DRAFT Overview: FY2022 HUD HMIS Logical Model v0.0.1

Purpose of the HMIS Logical Model

The HMIS Logical Model exists to define normative relationships for the HUD HMIS data elements, independent of any technical implementation considerations.

The model is intended to technically express the underlying constraints described in the HMIS Data Standards Model and the HMIS Data Dictionary (see https://www.hudexchange.info/resource/3824/hmis-data-dictionary). The Logical Model makes no assumption about how the HMIS standards will be implemented. The Logical Model is not intended to be specific to any database, data flows, or even to databases in general. The Logical Model simply describes basic data model constraints for any system fully adhering to the HMIS Data Standards.

You can use the HMIS Logical Model to:

- quickly get an overview of the HMIS data structures and how the relate to each

other
- map between HMIS and other existing models for interoperability purposes

What new for FY2022

For this HMIS Data Standard release, the HMIS Logical Model is represented as a proper ontology, useful for mapping to other ontologies in health and human services. The idea of web ontologies is very simple, though the topic can become technical: put the definitions in a format that is standard across all systems, then you can connect across those definitions.

One motivator for this model format change is the increasing convergence of health care and homeless services. Health care's standards are represented as a web ontology to model their data structures. See:
- human readable data structures list at: https://www.hl7.org/fhir/resourcelist.html
- the health ontology is downloadable here: https://www.hl7.org/fhir/downloads.html
- Overview: https://www.hl7.org/fhir/rdf.html .

Usage of this model has allowed health care to flexibly interoperate within itself, and with other domains.

In the past (pre FY2022), the HMIS Logical Model consisted of a human readable image of a class diagram. For 2022, we still have the images, but they are generated from an underlying computer model made in the Web Ontology Language (OWL), the W3C's format for defining Knowledge Graphs. For an introduction to ontologies and OWL, watch this short video: https://youtu.be/jfUPLuPL3Ho. This allows the HMIS Logical model to be compared and related to other models on the web, allowing for easy interoperability. OWL is built upon RDF, the same technology upon which RSS for podcasts is based.

How to get the HMIS Logical Model

The FY2022 Logical Model is hosted on Github, and the raw ontology file is located here: https://raw.githubusercontent.com/hmis-interop/logical-model/FY2022-latest/src/logical-model.n3.owl

How to view the HMIS Logical Model

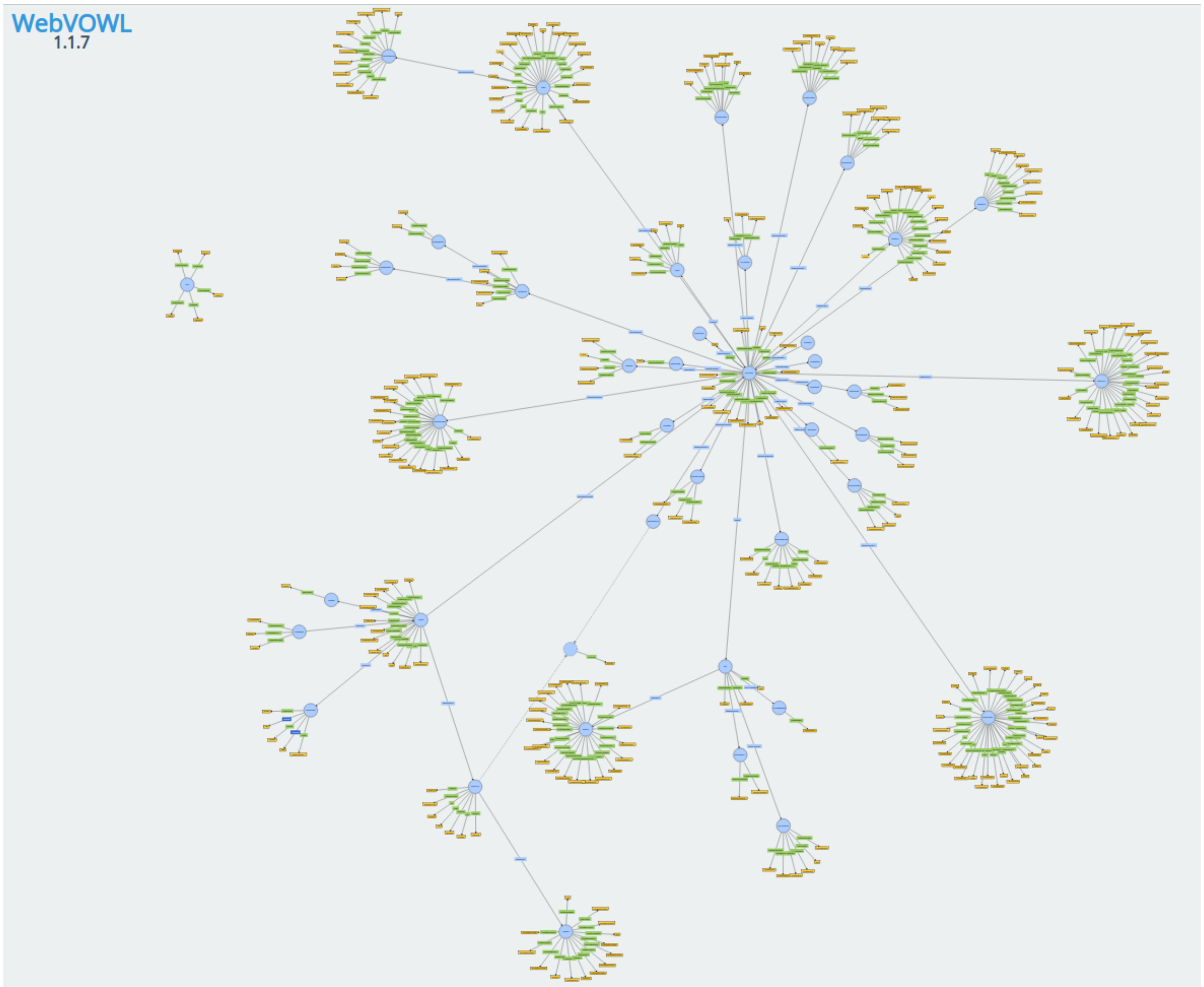
A convenient visualization of this model can be obtained from this url. http://www.visualdataweb.de/webvowl/#iri=https://raw.githubusercontent.com/hmis-interop/logical-model/FY2022-latest/src/logical-model.n3.owl

Adjust the filters (especially the degree of collapsing) for more data element detail.

Original editable location for this document is located at: https://docs.google.com/document/d/1XhJEU5rpiJE73CpWo8pTtIAfjbcGp2GYRJq21xsh

DRAFT *Overview: FY2022 HUD HMIS Logical Model v0.0.1*

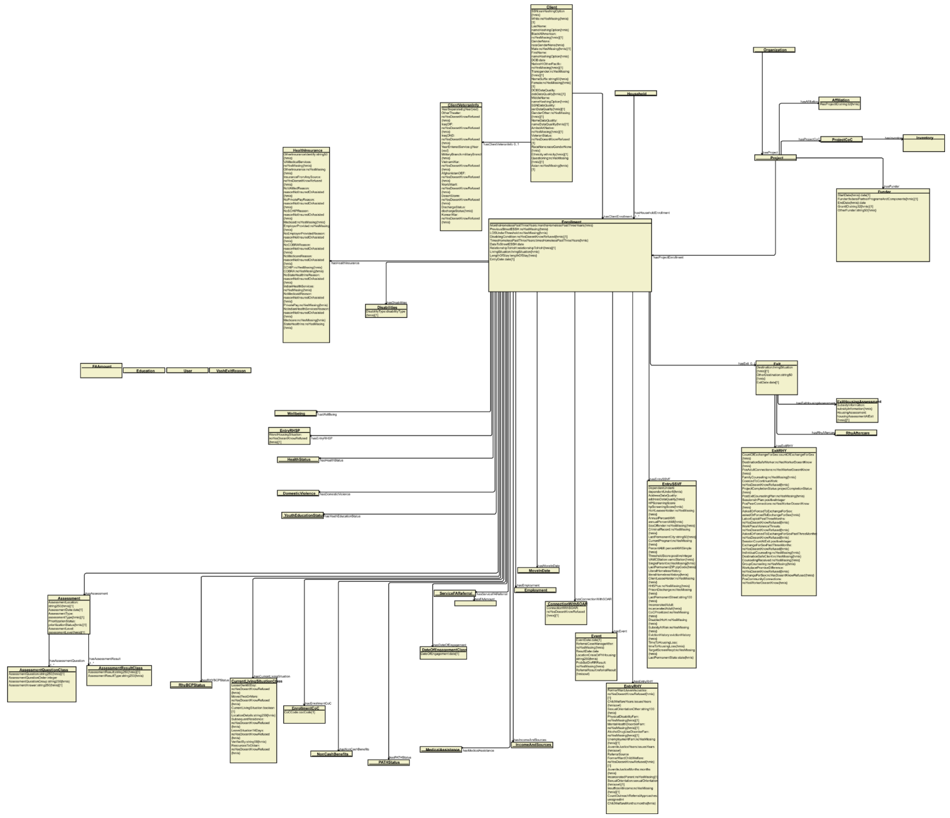
[NBk/edit#]



Draft Graph Diagram representation of FY2022 HMIS Logical Model, generated by browsing [here]

Alternatively, some prefer a "Class Diagram" representation is available at: https://github.com/hmis-interop/logical-model/blob/FY2022/doc/logical-model-class-diagram.svg (a .png image file of the Class Diagram is also available)

DRAFT Overview: FY2022 HUD HMIS Logical Model v0.0.1



Equivalent Draft Class Diagram representation of the Logical Model, generated from the ontology file using http://owlgred.lumii.lv The current generated version of this is here.

What capabilities does this new Logical Model have?

- **Web addresses for each HMIS data element, in a machine readable format**

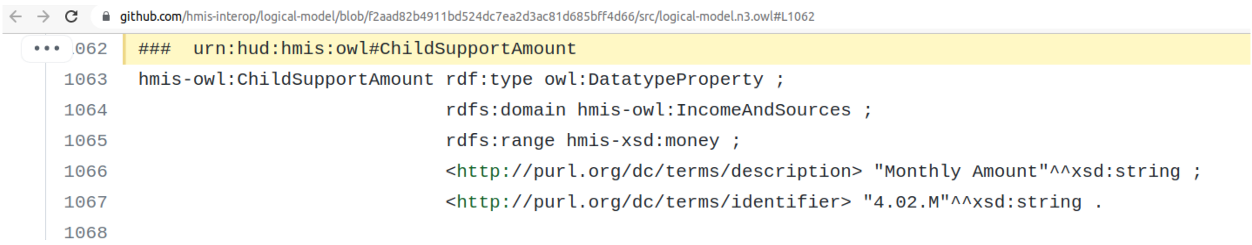
  Regardless of which tool you use to access the web, you can browse to a url, and pull down the definition.

  Each data element has a web url, so you can point to an HMIS Data Dictionary Elements

  For example, using an RDF tool that supports the "#" locator, the url

  https://github.com/hmis-interop/logical-model/blob/FY2022-latest/src/logical-model.n3.owl#ChildSupportAmount

  ...will take you to the definition for the ChildSupportAmount data element. There you can see it's RDF definition containing the HMIS Data Standard ID, text description, and data type.



github.com/hmis-interop/logical-model/blob/f2aad82b4911bd524dc7ea2d3ac81d685bff4d66/src/logical-model.n3.owl#L1062

```
###  urn:hud:hmis:owl#ChildSupportAmount
hmis-owl:ChildSupportAmount rdf:type owl:DatatypeProperty ;
                            rdfs:domain hmis-owl:IncomeAndSources ;
                            rdfs:range hmis-xsd:money ;
                            <http://purl.org/dc/terms/description> "Monthly Amount"^^xsd:string ;
                            <http://purl.org/dc/terms/identifier> "4.02.M"^^xsd:string .

```

This is great for interoperability, because machines can look up definitions on the fly, for mapping or other purposes.

- **Define mixed payloads, of HMIS and other data or make a Knowledge Graph**

  Use the HMIS Logical Model as a precise payload format definition. You can make payloads with it (as RDF XML, or JSON-LD). Like XML Schema, OWL is a template for actual data. It is more flexible than XML Schema, though, and you can quickly make custom payloads that have health RDF and HMIS RDF.

  Or, you can build a Knowledge Graph with data that follows the Logical Model, which could serve as a Data Lake for research or operational purposes.

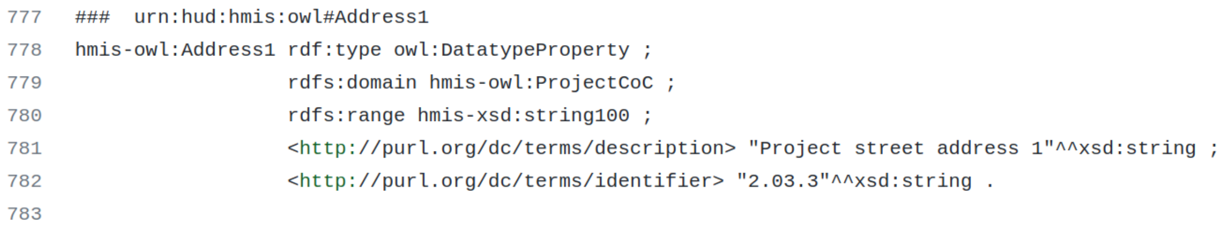
- **Make knowledge graphs, and perform mapping and logical inferencing with other ontologies**

  For introductory information on this, see:
  - https://web.stanford.edu/class/cs520/2020/notes/What_Are_Some_Inference_Algorithms.html
  - https://medcraveonline.com/IRATJ/ontology-mapping-and-merging-aspects-in-semantic-web.html

Differences from past versions of the HMIS Logical Models

- **HMIS Dictionary numbering and descriptions within each model element**

  In the Logical Model OWL snippet below, you can see that the HMIS Address1 data element is defined by the HMIS XML Schema as a 100 character string, has a description of "Project Street Address 1" and corresponds to FY2022 HMIS Data Dictionary data element #2.03.3.



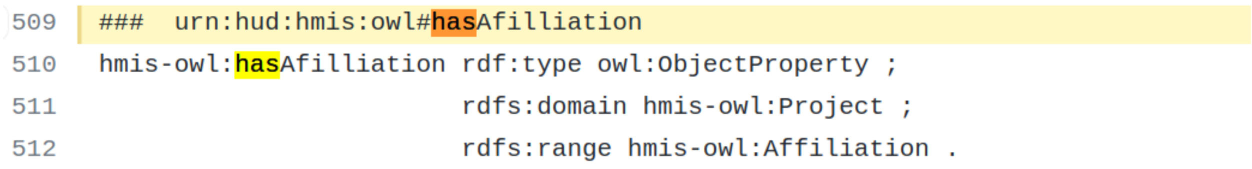
```
###  urn:hud:hmis:owl#Address1
hmis-owl:Address1 rdf:type owl:DatatypeProperty ;
                  rdfs:domain hmis-owl:ProjectCoC ;
                  rdfs:range hmis-xsd:string100 ;
                  <http://purl.org/dc/terms/description> "Project street address 1"^^xsd:string ;
                  <http://purl.org/dc/terms/identifier> "2.03.3"^^xsd:string .

```

- **No Foreign Keys, but there is an equivalent: relationship properties**

  There are no foreign keys in the model's classes. Instead, these relationships are defined as "object properties" shown as lines in generated diagrams replace foreign keys.

  These object properties can carry more meaning than foreign keys (but that's for a more in-depth article).

  However, the relationships still are keyed. In the data files (aka instance files) that can be generated from the HMIS Logical Model as JSON-LD, RDF XML, etc., the identifiers of the class and its other related classes are all in the data payload.



```
###  urn:hud:hmis:owl#hasAfilliation
hmis-owl:hasAfilliation rdf:type owl:ObjectProperty ;
                        rdfs:domain hmis-owl:Project ;
                        rdfs:range hmis-owl:Affiliation .
```

  More on creating data payloads and knowledge graphs from OWL logical definitions can be found here: https://www.w3.org/TR/rdf11-primer.

- **Data picklists are still defined in the HMIS XML Schema, and the model refers to those, instead of redundantly defining them**

  Diagram of how classes refer to XML Schema, FHIR, etc., CSV, XML Schema. etc.

  Move toward more automation tools with the model.

  Libraries for handling RDF and OWL specifically. Python for RDF

Available Tools

Protege: Stanford's open source editing software for OWL, used to make the HMIS Logical Model
Python RDFLib: Open source software libraries for reading and writing OWL and knowledge graph data in many formats
Java RDF4J: Open source software libraries for reading and writing OWL and knowledge graph data in many formats
TripleStores: to store your Knowledge Graph data

Frequently Asked Questions

- **Question: What file format is the Logical Model OWL file, that is checked into Github?**

  Response: The N3 format, one of many ways you can write OWL models. N3 is really easy for people to read. You can fairly easily convert between the many ways of writing RDF. See: https://en.wikipedia.org/wiki/Resource_Description_Framework#Serialization_formats

- **Question: How is this different from the existing HMIS XML Schema?**

  Response: XML Schema requires a hierarchy of concepts, which is good for simple things like lists. But, as HMIS has grown in complexity, and since cross-domain interoperability demands flexible non-hierarchical models, web ontologies are a better fit for information modeling. That said, the HMIS Logical Model refers to all the picklists and basic data type definitions in the XML Schema, since those constructs are what XML Schema is excellent at describing. So, why not just reuse those XML Schema picklists, by pointing to them? It keeps the HMIS Logical Model much shorter, that way, as well.

- **Question: What if I like JSON better than XML or n3?**

  Response: JSON-LD can also be used.

How to assist with the HMIS Logical Model

Submit HUD HMIS Ask-A-Questions, and/or post issues to correspond with the Logical Model developers.