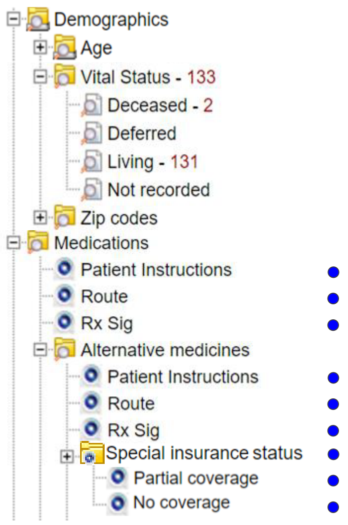
2.1. *Ontology converter*

The design work for ontology conversion boils down to extracting meaningful spanning trees from the RDF knowledge graph. Main concepts are discriminated and labeled as entry points from which graph explorations are performed.

Authors of [4] note that the *Subject - Predicate - Object* RDF base triple is similar to the i2b2 *Subject - Predicate - Object* principle. Yet, while with RDF an Object can itself be a Subject, i2b2 integrates a strict hierarchy between Concepts and Modifier values. In i2b2, Modifiers can only be subordinates to Concepts. This strict hierarchy is illustrated in Figure 1.

The RDF classes do not fit naturally into the two i2b2 categories (Concept, Modifier). We decided to consider as i2b2 Concepts only the medical concepts which could be instantiated without the need of an other concept as context. For example, a *Drug* observation for a patient does not bear a natural meaning outside of its context, say *Drug Allergy* or *Drug Prescription*. More formally, RDF classes that appear as a property of another class are a priori discarded from the list of entry points.

Yet, this list of classes is entirely configurable. It defines the entry points for the graph discovery.



Demographics
- Age
- Vital Status - 133
  - Deceased - 2
  - Deferred
  - Living - 131
  - Not recorded
- Zip codes

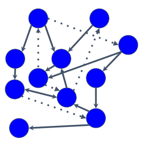
Medications
- Patient Instructions ●
- Route ●
- Rx Sig ●
- Alternative medicines
  - Patient Instructions ●
  - Route ●
  - Rx Sig ●
  - Special insurance status ●
    - Partial coverage ●
    - No coverage ●

Figure 1. An example of i2b2 ontology where "modifiers" are indicated by a blue dot, every other element is a "concept". Modifiers are not mandatory, and when present are directly or indirectly subordinated to a concept. They can also be subordinated to an other modifier.

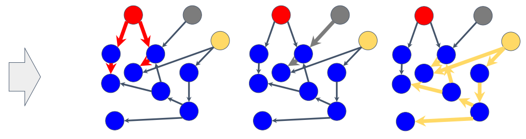
Extracting from RDF: We define a first module to extract a tree from the RDF ontology graph, abstracting all RDF-specific terminology into a simple parent/children interface. The steps of the algorithm are shown in Figure 2.

Starting from the main classes defined as "entry points" to the graph, a recursive walk is performed through specific edges only, in the same fashion as described in [5]. While Stöhr et al. were using `skos:topConceptOf/skos:hasTopConcept` to navigate the graph, we use by default the RDF Schema norm `rdfs:subClassOf/rdfs:domain/rdfs:range`. Our configuration files allow swapping these Unique Resource Identifier (URIs) for equivalent ones (if using an other RDF norm). By design of [1], the subgraph defined by such links does not contain cycles.

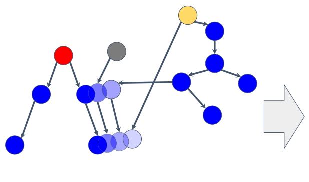
We project every visited RDF element onto two simple Python classes: `Concept` and `Property`. Each Concept instance contains an array of all Properties that apply to it (directly drawn from the RDF `rdfs:domain` predicate). Likewise, each Property instance contains an



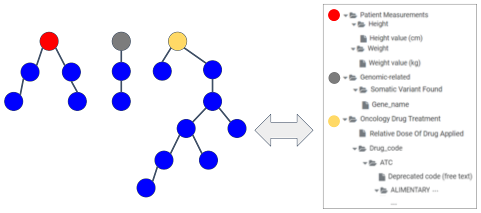
a. Initial graph.



b. Defining source nodes (red, gray, yellow) and discover subgraphs (acyclic by design) from each source node.



c. List every path from each source node. Subpaths can then be explored more than once if shared by distinct sequences.



d. The resulting forest is topologically the same data structure as an i2b2 ontology (here displayed in the MedCo client).

Figure 2. The goal of the RDF parsing module is to unfold a graph (a) into a collection of trees (d).

array of all Concepts it points to (directly drawn from the RDF `rdfs:range` predicate). This interface captures the useful information from the RDF graph without its complexity. At the same time, the two-classes system prefigures the i2b2 Concept/Modifier model.

Both our Concept and Property classes implement a `get_children()` interface, returning the elements of their respective array. They also implement information interfaces allowing to retrieve either comments, language-specific descriptions of the item, or the URI of the original RDF item.

Drawing i2b2 metadata models: The i2b2-specific module fills one or several i2b2 metadata tables given a collection of Python objects implementing the `get_children()` interface.

In i2b2, every ontology item is characterized by its *path*, which is the concatenation of the path of its parent and its own name – which can optionally be different from its display name. Based on a tree built as described in the previous section, it is enough for this i2b2-specific module to visit every element in the tree top-down (using recursive calls to `get_children()`), constantly adding new *paths* to the output data frame.

A set of i2b2-specific routines using the other available interfaces (to determine a display name, a display icon, tooltips or a contextual menus, etc.) are also triggered to fulfill the i2b2 database norms. We define a trivial policy for Concept/Modifier distribution: the children of a python Concept (see subsection 2.1) yield modifiers unless the said concept is flagged as abstract, and all the children of a modifier are modifiers. Extending the Converter to a new data model implies writing a new CDM-specific module using the same interface. No querying of the RDF graph is needed.

2.2. *Data samples converter to i2b2*

Data loading occurs more frequently than ontology creation. This Data converter is then less complex than the Ontology converter. It implements direct links between RDF instances and i2b2 data tables.

The i2b2 data model implements SQL joins within the data repository (CRC) cell [3] using a unique *basecode*. The Ontology converter and the Data samples converter should independently generate matching basecodes.

In our design, the unique i2b2 basecode is drawn directly from the URI of the underlying RDF class or property (accessible through the Python interfaces) and its parent basecode. It encapsulates a *RDF-like* path at a very low level. Thanks to this design, the "data graph" already contains all necessary information to construct the basecodes, without embedding the whole "ontology graph".

Primary types such as text and numerical values encountered in the RDF resources are then added in the `value` fields of the appropriate table. The mappings between RDF types and the i2b2 specific tables and columns are entirely configurable.

2.3. Configuration

We provide three configuration files in JSON format. One is for the RDF ontology reading, one for the CDM-specific metadata writing and one for the data converter. They allow link configuration, concept blacklisting, etc. They also include lookup tables between the data model columns and the RDF data types (primary types (`xsd:string`, `xsd:double`) or more complex items (patient information, encounter details, etc).

3. Results

The expected output of the Converter is a full i2b2 schema as collection of mandatory tables in CSV format that can be pushed to an i2b2 database using simple SQL commands. As input, the Converter can take several ontology graphs and merge them if necessary. Terminology-specific graphs (such as ICD-10, LOINC, SNOMED-CT, etc.) are typically loaded separately in memory. In this implementation, all such terminologies are provided and maintained in RDF representation by the SPHN Data Coordination center [10]. We successfully tested the software against both the SPHN core ontology [9] and project-specific ontologies from the Swiss Precision Oncology and the Swiss BioRef SPHN-funded projects[11], which all feature large terminologies such as SNOMED-CT and ICD-10, for a total number of parsed items going over hundreds of thousands. All ontologies displayed in current MedCo demos and deployments were generated by the Converter based on RDF graphs issued by SPHN.

The outcome of the converter was validated from end-users and clinical researchers of the above-mentioned SPHN projects and the SPHN Data Coordination Center. Their feedback on the i2b2-formatted ontology allowed to define clearly which irrelevant RDF properties should be dropped, if macro-concepts should be created to increase user experience, etc. For instance, thanks to end-users' feedback, we integrated into the RDF to i2b2 ontology conversion process the codes' identifiers within the display names of all terminology items. This means the RDF ontology item of class *atc:A01A* with property *rdfs:label STOMATOLOGICAL PREPARATIONS* maps to the i2b2 element displayed as *ATC:A01A - STOMATOLOGICAL PREPARATIONS*. This allows both to leverage the automatic lexicographical ordering in i2b2's Web client and for end-users to search items by their code rather than by their label. The Converter is open-source and hosted on Github at the following address [12].

4. Discussion

The ontology Converter performs two distinct explorations: it first queries the RDF graph to construct a tree exposing an interface (2.1), and then walks the tree to craft i2b2 metadata entries (2.1). We consider this an acceptable trade-off between performance and modularity. Other works either implement direct pipelines from SPARQL (the query language for RDF) to i2b2 [4], [5], or cap the complexity of the supported RDF graphs [5]. With our design, adding support to a new common data model only requires replacing the i2b2-specific module, without touching the RDF parsing module. Moreover, an ontology converter is typically not a critical application in terms of performance requirements, as opposed to a data ETL.

A useful enhancement of the Converter would be handling of distant RDF graphs. For now, all graphs are loaded in the local memory of the machine running the tool. It is then limited by the RAM capacity. We do not either provide devops tools to create the output CSV tables on a distant machine (say, an i2b2 server).

Adding support of synonyms would also be a benefit, since both i2b2 and RDF feature such notions. The i2b2 storage system avoid duplications, which we do not do for now.

Prior work has built bridges between i2b2 and other data models, either through query translation [7] or full data model conversion pipelines [8]. The complexity of these tasks is partly due to the i2b2 specific representations that our interface aims to isolate. This advocates for a generic tool such at this one, that can be augmented for multi-model support.

5. Conclusion

The strength of our tool resides in its modularity and low-cost extensibility to new data models, but also in the fine-grained configuration of specific mappings. Designing the Ontology converter featuring a neutral tree interface also increases the tool's resilience to changes in the RDF schema.

Either it is used with the attached data converter or standalone, the ontology converter holds a predictable logic to map arbitrary large RDF knowledge graphs to the i2b2 common data model. Indeed, several hospitals participating in the SPHN initiative already have an i2b2 ETL pipeline from their data warehouse. In such cases, it would make sense to simply provide the converted i2b2 ontology for mapping local variables to i2b2 observations.

References

[1] Gaudet-Blavignac C, Raisaro JL, Touré V, Österle S, Crameri K, Lovis C. A National, Semantic-Driven, Three-Pillar Strategy to Enable Health Data Secondary Usage Interoperability for Research Within the Swiss Personalized Health Network: Methodological Study. JMIR Med Inform. 2021 Jun 24;9(6):e27591.

[2] RDF Schema 1.1 [Internet]. [cited 2022 Jan 18]. Available from: https://www.w3.org/TR/rdf-schema/

[3] Murphy SN, Weber G, Mendis M, Gainer V, Chueh HC, Churchill S, et al. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). Journal of the American Medical Informatics Association. 2010 Mar 1;17(2):124–30.

[4] Solbrig HR, Hong N, Murphy SN, Jiang G. Automated Population of an i2b2 Clinical Data Warehouse using FHIR. AMIA Annu Symp Proc. 2018 Dec 5;2018:979–88.

[5] Stöhr, Hr MR, Majeed RW, Günther A. Metadata Import from RDF to i2b2. German Medical Data Sciences: A Learning Healthcare System. 2018;40–4.

[6] Raisaro JL, Troncoso-Pastoriza JR, Misbach M, Sousa JS, Pradervand S, Missiaglia E, et al. MedCo: Enabling Secure and Privacy-Preserving Exploration of Distributed Clinical and Genomic Data. IEEE/ACM Trans Comput Biol Bioinform. 2019 Aug;16(4):1328–41.

[7] Majeed R, Fischer P, Günther A. Accessing OMOP Common Data Model Repositories with the i2b2 Webclient – Algorithm for Automatic Query Translation. In: Studies in health technology and informatics. 2021.

[8] Klann JG, Joss MAH, Embree K, Murphy SN. Data model harmonization for the All Of Us Research Program: Transforming i2b2 data into the OMOP common data model. PLOS ONE. 2019 Feb 19;14(2):e0212463.

[9] The SPHN RDF Schema [Internet]. [cited 2022 Mar 31] Available from: https://www.biomedit.ch/rdf/sphn-ontology/sphn

[10] Krauss P, Touré V, Gnodtke K, Crameri K, Österle S. DCC Terminology Service—An Automated CI/CD Pipeline for Converting Clinical and Biomedical Terminologies in Graph Format for the Swiss Personalized Health Network. Applied Sciences. 2021 Jan;11(23):11311.

[11] SPHN Driver projects [Internet]. [cited 2022 Mar 31]. Available from: https://sphn.ch/network/projects/driver-projects/

[12] RDF-i2b2 converter, on CHUV Data Science Lab's GitHub, https://github.com/CHUV-DS/RDF-i2b2-converter.