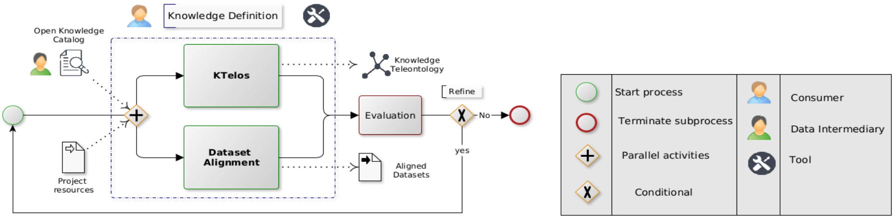
Part 5.6
Phase 4 - Knowledge Definition

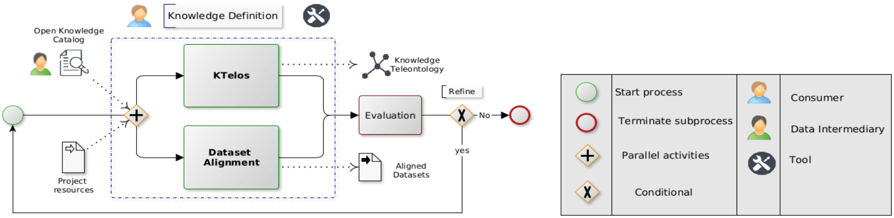
Phase 4 - Knowledge Definition (Theory)

Knowledge Definition Phase



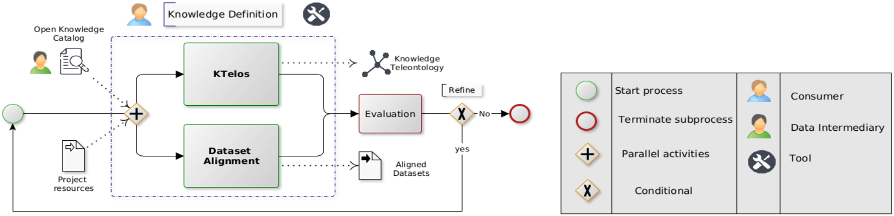
- **Input**: the gathered information resources for the KGE project, the formalized user's purpose and the (open) knowledge catalog, e.g., the LiveKnowledge catalog.
- **Output**: the **knowledge teleontology** relevant to the KGE project and the aligned datasets.
- **kTelos**: the kTelos process aims at choosing and reusing the knowledge resource (termed as a knowledge teleontology) relevant for generating the schema of the final Entity Graph of your KGE project.

Knowledge Definition Phase (contd.)



- The **Dataset Alignment** activity aims at aligning the dataset previously collected, cleaned and formatted, with the modelling choices decided and encoded in the knowledge teleontology.
- The knowledge teleontology considered for the final KG should represent the entity types, properties and data types in a similar structure with respect to their representation into each single datasets you considered for your KGE project.

Knowledge Definition Phase (contd.)



- At the end of the phase, an **evaluation** is carried out whether the knowledge teleontology chosen covers the general entity types relevant to the KGE project purpose and whether the datasets are aligned to the knowledge teleontology in terms of properties and data types. To correct any misalignment, the phase allows a backward **loop**.

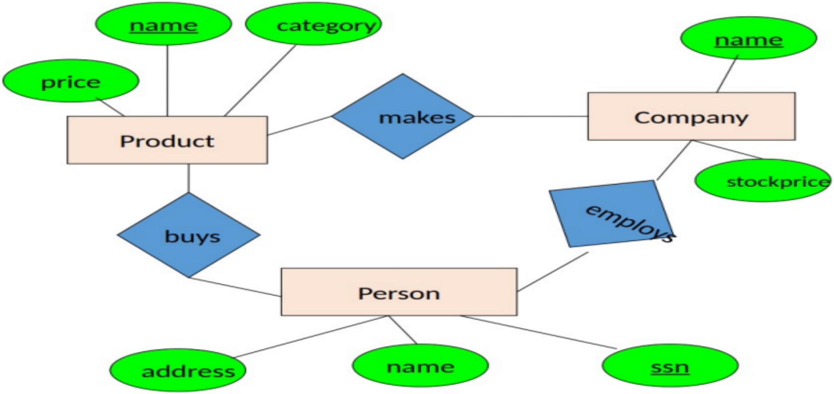
- Overall, the knowledge definition phase aims at:
  - unifying the representation of the information;
  - improving the **interoperability** of the final KG(s) by aligning to a knowledge teleontology.

Phase 4 - Knowledge Definition (Theory)

What are ER Models?

- An **Entity–Relationship (ER) Model** describes interrelated things of interest in a specific domain of knowledge.
- It is composed of **classes** / **entity types** (etypes) (which classify the things of interest, i.e. **entities**) and specifies **relationships** that can exist between entities (instances of those entity types).
- The ER model is, thus, an **abstract data model** that defines a data or information structure which can be implemented in a data/knowledge base.
- It is usually drawn in a graphical form as **boxes (classes)** that are connected by **lines (relationships)** which express the associations and dependencies between entities.

ER Model: A Complete Example



Reference: M. Hahsler. DS1300: The ER Model.

Phase 4 - Knowledge Definition (Theory)

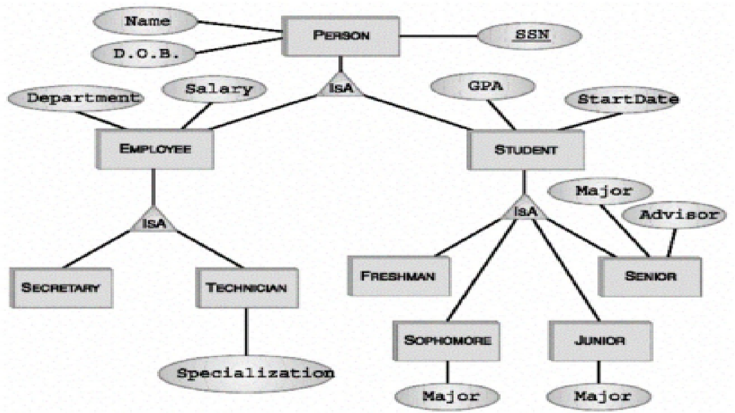
Extended ER (EER) Model

- The **Extended ER (EER) Model** includes all of the concepts introduced by the ER model.
- Additionally it includes the concepts of a **subclass and superclass ('is-a' relation)**. Super class is an entity that can be divided into further sub-classes. Sub class inherits the properties and attributes from super class.
- It also includes **Generalization / Specialization**. Generalization is a process of generalizing an entity which contains generalized attributes or properties of generalized entities. Specialization is a process of identifying subsets of an entity that share some different characteristic.
- It was developed to reflect more precisely the properties and constraints that are found in more **complex data/knowledge bases**.

Characteristics of EER Models

- A subclass is said to **inherit** from a superclass. A subclass can inherit from many superclasses in the hierarchy.
- When a subclass inherits from one or more superclasses, it inherits all their attributes.
- In addition to the inherited attributes, a subclass can also define its own specific attributes.
- The process of making a superclass from a group of subclasses is called generalization.
- The process of making subclasses from a general concept is called specialization.

EER Model Example



Reference: jcsites.juniata.edu

Phase 4 - Knowledge Definition (Theory)

What is an Ontology (The key aspects)?

- "explicit specification of a conceptualization" [Gruber, 1993]
- "formal specification of a shared conceptualization" [Borst, 1997]
- "An ontology is a formal, explicit specification of a shared conceptualization" [Studer et al., 1998]
- But....
  - What is a conceptualization?
  - What is a proper formal, explicit specification?
  - Why is 'shared' of importance?

What is a Conceptualization?

- Formal structure of (a piece of) reality as perceived and organized by an agent, independently of:
  - the vocabulary used
  - the actual occurence of a specific situation
- Different situations involving same objects, described by different vocabularies, may share the same conceptualization
- "mela", "apple": different terms for the same conceptualization…

Formal, Explicit Specification

- We need to use a language to refer to the elements of a conceptualization
  - the language **commits** to a conceptualization
- Problem: a logical signature can be interpreted in arbitrarily many different ways
- Once we commit to a certain conceptualization, we have to make sure to only admit those **models** which are **intended** according to the conceptualization.
  - the intended models of a relation predicate will be those such that the interpretation of the predicate returns one of the various possible extensions (one for each possible world) of the conceptual relation denoted by the predicate.

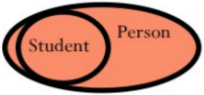
Why is SHARED of importance?

- Sharing whole conceptualizations may not be possible (private to the mind of the individuals)
- Sharing approximations of conceptualizations based on a limited set of examples, and showing the actual circumstances where a certain conceptual relation holds
- Without such minimal sharing, the benefits of having an ontology are limited
  - ontology may turn out useless if it is used in a way that runs counter the understanding of the primitive terms in the appropriate way.
- Any ontology will always be less complete and less formal than it would be desirable in theory.

The Ontology Building Block: IS-A Relation

Is-a Relation

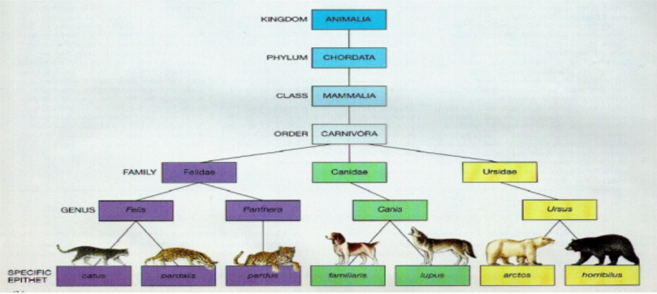
- **is-a relation**: binary relation between concepts (not individuals)
- Examples: Student is-a Person, Air Pollutant is-a Pollutant
  - Informal meaning: all the students are persons (or all the individuals that are students are also persons); if something is an air pollutant, it is also a pollutant
- In set-theoretical terms:



IS-A Hierarchy: Example Taxonomy

Is-a hierarchy

- **taxonomy**: a hierarchical organized subject-based classification system
  - typically depicted in a tree-like structure
- **is-a hierarchy**: taxonomy of concepts organized according to the is-a relation.

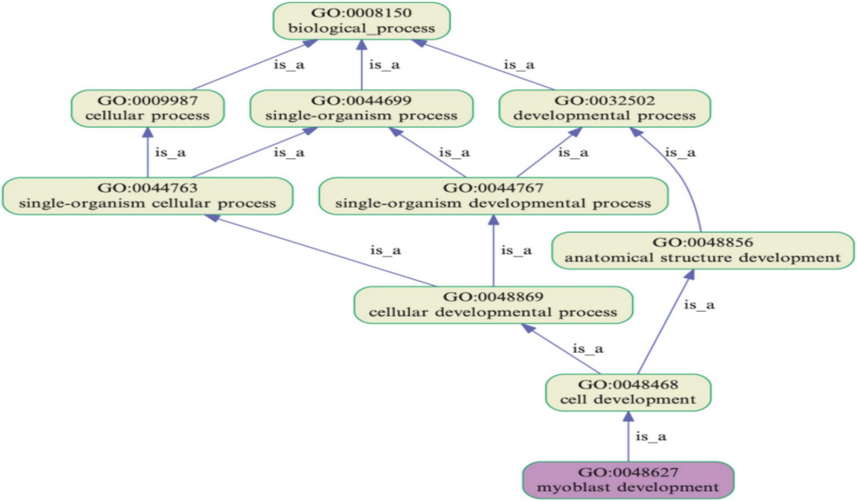


Types of Ontologies

Guarino (1998) categorization

- Top-level ontologies describe very general concepts like space, Time, etc which are independent of a particular problem or domain.
- Domain ontologies and task ontologies describe, respectively, the vocabulary related to a generic domain (like medicine, or automobiles) or generic task or activity (like selling) by specializing the terms introduced in the top-level ontology.
- Application ontologies describe concepts depending both on a particular domain and task, which are often specializations of both the related ontologies.

Example: The GENE Ontology



Phase 4 - Knowledge Definition (Theory)

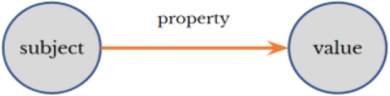
What is RDF?

- A language for representing Web resources and information about them in the form of metadata [RDF Primer]
- A language to represent all kinds of things that can be identified on the Web [RDF Primer]
- A domain independent data model for representing information on the Web [G. Antoniou and F. van Harmelen, 2004]
- A language with an underlying model designed to publish data on the Semantic Web [F. Giunchiglia et al., 2010]

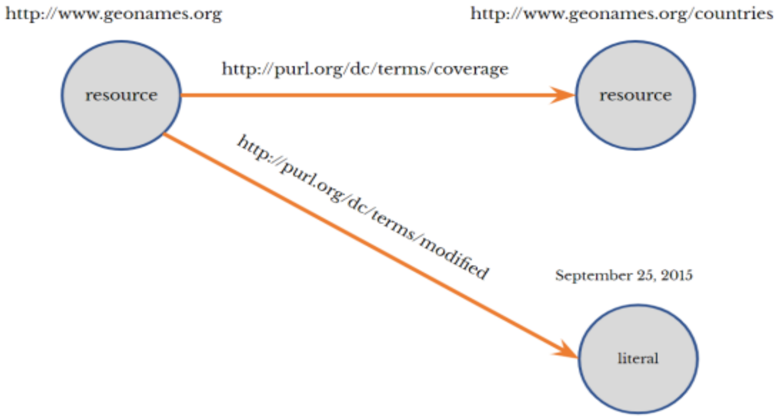
RDF language and data model

RDF language:

- A language for expressing simple statements of the form subject-property-value (binary predicates), with reasoning and inferencing capabilities
- The data model in RDF is a graph data model
- An edge with two connecting nodes forms a triple



RDF Graph


http://www.geonames.org
http://www.geonames.org/countries
http://purl.org/dc/terms/coverage
resource
resource
http://purl.org/dc/terms/modified
September 25, 2015
literal

RDF Schema (RDFS)

RDF:

- RDF is a universal language that lets users describe resources in their own vocabularies
- RDF by default does not assume, nor defines semantics of any particular application domain

RDF Schema (RDFS) [Contd.]

RDF Schema (RDFS): A language defined to provide mechanisms to add semantics to RDF resources, in terms of:

- Classes (**rdfs:Class**) and Properties (**rdfs:Property**)
- Class Hierarchies and Inheritance (**rdfs:subClassOf**)
- Property Hierarchies (**rdfs:subPropertyOf**)
- Domain (**rdfs:domain**) and range (**rdfs:range**) of properties

Requirements for Ontology Languages

Ontology languages allow users to write explicit, formal conceptualizations of domain models (i.e. formal ontologies). The main requirements are:-

- A well-defined formal syntax
- Sufficient expressive power, and convenience of expression
- Formal semantics, and support for efficient reasoning
- A good tread-off between expressivity and efficiency

OWL (Web Ontology Language) has been designed to meet these requirements for the specification of ontologies and to reason about them and their instances

OWL RDF/XML Syntax

HEADER

```
<rdf:RDF
        xmlns:owl ="http://www.w3.org/2002/07/owl#"
        xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:xsd ="http://www.w3.org/2001/XLMSchema#">
```

ONTOLOGY

```
<owl:Ontology rdf:about="">
        <rdfs:comment>An example OWL ontology
</rdfs:comment>
        <owl:priorVersion

rdf:resource="http://www.mydomain.org/uni-ns-old"/>
        <owl:imports
                rdf:resource="http://www.mydomain.org/persons"/>
        <rdfs:label>University Ontology</rdfs:label>
</owl:Ontology>
```

Protégé Ontology Editor (Click Here)

WHY PROTÉGÉ

Protégé's plug-in architecture can be adapted to build both simple and complex ontology-based applications. Developers can integrate the output of Protégé with rule systems or other problem solvers to construct a wide range of intelligent systems. Most important, the Stanford team and the vast Protégé community are here to help.

ACTIVE COMMUNITY

Protégé is actively supported by a strong community of users and developers that field questions, write documentation, and contribute plug-ins.

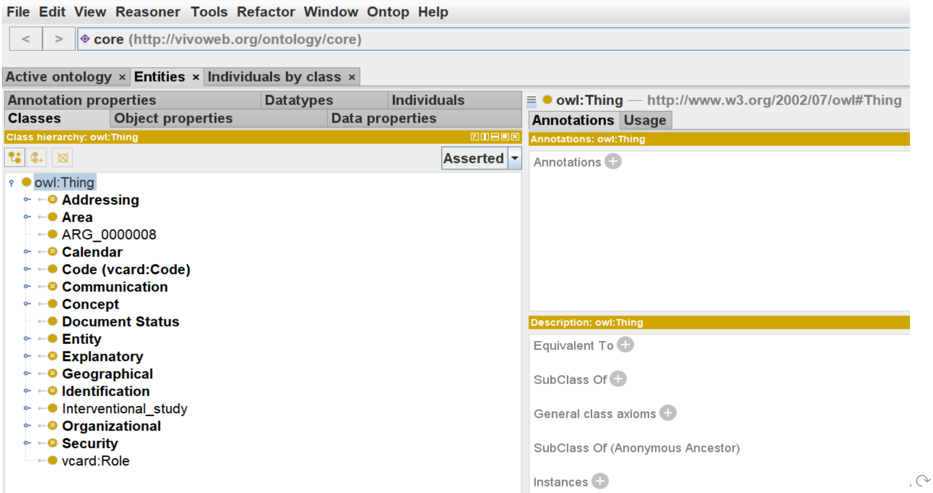
W3C STANDARDS SUPPORT

Protégé fully supports the latest OWL 2 Web Ontology Language and RDF specifications from the World Wide Web Consortium.

EXTENSIBLE OPEN SOURCE ENVIRONMENT

Protégé is based on Java, is extensible, and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development.

Protégé: Interface Illustration



Phase 4 - Knowledge Definition (Theory)

ER/EER models - limitations

ER/EER models have three main weaknesses which hugely affect the reuse of data:

- What **situational context** is the ER model modeling? Its spatio-temporal coordinates are left implicit, *as if* the ER model could be used unchanged at all times and in all locations.
- Where do **data and object properties** come from? A theory providing the guidelines for thinking of the possible ways in which entities interact is missing.
- Where do the **extra etypes of the EER model** come from? The step from an ER model and an EER model is completely undefined.

NOTE: The design of ER models is driven by the application. The design of EER models, as extensions of ER models, is driven by the need of quality and of facilitating reuse.

The design of EER models, as extensions of ER models, should be driven by the need of facilitating reuse.

- Where do the **extra etypes of the EER model** come from?

  The step from an ER model to an EER model is completely undefined.

  The etypes in ER models do not conform to a general theory about *what exists* in the world around us.

  As a result, the etype hierarchies in EER models are developed in a focus-less fashion, without a clear methodology.

  This results in hindrance of:
  (i) data and knowledge reuse, and consequently
  (ii) lack of semantic interoperability.

Phase 4 - Knowledge Definition (Theory)

Limitations of Ontologies

- The **language** in which a standard ontology is written is **ambiguous**, i.e., the words denoting entity types and properties are neither explicitly grounded in natural language (e.g., via WordNet) nor are they semantically disambiguated (e.g., via unique identifiers)
- Given the limitations of the underlying EER model behind a standard ontology, the **design decisions behind modelling how entity types** in a standard ontology are described and interrelated using properties are **left implicit and unspecified.**
- Further, in a standard ontology, it is implicit and unspecified **whether an entity type is purpose-specific**, e.g., for a particular KGE project, **or is a common entity type**, reusable across differet purposes and applications.
- Finally, due to the limitations of the underlying EER model behind a standard ontology, the **process behind how the hierarchy of entity types is modelled** in standard ontologies remains equally **unclear and underspecified.**

Phase 4 - Knowledge Definition (Theory)

Knowledge Teleontology - Definition

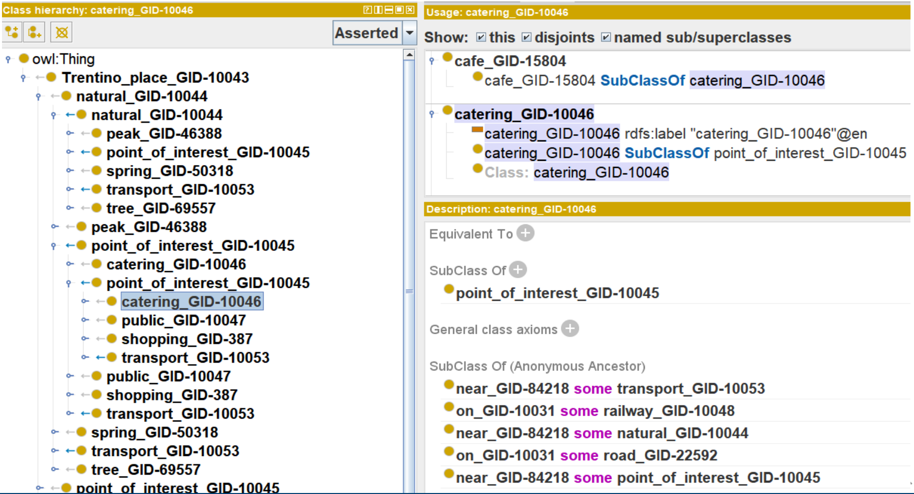
- A knowledge telentology is a graph which encodes entity types:
  - hierarchically structured via **IS-A** and/or PART-OF relations
  - interrelated via **object properties** connecting two etypes
  - described by **data properties** relative to an etype

- A knowledge teleontology is grounded into what exists in the world as the terms modelling **the entity types and properties in a knowledge teleontology** are formally aligned to the UKC hierarchy and **identified uniquely via UKC global identifiers**.

How Knowledge Teleontology overcome limitations of ontology

- The **language** in which a KGE knowledge teleontology is written is **completely unambiguous**, i.e., the words denoting entity types and properties are explicitly grounded in natural language (e.g., via annotating the **UKC**) and they are semantically **disambiguated** (e.g., via unique identifiers - GID - provided by the UKC)
- The **design decisions** behind modelling entity types in KGE via describing and interrelating using properties are **made explicit** by documenting the design decisions encoded in a **knowledge teleontology**.
- Finally, the hierarchy of entity types modelled in KGE is made explicit and specified by following the dedicated **kTelos process** of modelling the knowledge teleontology.

Example Knowledge Teleontology (OSM)

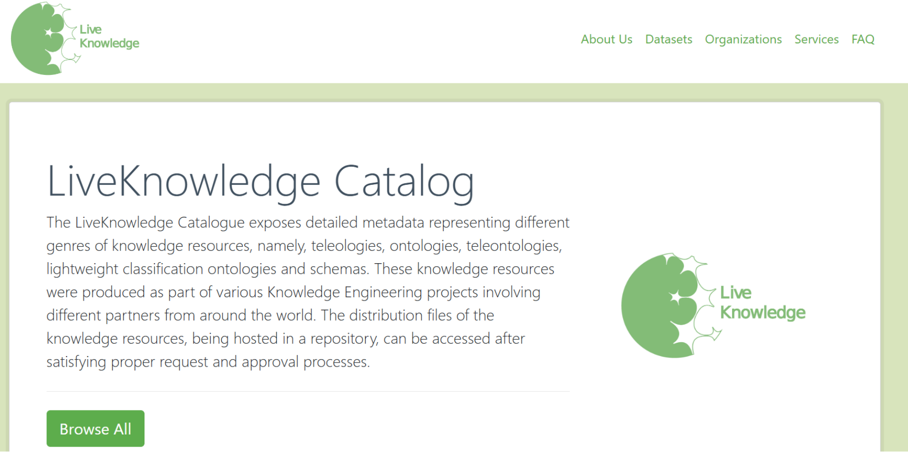


Phase 4 - Knowledge Definition (Theory)

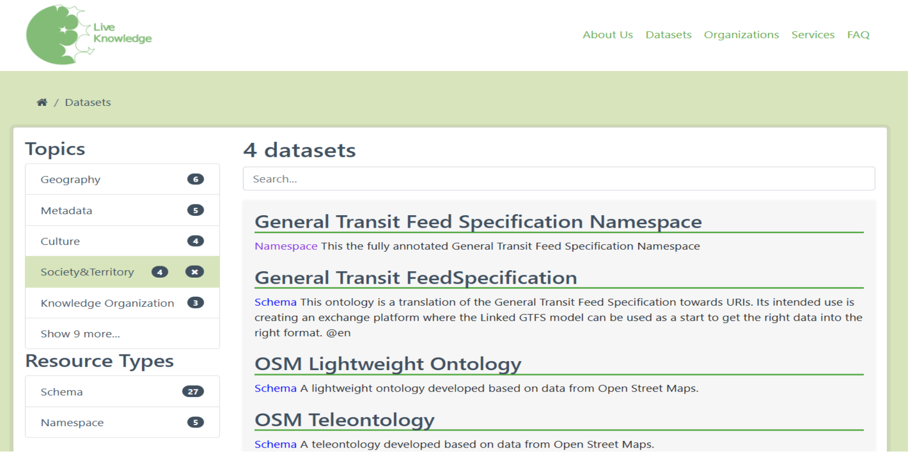
kTelos

- The top-down process for modelling knowledge as knowledge teleontologies is known as the **kTelos** process. It is as follows:
  1. **Input to kTelos:** the domain language aligned to the UKC hierarchy defining unambiguously words which should be used to model entity types and properties.
  2. **kTelos:** using the words in input to generate a knowledge teleontology via the following process:
     - modelling the **entity type hierarchy** by selecting the words denoting entity types and placing them in an IS-A hierarchy.
     - modelling the **object properties** by selecting the words denoting object properties and defining their **domain and range entity types**.
     - modelling the **data properties** by selecting the words denoting data properties and defining their **domain entity type and data type**.
  3. **Output of kTelos:** the knowledge teleontology file in output.
  4. **Knowledge Reuse in kTelos** The knowledge teleontologies thus modelled are discoverable from the **LiveKnowledge** catalog developed at the KnowDive Group. See next slides.

Live Knowledge: Home Page



Live Knowledge: Example Resources



Phase 4 - Knowledge Definition (Theory)

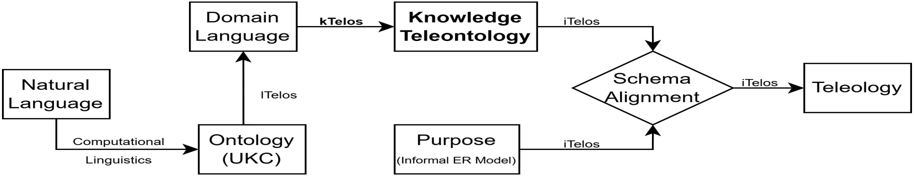
Dataset Alignment in Knowledge Definition Phase

- **Dataset Alignment**: For the data layer, the activity aims at aligning the dataset previously collected, cleaned and formatted, with the modelling choices decided and encoded in the knowledge teleontology and teleology.

- The reference knowledge teleontology considered for the final KG should represent the entity types in a similar structure (in terms of entity types, properties and data types) respect to their representation into each single datasets you considered for your KGE project.

Phase 4 - Knowledge Definition (Theory)

Need of Schema Alignment



- The above high-level process illustrates how:
  - ontology (UKC) is generated from natural languages
  - domain languages are designed using ontology (UKC)
  - knowledge teleontologies are generated using domain languages via kTelos
  - finally, **the purpose (informal ER model) is aligned to a knowledge teleontology, via iTelos, to generate the teleology** for final EG generation. **Without the schema alignment, a teleology cannot be generated. Therefore, the need of schema alignment in KGE**.

iTelos Schema Alignment

- The iTelos schema alignment approach (see: Paper) is as follows:
  - Given a knowledge teleontology, the main idea is "**properties can help to match/identify an entity type**" in a schema, so, in the KnowDive group, we utilize a machine learning-based matching approach wherein we exploit a property matcher and an entity type matcher
  - **Property alignment**: aligning the label of properties by lexical similarities, like n-gram, WordNet.
  - **Entity Type alignment**: We organize entity type alignment as a machine learning-based binary classification task. We use similarity metrics and also lexical similarities as attributes for training and testing our ML model.
- While we **don't execute** the above **schema alignment process** in this class, **it is key to automate the generation of Teleologies.**
- Some details about how we do schema alignment in the KGE course are provided in the next slides.

Schema Alignment in KGE

- We have the purpose already specified as an informal ER model.
- We formalize the informal ER model as an OWL file.
- We align the entity types of the above formal OWL file to their general entity types in the chosen knowledge teleontology (also in OWL) for the KGE purpose. This is a formalization of the EER model for the KGE purpose.

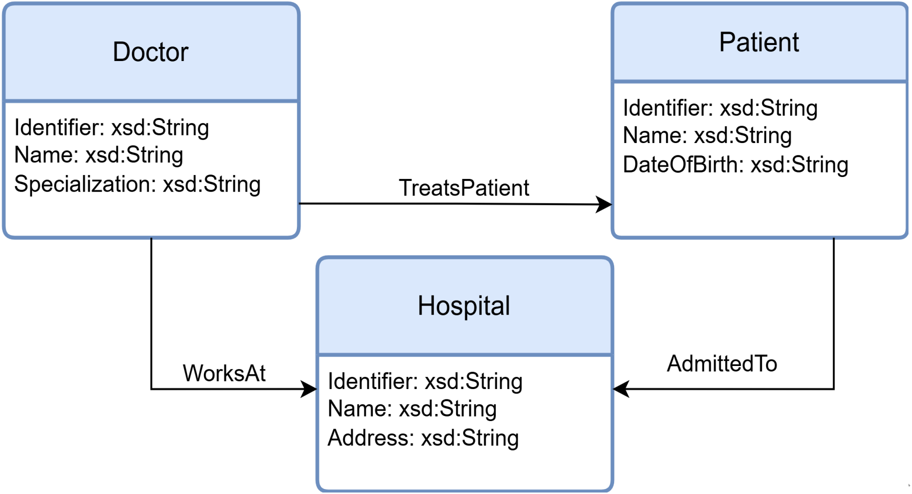
Schema Alignment in KGE (contd.)

- Finally, the **teleology** is produced as an OWL file by:
  - first, identifying only the leaf entity types for which we have data
  - second, dropping all the entity types more general to the leaf entity types
  - third, adding all the purpose-specific object and data properties of the general entity types to the leaf entity types (if applicable)
- **Revisiting and rechecking language definition**: In case any entity type, object property or data property are left without a unique UKC identifier, such a definition is achieved here.
- Next, we define the notion of a teleology in more detail and provide illustrations of an example schema alignment.

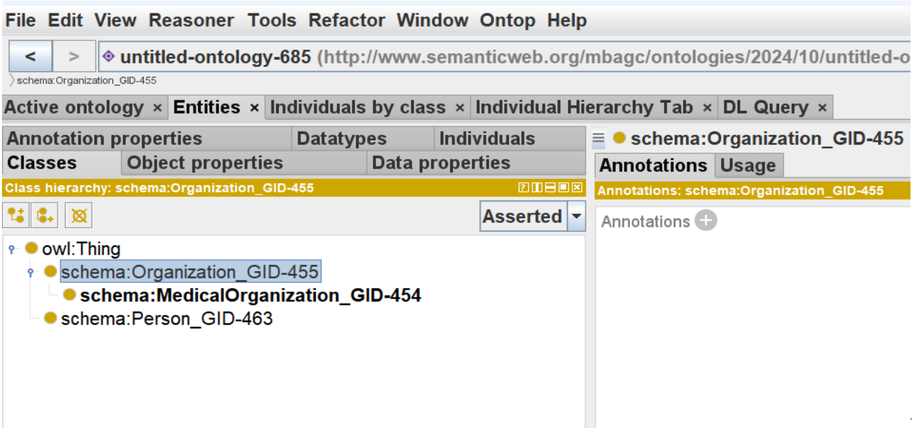
Teleology - Definition

- A **teleology** (see: ER 2017) focuses on purpose and on how a chosen representation fits a certain ***purpose***, this being the basis for a general model for the ***diversity of knowledge***.
- A teleology, therefore, makes **explicit the purpose** which it models via purpose-specific object and data properties.
- A teleology **does not encode a hierarchy** and is **flat** as the more general entity types are dropped.
- **A teleology should always be compliant to one or more knowledge teleontologies.**
- Notice also that we have **teleologies which are modelled as individual objects** whereas Entity Graph **schemas are embedded in entity graphs** and don't exist as independent objects.

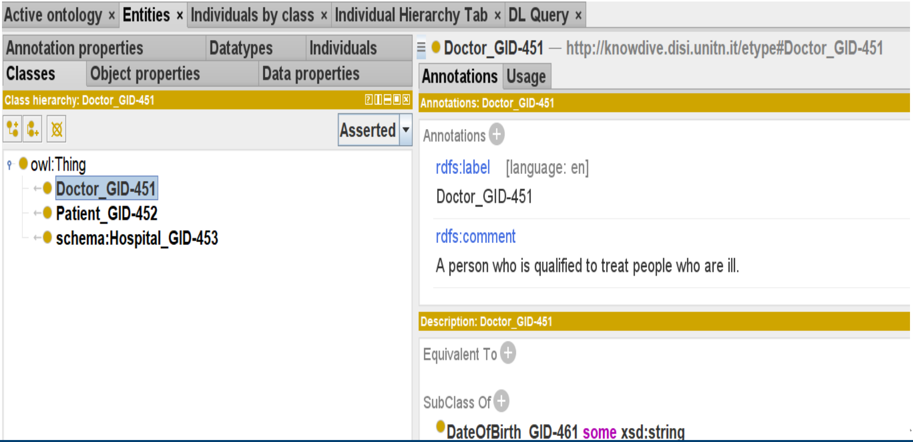
Schema Alignment Input: ER Model



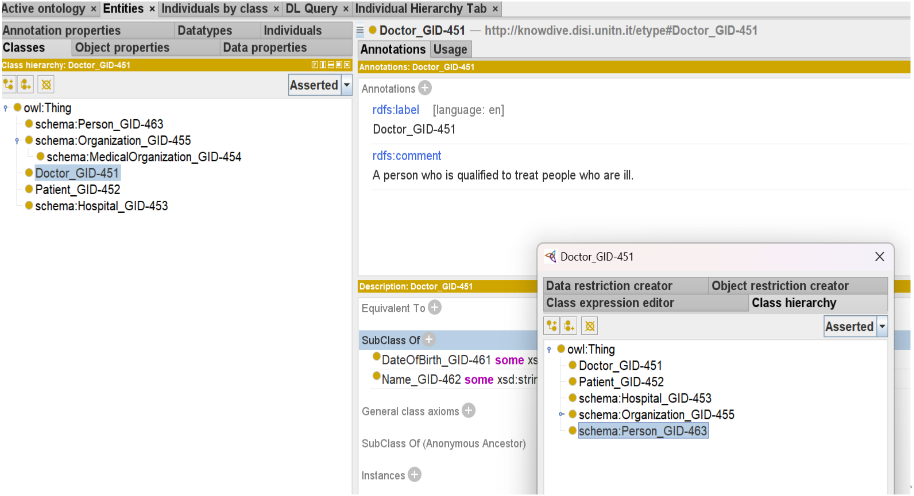
Alignment Input: Knowledge Teleontology



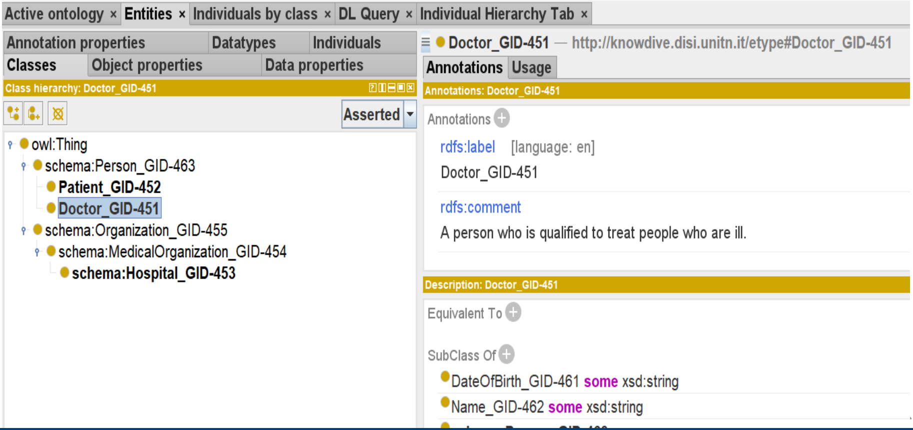
Schema Alignment Process: formalize ER Model



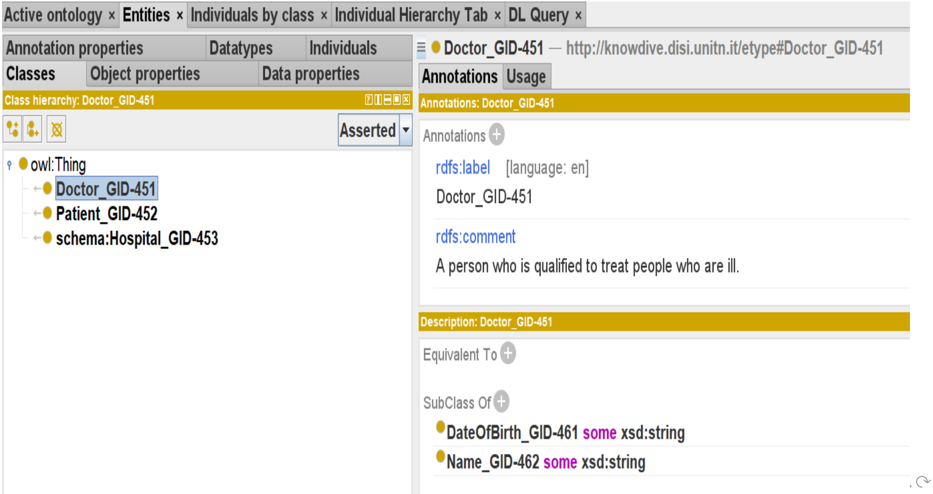
Schema Alignment Step



Schema aligned to Knowledge Teleontology



Teleology


Active ontology
Entities
Individuals by class
Individual Hierarchy Tab
DL Query
Annotation properties
Datatypes
Individuals
Classes
Object properties
Data properties
Class hierarchy: Doctor_GID-451
Asserted
owl:Thing
Doctor_GID-451
Patient_GID-452
schema:Hospital_GID-453
Doctor_GID-451 — http://knowdive.disi.unitn.it/etype#Doctor_GID-451
Annotations
Usage
Annotations: Doctor_GID-451
Annotations
rdfs:label [language: en]
Doctor_GID-451
rdfs:comment
A person who is qualified to treat people who are ill.
Description: Doctor_GID-451
Equivalent To
SubClass Of
DateOfBirth_GID-461 some xsd:string
Name_GID-462 some xsd:string

Revisit Language Definition

- Finally, each teleology concept: {entity type, object property and data property}, one at a time, is checked with the language resource sheet whether its UKC Global Identifier (GID) exist.
- There can be two cases:
  1. if the UKC GID exists in the language resource sheet, then check whether it is written in the teleology OWL file. If not, rewrite the GID as, e.g., **conceptname_GID-theactualGID**, e.g., **doctor_GID-451**, OR,
  2. if the UKC GID does not exist in the language resource sheet, then perform language definition for the concept and do the rewriting in the OWL file.