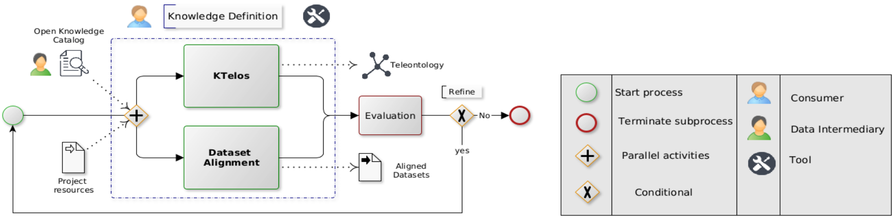
Part 4.5
Phase 4 - Knowledge Definition

Phase 4 - Knowledge Definition



- **Input**: the collected project resources, the formalized user's purpose and the data intermediary knowledge catalog.
- **Output**: the KG's teleontology (or a set for each KGs to be produced).
- **Objective**: the knowledge resources produced in this phase aims at:
  - unifying the representation of the information;
  - improving the **interoperability** and **reusability** of the final KG(s), by building knowledge resources reusing as mush as possible well-known standard domain ontologies and data schema.

Knowledge Definition - Activities

- In this phase, like in the others, the activities are divided over the knowledge and data layers.
- (knowledge layer) **KTelos**: In this activity the KTelos process is exploited to produce the final KG(s) **interoperable** ontology(ies). **Such ontology is produced by reuse!**
  - Reusing as much as possible the **standard reference domain ontologies** provided in input by the Intermediary Knowledge Catalog [43]
  - Merging the (portion of) reference domain ontologies with the **Purpose knowledge** formalized into the **ER model**.
  - The output of this activity will be interoperable due to the reuse of standard already existing knowledge, and purpose specific being based on the formalized purpose.
    - **How to choose whether to reuse ETypes or keep the Purpose's ones** ?

[43] The LiveKnowledge catalog is the main knowledge source but not the only one.

Knowledge Definition - Activities

- (data layer) **Dataset Alignment**: On the data layer this activity aims at aligning the dataset previously collected, cleaned and formatted, with the modelling choices operated in the above parallel knowledge layer activity.
  - **Dataset updates** based on the ETypes modelled in KTelos.
    - The unique ontology produced for the final KG could represent the ETypes in a different way respect to their representation into each single datasets.
  - **Data types alignment**.

Knowledge Definition - Producer & Consumer

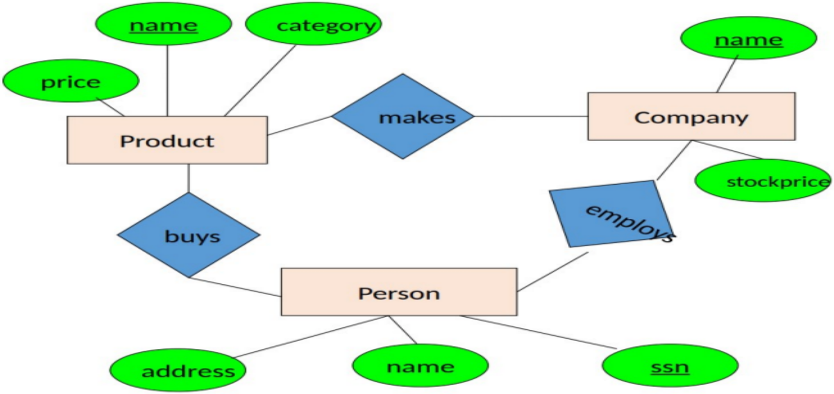
- **Producer**: at producer side the objective is to model **interoperable** (thus by reuse) **ontologies**, for each KG (thus for each datasets) to be created.
  - This means that **more ontologies files are produced**, one for each KG to be generated by the Producer.
- **Consumer**: at consumer side the objective is to model a single unique interoperable (thus by reuse) ontology, for the single composed final KG.
  - In this case a **single ontology file is produced**.

Phase 4 - Knowledge Definition

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

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**.

EER: Superclass/Subclass

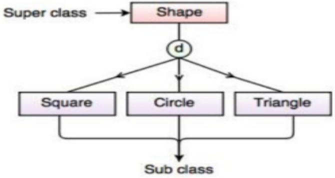


Fig. Super class/Sub class Relationship

Image Source

A superclass is a high-level entity that can be further segmented into subclasses or subsets. It is also referred to as a Parent class. For example, if Shape is considered an entity, then a Square, Circle, and Triangle are possible subclasses. A subclass can be referred to as a child or derived class. In this case, Shape is the superclass.

Note: 'd' stands for disjoint (subclasses).

EER: Characteristics

- 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.

E(E)R: Relationship Cardinality

CHEN notation

Cardinality – The number of entities to which another entity can be associated through a relationship

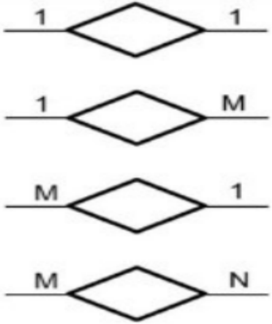
The diagrams on the right show, in order:

one-to-one

one-to-many

many-to-one

many-to-many

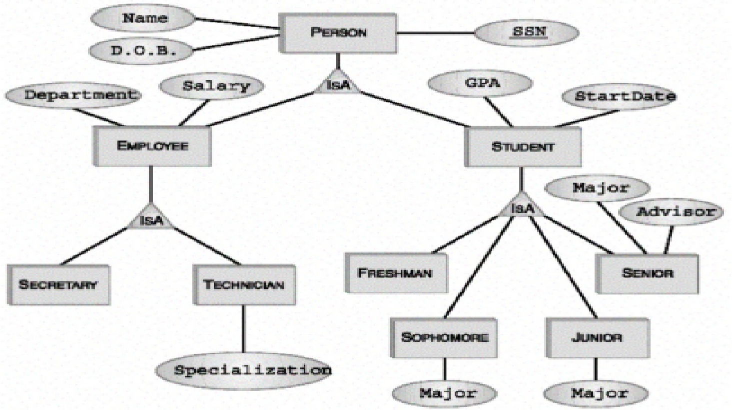


NOTE: CHEN notation is Peter PS. Chen's original ER diagram notation.

E(E)R: Cardinality Explained

- **One-to-One:** One entity from entity type X (e.g., 'Student Residence') can be associated (e.g., 'near') with one entity of entity type Y (e.g., 'Bus Stop').
- **One-to-Many:** One entity from entity type X (e.g., 'Bus Agency') can be associated (e.g., 'operates') with multiple entities of entity type Y (e.g., 'Bus Route').
- **Many-to-One:** Multiple entities from entity type X (e.g., 'Bus Stop') can be associated (e.g., 'spatialPartOf') with one entity of entity type Y (e.g., 'Bus Route').
- **Many-to-Many:** Multiple entities from entity type X can be associated with multiple entities of entity type Y, e.g., multiple students supervised by multiple faculty members, AND, multiple faculty members supervising mutiple students.

EER: Example



Reference: jcsites.juniata.edu

Phase 4 - Knowledge Definition

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.

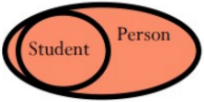
Why is SHARED of importance?

- 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

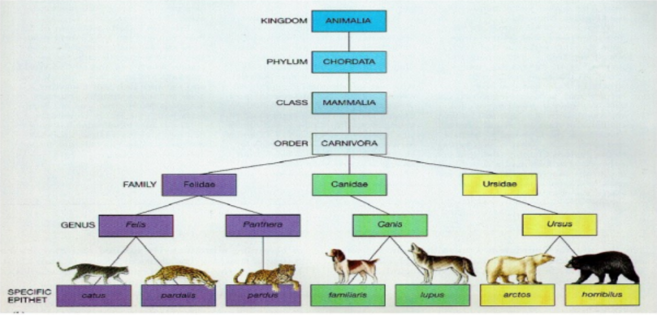
- **is-a relation**: binary relation between concepts (not individuals)
- Examples: Student is-a Person, Air Pollutant is-a Pollutant
  - Informal meaning: all the students are persons (or all the individuals that are students are also persons); if something is an air pollutant, itis also a pollutant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



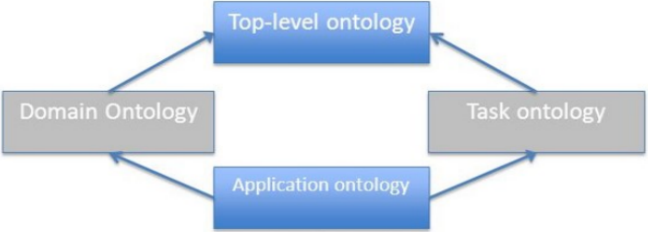
IS-A Hierarchy: Example Taxonomy

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.

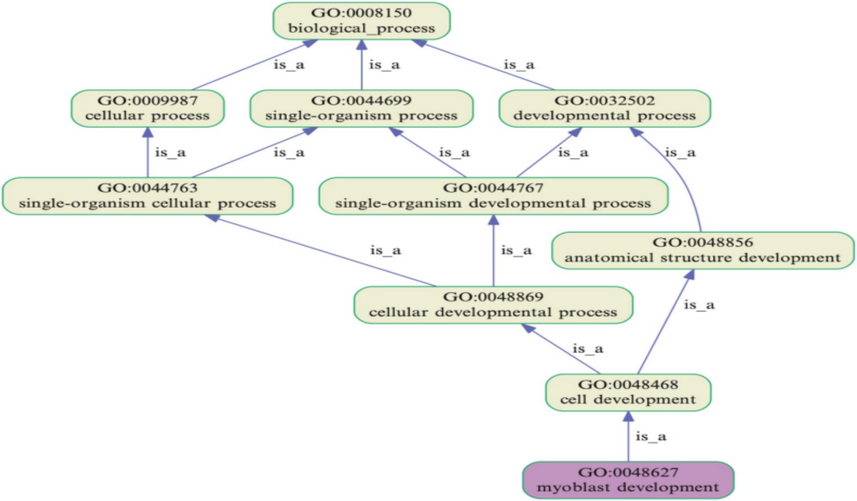
Domain Ontologies

- Domain ontologies are reusable in a given **specific domain** (medical, law, automobile, etc.).
- These ontologies provides **vocabularies** about concepts within a domain and their relationships, about the activities taking place in that domain.
- The concepts in domain ontologies are usually **specializations** of concepts already defined in top-level ontologies

Example: schema.org (fragment)

- Organization
  - Airline
  - Consortium
  - Corporation
  - EducationalOrganization
    - CollegeOrUniversity
    - ElementarySchool
    - HighSchool
    - MiddleSchool
    - Preschool
    - School
  - FundingScheme
  - GovernmentOrganization
  - LibrarySystem
  - LocalBusiness
    - AnimalShelter
    - ArchiveOrganization
    - AutomotiveBusiness
      - AutoBodyShop
      - AutoDealer

Example: The GENE Ontology



Domain ontology - Characteristics

- *Sharing common understanding* of the structure of information in the selected domain (among people or software agents)
- Enabling *reuse of domain knowledge*
- Making *explicit the domain assumptions* underlying an implementation. (key feature: It is thus easier to change these assumptions easily if our knowledge about the domain changes).

Ontologies - our own perspective

- The language of an ontology is written in a conceptual language which eliminates the ambiguites of NLP
- Given a language, an ontology is an entity type (etype) graph (ETG). An ETG is a set of etypes which associated with a set of objects and data properties (similarly to ER models)
- The ISA hierarchy is modeled as specialization hierarchy where more specific etypes inherit properties (similarly to EER models)
- Being designed with a more general purpose than a teleology (never made explicit in the SoA ontologies), an ontology provides a general view, and extra etypes and properties
- An ontology provides a more general view of the domain and extra etypes and properties (more general, but not only) which can be used to enrich a teleology, towards sharing and reusability.

Phase 4 - Knowledge Definition

1. Preliminaries definitions
   - ER Models (recap)
   - EER Model
   - Ontologies
   - WC3 Technologies & Tools (recap)
2. iTelos Knowledge Modelling
   - EER Models Limitations
   - Teleologies & Teleontologies
   - KTelos process
3. Dataset Alignment

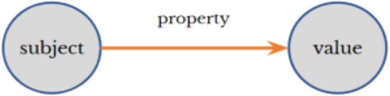
What is RDF?

- 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]

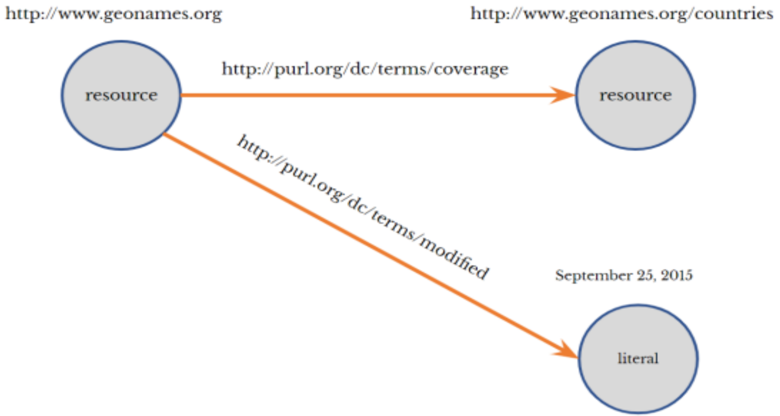
RDF language and data model

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
resource
http://purl.org/dc/terms/coverage
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

```
<rdf:RDF
```

HEADER

```
        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.

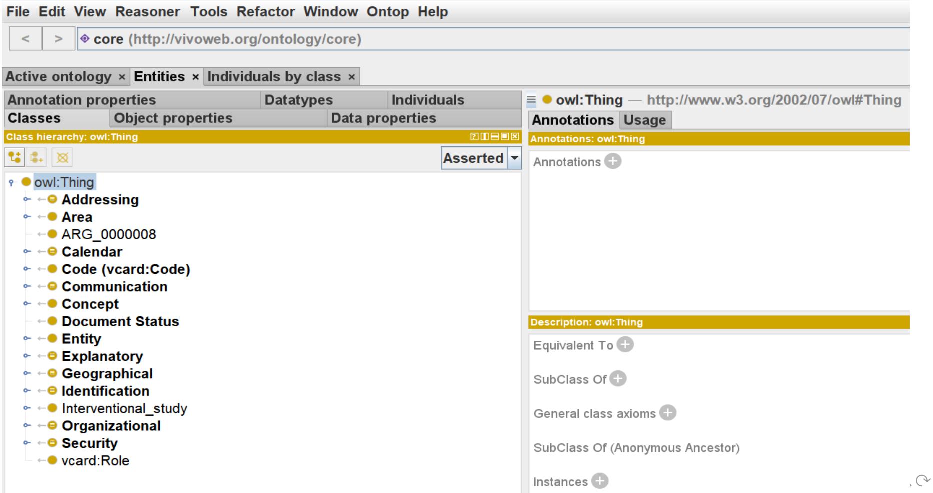
W3C STANDARDS SUPPORT

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

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

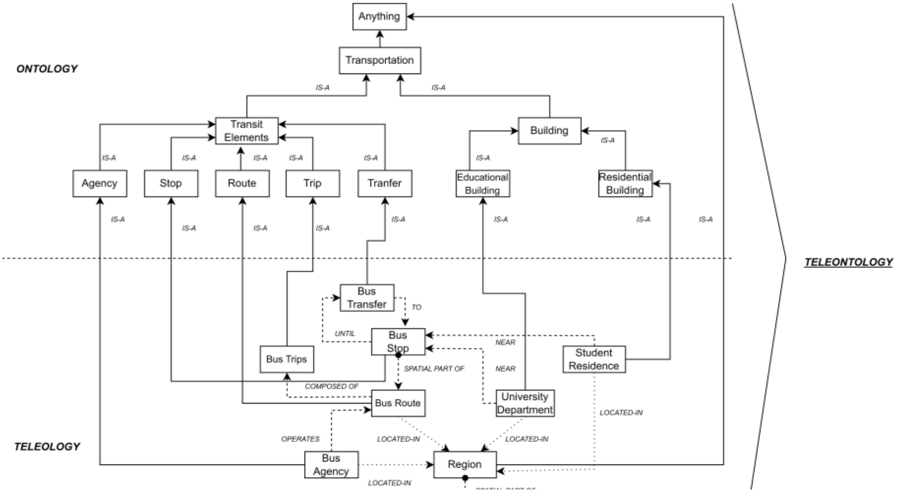
Teleology - Definition

- A **teleology** (see: ER 2017) is an ontology with the *proviso* that teleologies focus on function 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 spatio-temporal context which it models.
- This results in explication of the underlying design assumptions of a teleology and thereby makes it flexible for data and knowledge reuse and integration.
- The explicit split between constructing teleologies vs. how it is semantically aligned to more general etypes (in summary, modelling teleontologies) allows for large scale data and knowledge reuse.

Teleontology - Definition

- **Teleontology:** A teleontology is a teleology extended with more etypes which grounds it to what exists in the world around us. Given a teleology generated according to a purpose, a reference ontology is selected/designed to which the teleological concepts are aligned. Concretely axiomatized as an OWL RDF/XML file.
- It is, thus, a graph encoding **IS-A**, **SPATIAL PART OF**, **LOCATED IN** relations + functions and actions (in the form of object and data properties) amongst the potential concepts to be modelled.
- A teleontology models the specific concepts of a domain characterized by their (object and data)properties via the teleology.
- While at the same time, it semantically constrains the ontological meaning of the concepts in the teleology via their link to the chosen reference ontology.

Example Illustration

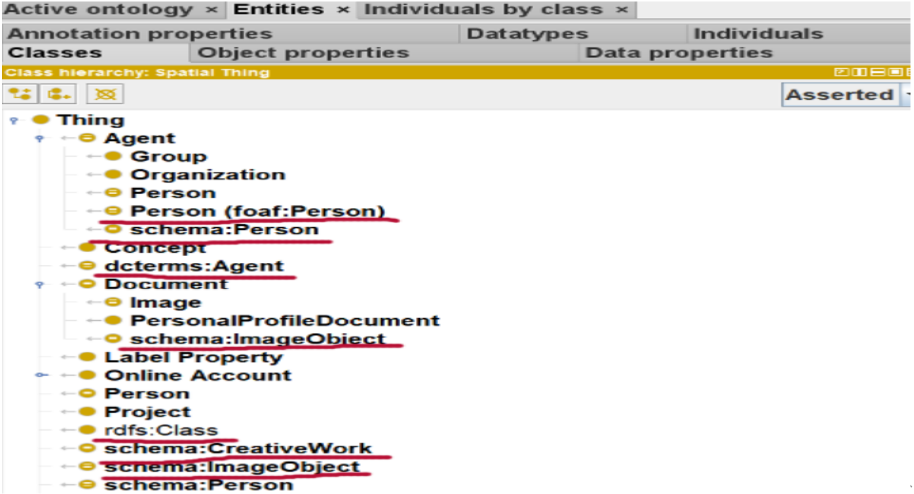


Phase 4 - Knowledge Definition

kTelos Process

- The concrete steps for modelling knowledge as teleologies and teleontologies is via the **kTelos** process. It is as follows:
  1. **Top-Down:** reuse of a Lightweight Ontology (aligned to the UKC)
  2. **Bottom-Up:** modelling of a Teleology (aligned to the requirements modelled as CQs)
  3. **Middle-Out:** aligning of a Teleology grounded into the Lightweight Ontology to generate a Teleontology.
  4. Finally, the **knowledge annotation** of a Teleontology.
- **Note:** All the above knowledge artefacts can **reuse concepts** from existing knowledge resources.

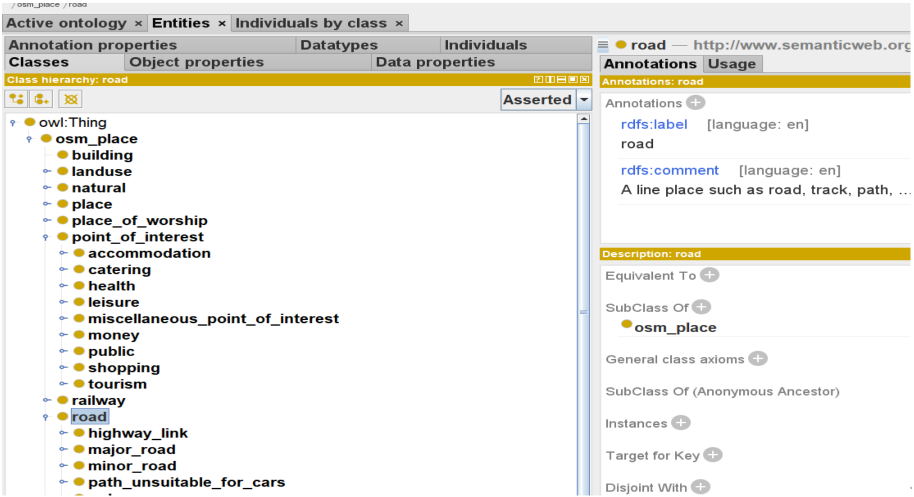
Reuse General Example: FOAF



Top Down: Lightweight Ontology

- A lightweight ontology is an ontology which is modelled, top-down, as *"consisting of backbone taxonomies only"* (see: Paper).
- It is modelled in alignment with the hierarchy of the UKC.
- The hierarchy of concepts in lightweight ontologies is modelled via the **subset of** relation. The child node is a subset of the parent superset.
- The chief objective of a lightweight ontology is to heirarchically classify the datasets which would be finally integrated in the Entity Graph (EG).
- A lightweight ontology file (in `OWL RDF/XML` format) on OpenStreetMaps (OSM) is provided to you for reuse.

Lightweight Ontology Example



Bottom-Up: Teleology Modelling

- A teleology is modelled, bottom up, starting with a tabulated list of **Competency Questions (CQs)** which encode the etypes and properties relevant to be modelled.
- Out of the CQs, the etypes and properties are modelled into an **E(E)R diagram**.
- Next, the E(E)R diagram is formalized as a formal **schema** (e.g., in `OWL RDF/XML`). This might have certain application-specific attributes.
- Finally, the **teleology** is produced (e.g., in `OWL RDF/XML`). It might be the same as the formal schema above, or without application-specific attributes, decided on a case-to-case basis.

Example: CQs

In the scenarios defined above, we represents a set of real users with specific features included in the project purpose, whhich are listed as follows:

- **Personas 1.** Lily, a 60-year-old woman living in Obergummer, Rovereto, is an outdoor enthusiast. She loves to explore the natural attractions of Trentino.
- **Personas 2.** Anna, a 28-year-old doctoral student, plays a key role in her research group. She is responsible for organizing various activities for the group.
- **Personas 3.** Luca, a 25-year-old master student lives around the Trento train station of the center of Trento, he has a passion for cooking and tasting food.

Taking into account the personas in the scenarios defined, we create Competency Questions (CQs):

- **CQ 1.** Please recommend Lily the nearest 3 peaks from her home.
- **CQ 2.** Can you provide the roads information that Lily can drive to the 3 peak destinations?
- **CQ 3.** On Monday, Anna wants to find an opening restaurant near DISI for the party of her research group. For the transportation convenience of students, this restaurant should near at least one bus stop.
- **CQ 4.** Luca is looking for a library near his home that is easily accessible by bus stops and surrounded by restaurants.
- **CQ 5.** Luca wants to shop in the supermarket that is nearest from his home.

Example: ER Model

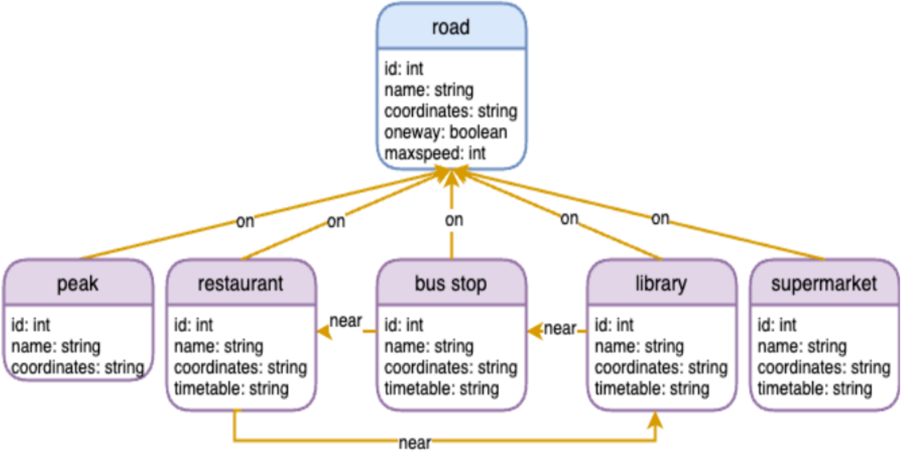
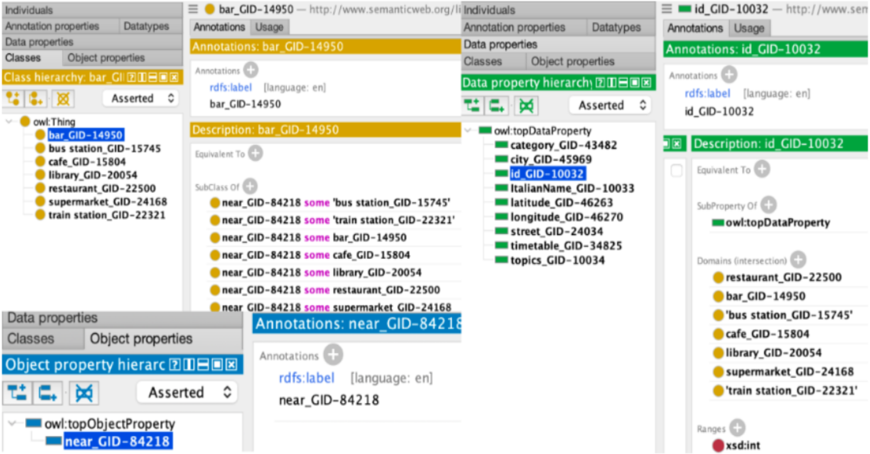


Figure 1: ER model.

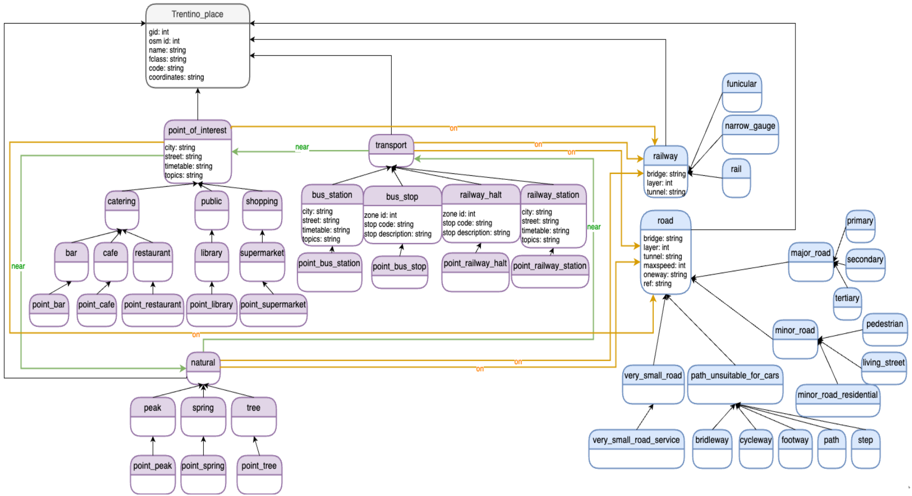
Example: Teleology



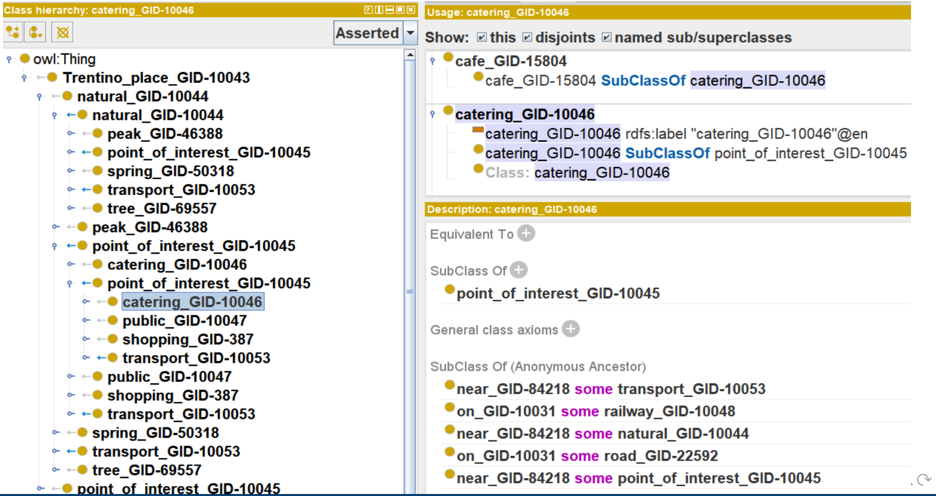
Middle-Out: Teleontology Modelling

- In this step, the teleology is semantically aligned to the Lightweight Ontology to form a Teleontology.
- Semantic Alignment: each concept in the teleology (e.g., *Professor*) is added as a child (via *IS-A*) to their related general concept (e.g., *Person*) in the lightweight ontology.
- The above process is done, one at a time, for all the concepts in the teleology.
- Finally, the teleontology is produced, e.g., in `OWL RDF/XML`).

Example: Teleontology



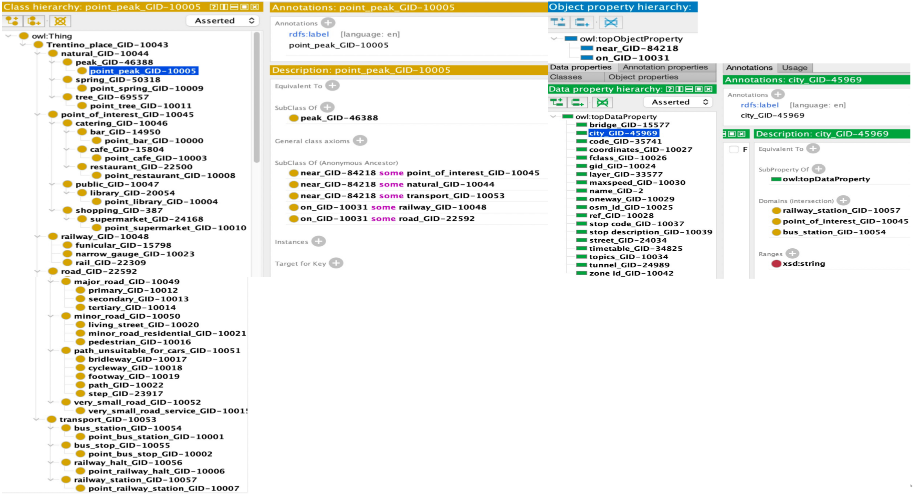
Example: Teleontology in OWL RDF/XML



Knowledge Annotation

- Finally, each teleontology concept: {entity type, object property and data property}, one at a time, is checked with the Language Annotation spreadsheet to identify its GID.
- Once the GID is identified, the concept in Protégé should be updated with the GID and re-written.
- There can be two syntactical cases of the re-writing:
  1. if concept exists in Language Annotation spreadsheet, then the re-writing syntax would be, e.g., `transport_GID-10053`, OR,
  2. if you add a new concept in the teleontology from another reference standard/ontology (e.g., from FHIR), the syntax would be, e.g., `fhir_hospital`.

Knowledge Annotation Example



Phase 4 - Knowledge Definition

Datasets Alignment Activity

- (data layer) **Dataset Alignment**: On the data layer this activity aims at aligning the dataset previously collected, cleaned and formatted, with the modelling choices operated in the above parallel knowledge layer activity.
  - **Dataset updates** based on the ETypes modelled in KTelos.
    - The unique ontology produced for the final KG could represent the ETypes in a different way respect to their representation into each single datasets.
  - **Data types alignment**.

Phase 4 - Knowledge Definition - Summary

- What has been done in this phase.
- The **heterogeneity at knowledge level** has been handled.
  - By modelling a unique representation of the information in the final KG(s), through a (a set of) Teleology(ies).
- The output Teleology(ies) are **interoperable** and **reusable** for other purposes (uploaded in the knowledge catalog), thus actually **reducing the effort in knowledge modelling** for further projects.
- The dataset, containing the Entities modelled in the Teleology(ies) through their ETypes, have been aligned with such ETypes, with the objective of **facilitating the mapping between data and knowledge layer** (iTelos Data Definition Phase).