TALOS ERA CHAIR IN ARTIFICIAL INTELLIGENCE FOR HUMANITIES AND SOCIAL SCIENCES

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
UNIVERSITY OF CRETE

Protégé

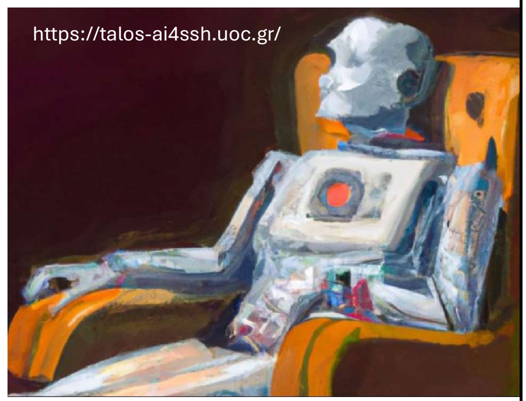
Ontology Building Environment

Prof Christophe Roche

ERA Chair Holder - University of Crete

https://christophe-roche.fr/

May 2025

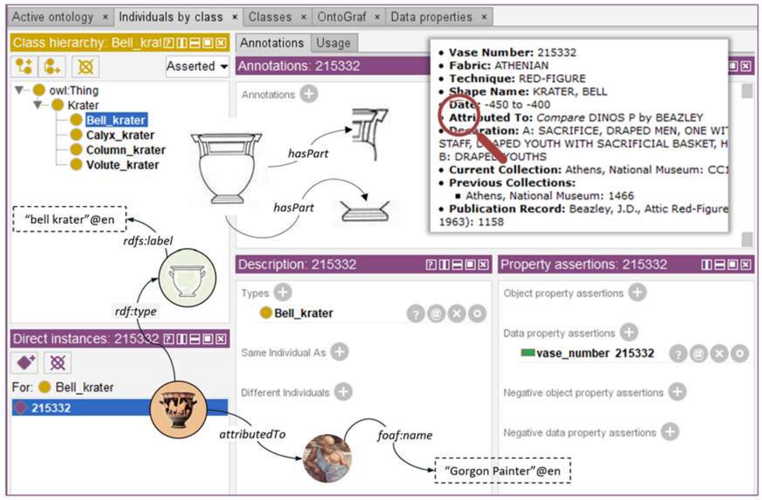


Horizon ERA Chair TALOS AI4SSH Project funded by the European Commission
Grant Agreement n° 101087269, https://cordis.europa.eu/project/id/101087269

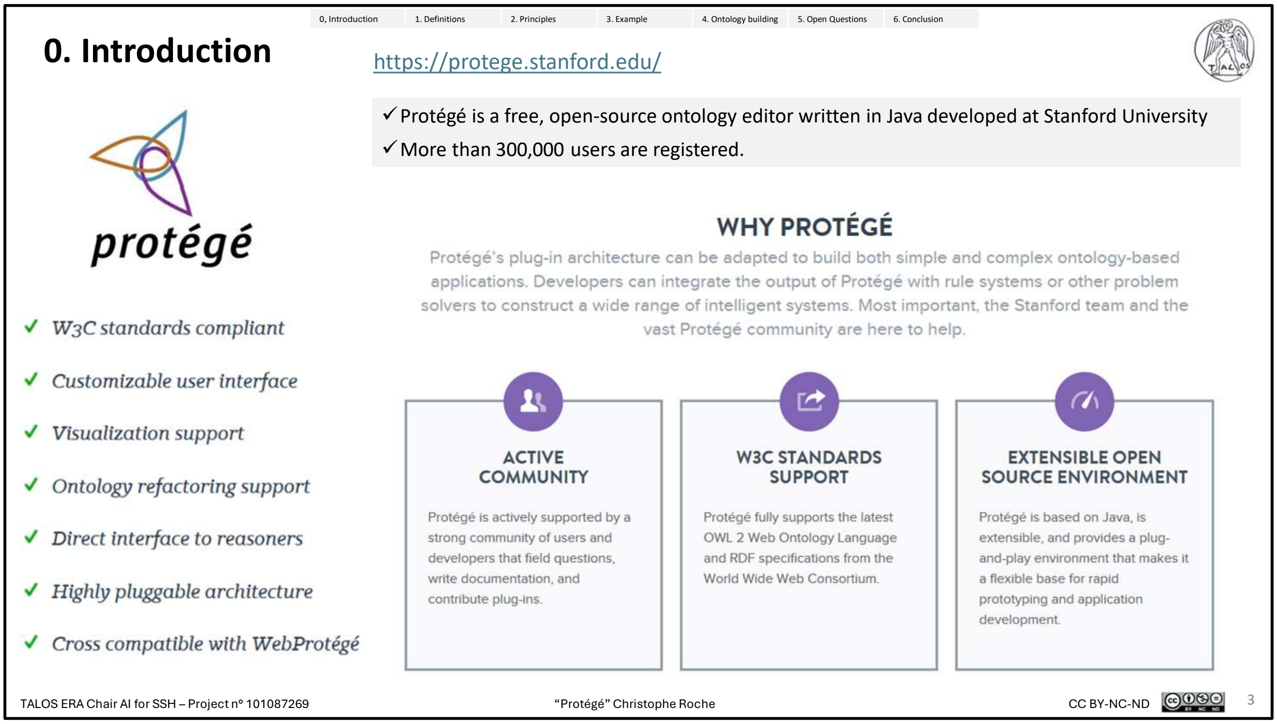
Funded by the European Union

This MOOC is dedicated to Protégé, the currently most widely used ontology building environment.

Contents



The MOOC is divided into 6 parts. We'll begin by recalling some definitions, followed by the theoretical principles on which Protégé is based. We'll then see how to use Protégé to build an ontology with an example from Digital Humanities. We'll conclude with two open questions.



Protégé is the most widely used ontology-building environment. Written in Java, it is a free, open-source software developed by Stanford University. It is supported by a large community of users. Protégé is based on the first-order logic that allows the use of reasoners to verify the consistency of ontologies. Protégé supports W3C formats such as OWL, which allows ontologies to be exported as RDF knowledge graphs.

1. Definitions

Ontology

An ontology is a formally-defined vocabulary for a particular domain of interest. Ontologies are typically based on a class hierarchy (asserted and/or inferred), supplemented by assorted properties.

https://protegewiki.stanford.edu/wiki/Pr4_UG_mi_Glossary#Ontology

☞ **Ontologies are used to capture knowledge about some domain of interest. An ontology describes the concepts in the domain and also the relationships that hold between those concepts.**

A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools - Edition 1.3

Language

OWL provides the theoretical basis for Protégé ontologies.

https://protegewiki.stanford.edu/wiki/Pr4_UG_mi_Glossary#Ontology

Different ontology languages provide different facilities. The most recent development in standard ontology languages is OWL from the World Wide Web Consortium (W3C).

A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools - Edition 1.3

Reasoner

The logical model allows the use of a reasoner which can check whether or not all of the statements and definitions in the ontology are mutually consistent and can also recognise which concepts fit under which definitions. The reasoner can therefore help to maintain the hierarchy correctly.

A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools - Edition 1.3

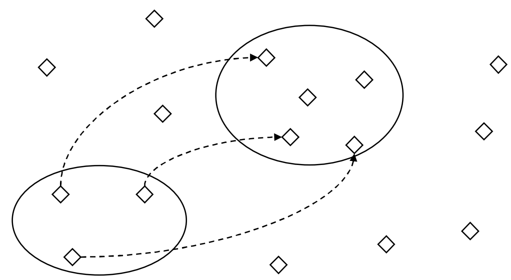
☞ **an inconsistent class is a class which cannot contain any individual because of its definition**

Let's start by recalling what an ontology is. Some define an ontology as a formally defined vocabulary. But it is above all a specification of a conceptualization used to represent the knowledge of a domain. An ontology defines the concepts of a domain and the relationships between these concepts.
These definitions are written using a knowledge representation language, including the W3C standard Ontology Web Language (OWL).
Logical foundations allow the use of reasoners to verify, for example, the consistency of an ontology.

2. Principles

Extensional Logic

Goal: **Organising** the objects which populate the world into **classes** according to the **relationships** that linked objects together



⇨ An object is not defined by its “nature“, but by its relations with other objects

To master a tool, whatever it is, it is mandatory to master the principles on which it is based. Protégé is based on an extensional logic, that is, on manipulating objects, more generally called individuals. The goal is to organise the objects that populate a reality into sets, called classes, according to the relationships linking the objects to each other. The notion of Class replaces that of Concept.
Thus, an object is not defined by its nature, but by its relationships with other objects.

2. Principles

Components of OWL Ontologies

1) Individuals

Individuals, represent objects in the domain in which we are interested

Terminology: *« individual », « instance », « object »*

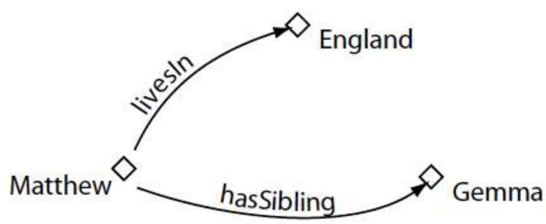
The first principle is therefore the notion of individual, or object, even if it is not necessary to create individuals to define classes. Individuals can represent any object in the field of application.

2. Principles

Components of OWL Ontologies

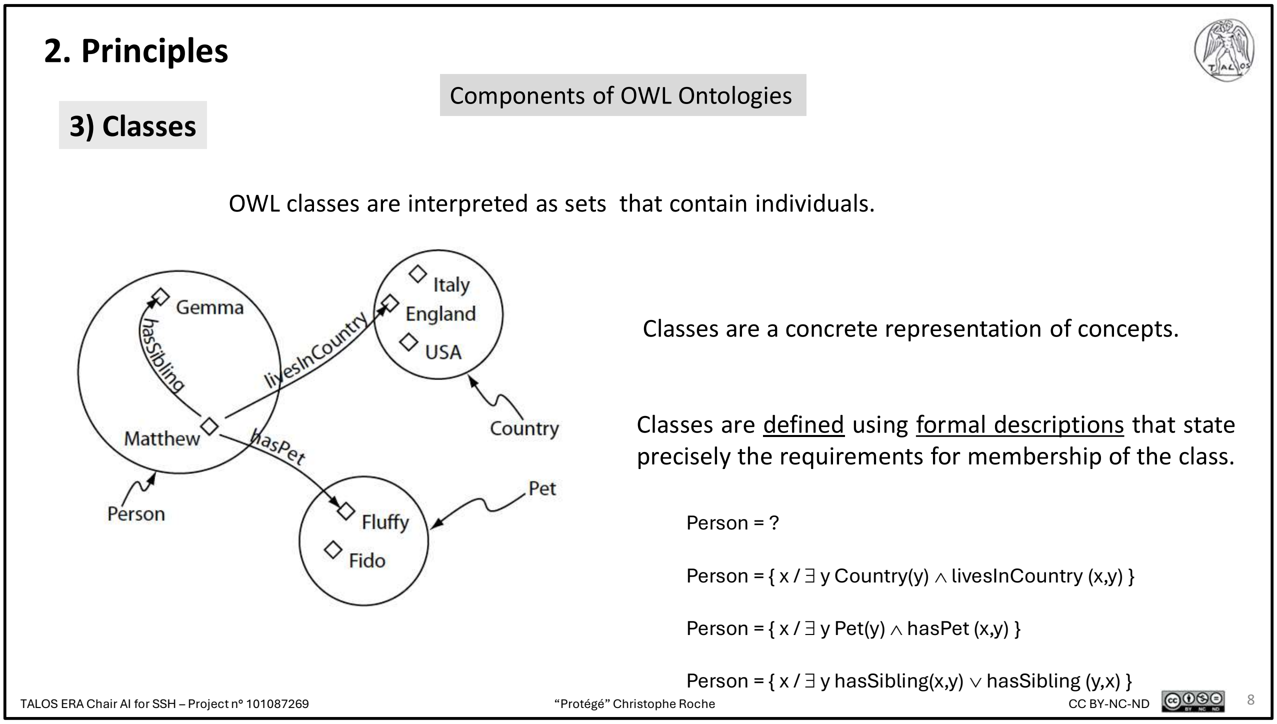
2) Object Properties

Properties are binary relations on individuals, i.e. properties link two individuals together.



Terminology: « properties », « slots » (Protégé), « roles » (DL), « relations », « attributes »

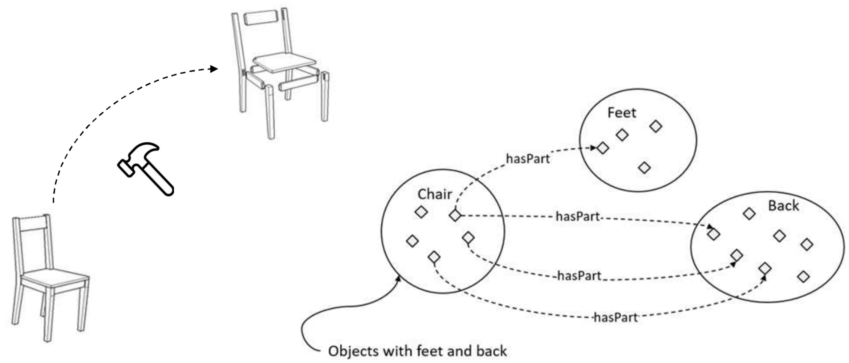
The second principle is that of relation or property. Properties are binary relationships between two individuals. The same individual can, as in this example, be linked to several individuals, whether through the same relationship or different relationships. Relationships are always oriented and named.



It is then possible to define classes, sets of individuals, according to the relationships linking individuals. So, in this example, a person is not defined according to his nature but according to his relationships. It is then possible to propose different definitions of the Person class. These definitions are not necessarily equivalent.

2. Principles

Change your way of thinking

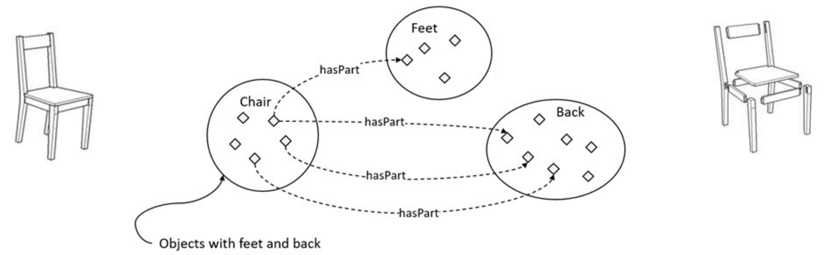


This approach to definition requires changing the way we think about things. A thing is no longer a whole, but a set of related entities. Some domains are well-suited to this approach. For example, a chair is an individual linked to its parts, its backrest, its legs, and so on.

2. Principles

4) Property restriction A means to define classes of individuals

a) **Existential Restrictions**: describes (anonymous) classes of individuals that participate in *at least one* (some) relationship along a specified property to individuals that are members of a specified class.



b) **Universal Restrictions**: describes (anonymous) classes of individuals that for a given property *only* (only) have relationships along this property to individuals that are members of a specified class (all values of the property must be of a certain type)

c) **Has value**: at least one of the values of the property is a certain value

A class is then defined as a set of property restrictions, that is, restrictions on the objects that can be linked by the property. Restricting the hasPart relationship to the Back Class allows to define the class of individuals which have a Back of which the Chair Class is a subset.

0, Introduction | 1. Definitions | 2. Principles | 3. Example | 4. Ontology building | 5. Open Questions | 6. Conclusion

3. Example

CLASSICAL ART RESEARCH CENTRE

UNIVERSITY OF OXFORD

Beazley Archive: https://www.carc.ox.ac.uk/carc/Home

The Krater Ontology

“The term '**krater**' suggests a mixing-vessel (compare Greek kerannumi - to mix), and we know that the wine served at the symposium was mixed with water.

Column-krater: Named for its column-like handles, the column-krater is first known from Corinthian examples dated to the late seventh century. It is regularly produced by Athenian potters from the first half of the sixth-century until the third quarter of the fifth. It seems from graffiti on Athenian red-figure examples that the vessel was referred to as *Korinthios* or *Korinthiourges*.

Volute-krater: The volute-krater is named after its handles. The François Vase is a famous and early example, but the typical Athenian form occurs only later in the sixth century, with the handles tightly curled so that they look like the volutes on Ionic columns. The shape is also found in metal. Over the course of the fifth and fourth centuries, examples become slimmer, and Apulian volute-kraters from South Italy are particularly elaborate.

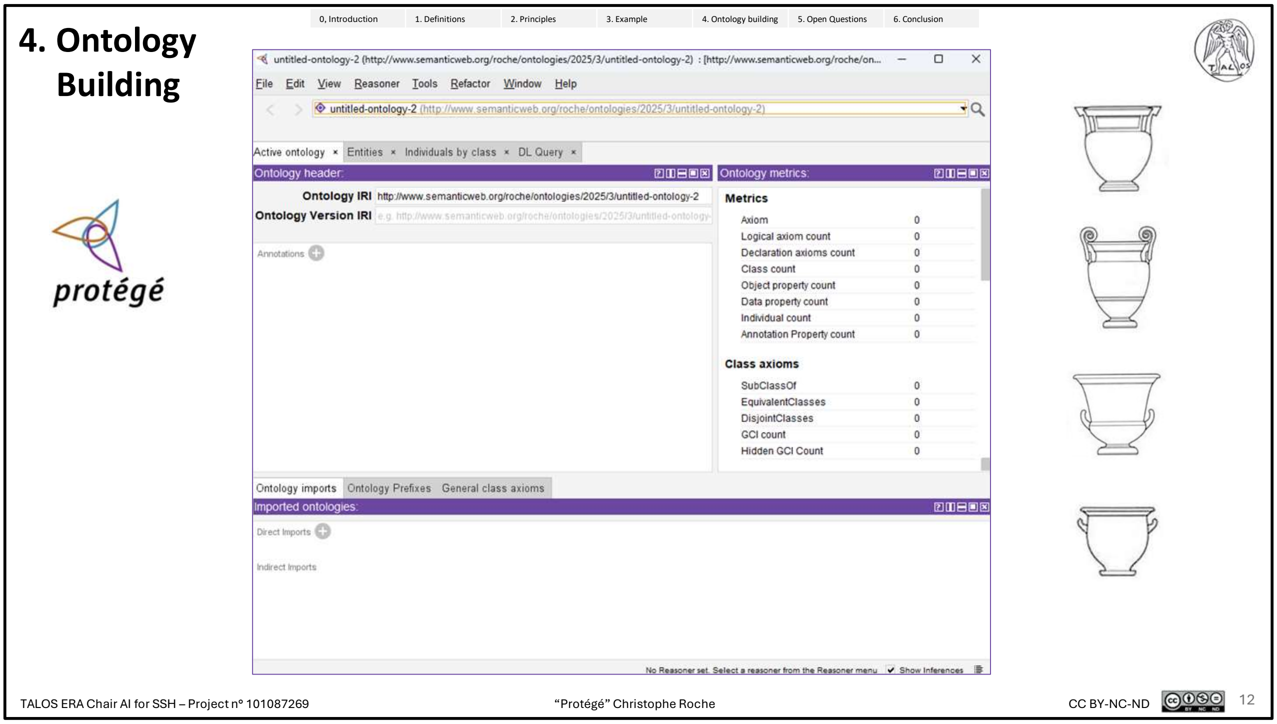
Calyx-krater: The handles of the calyx-krater are placed low down on the body, at what is termed the cul. Their upward curling form lends the shape an appearance reminiscent of the calyx of a flower, hence the name. The earliest known example was possibly made by Exekias in the third quarter of the sixth century. It continues to be produced, mainly in red-figure, becoming more elongated over the course of the fifth and fourth centuries.

Bell-krater: The latest of the four krater-types, it first occurs in the early fifth century, and is not found decorated in black-figure. It is named for its bell-like shape, perhaps originating in wood. It has small horizontal upturned handles just over halfway up the body. Some do not have a foot, and earlier examples may have lugs for handles.

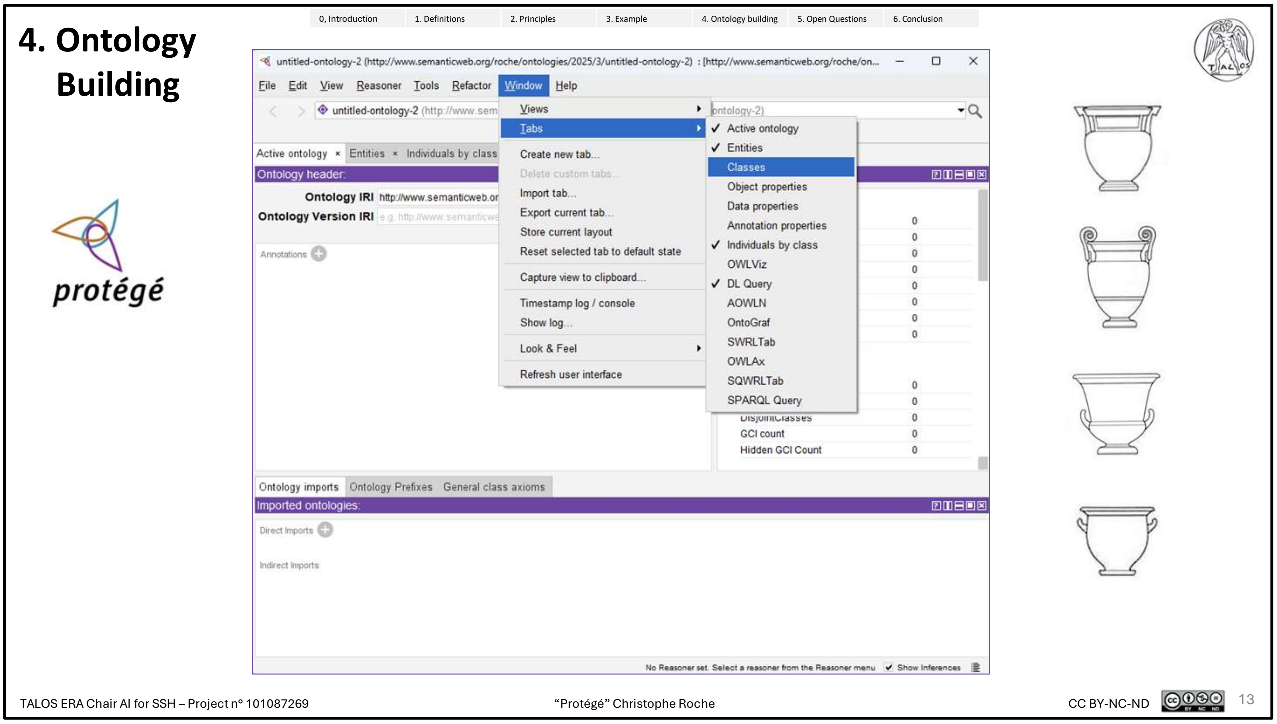
TALOS ERA Chair AI for SSH – Project nº 101087269 “Protégé” Christophe Roche

 11

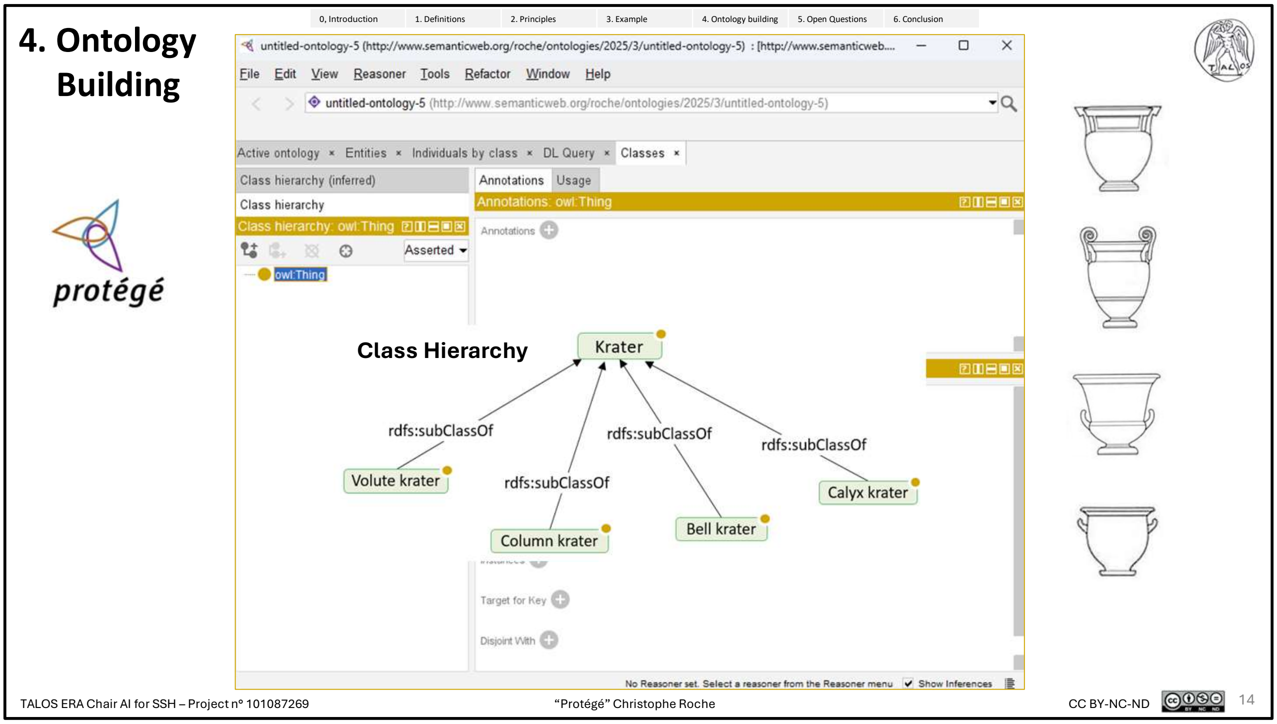
Let us take for example the definition in Protégé of the ontology of Greek vases, and more precisely, the ontology of kraters as defined in the Beazley archive. A krater is an ancient Greek vessel used to mix water and wine. There are different types of kraters: column, volute, calyx, bell craters, etc. These types of kraters are distinguished by the type of their handles or their shape.



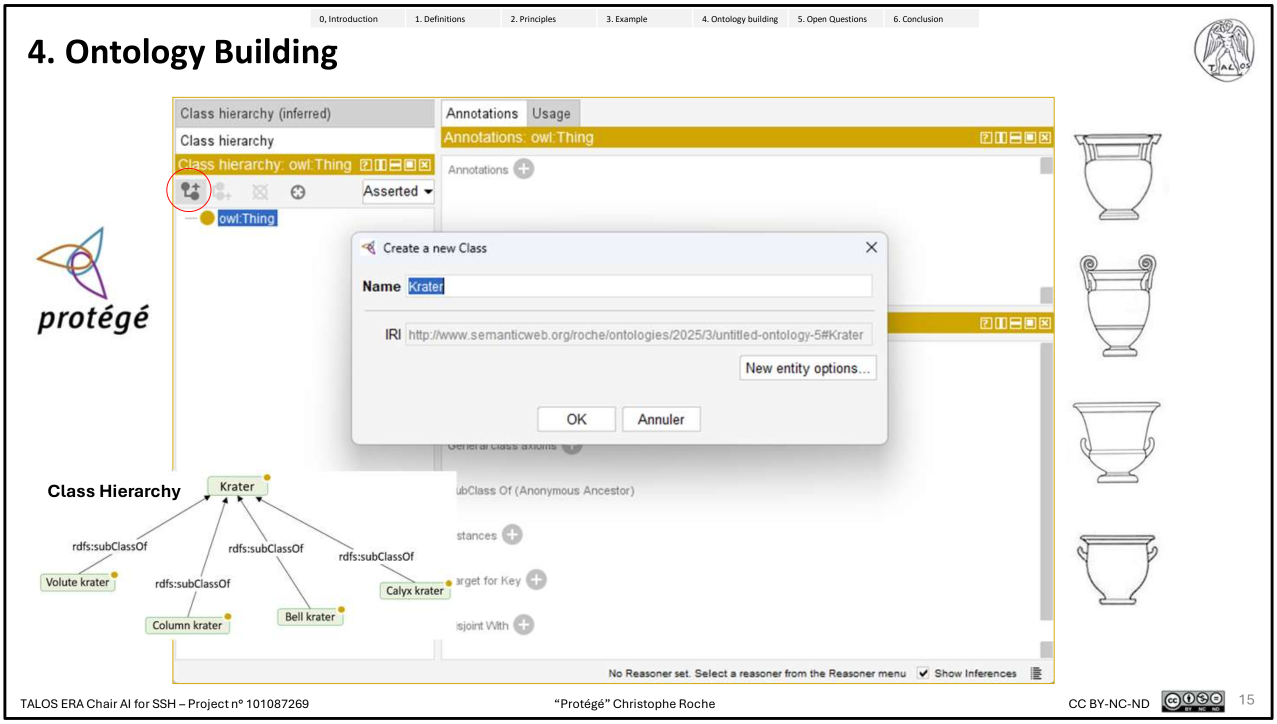
When Protégé is opened, a window is displayed with different tabs. Each tab is divided into views. The first tab named 'Active ontology' lists information about the ontology in use such as the number of classes, individuals, properties, etc.



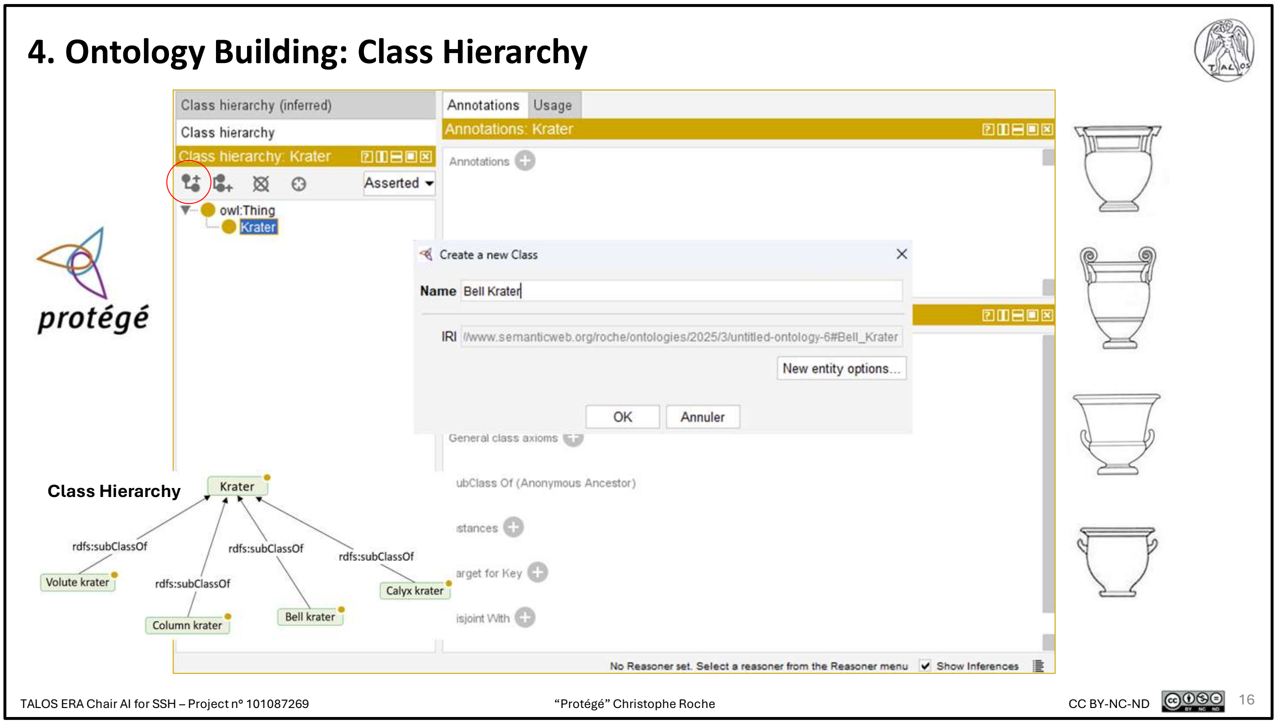
The definition of classes is done using a dedicated tab that must be created, as illustrated in the figure of slide 13



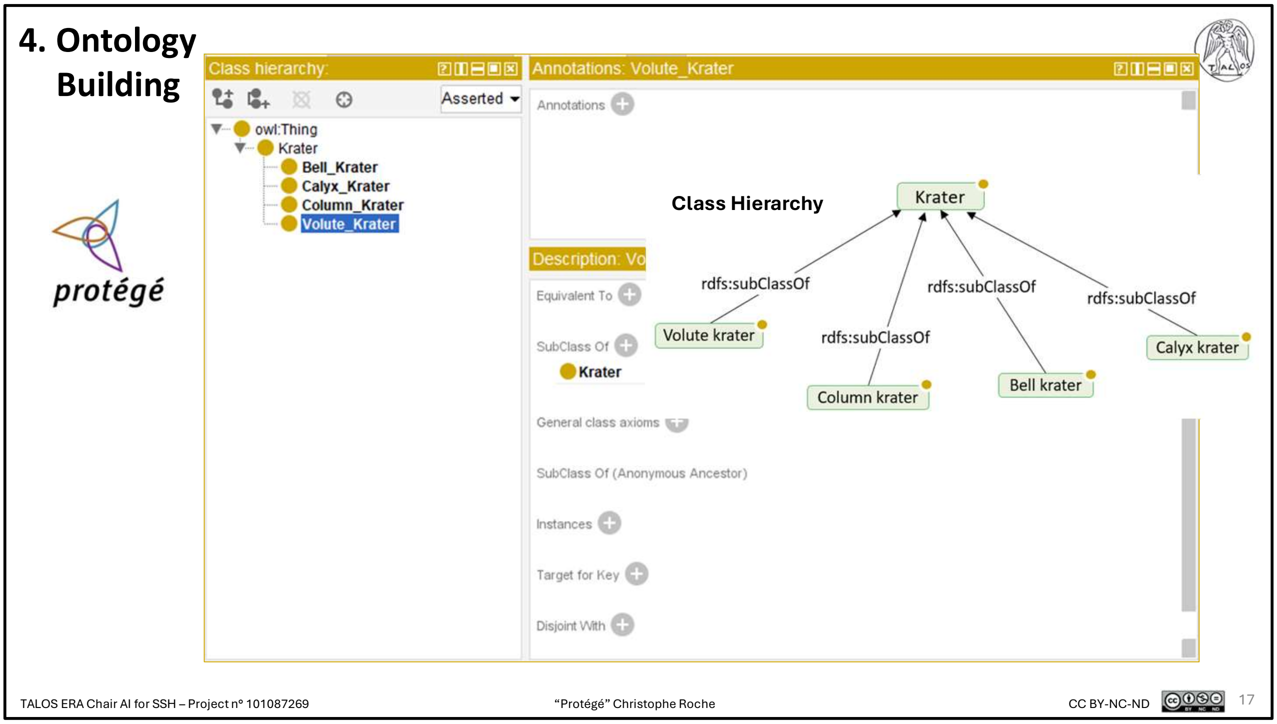
The ontology of kraters is a set of classes organized in a hierarchy according to the set relationship of inclusion: column krater is a subclass of the class krater.



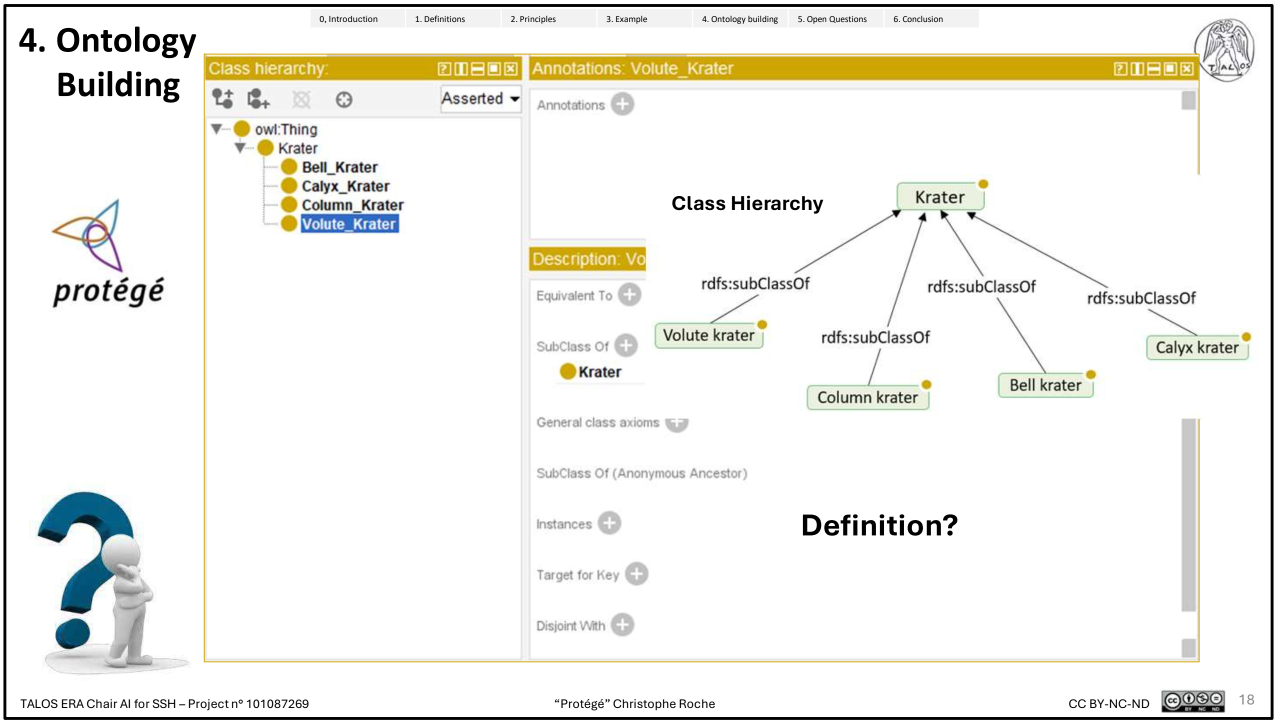
Every class is a subclass of the Thing root class, which groups all objects. The creation of the Krater class is therefore done from the Thing root class.



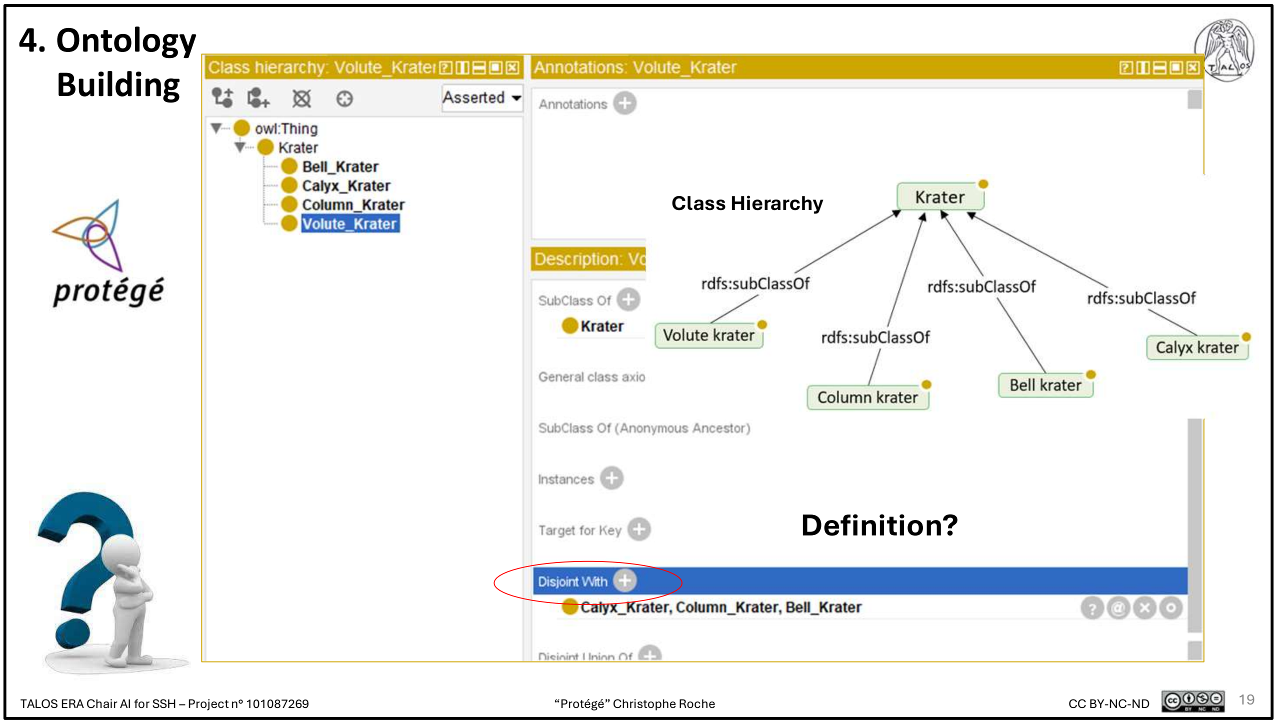
The different types of Kraters are subclasses of the Krater class. They are created in a similar way to what we have just seen.



The result is the class hierarchy of slide 17



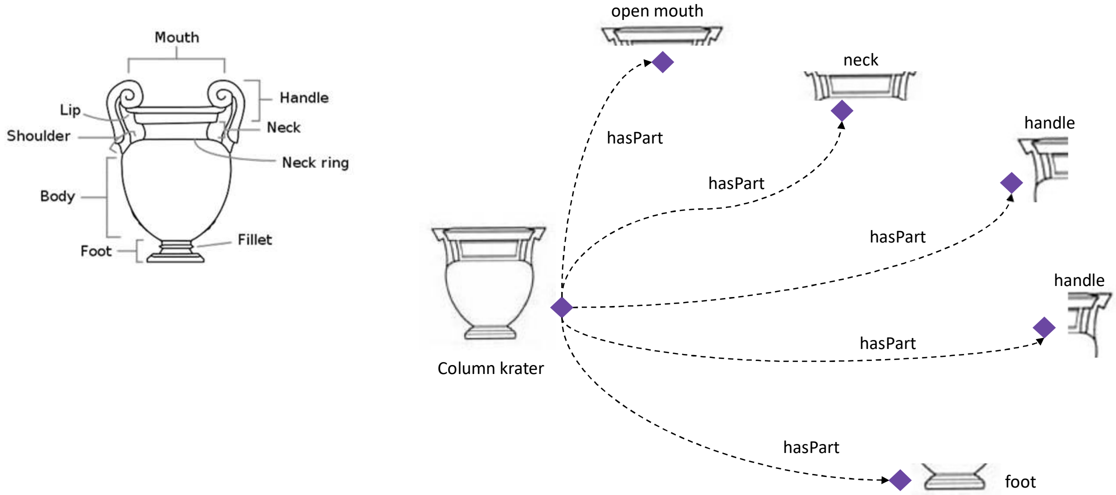
Defining the different types of kraters as subclasses of the class krater does not define them. Indeed, nothing distinguishes, except their name, the different types of kraters. They still need to be defined more precisely.



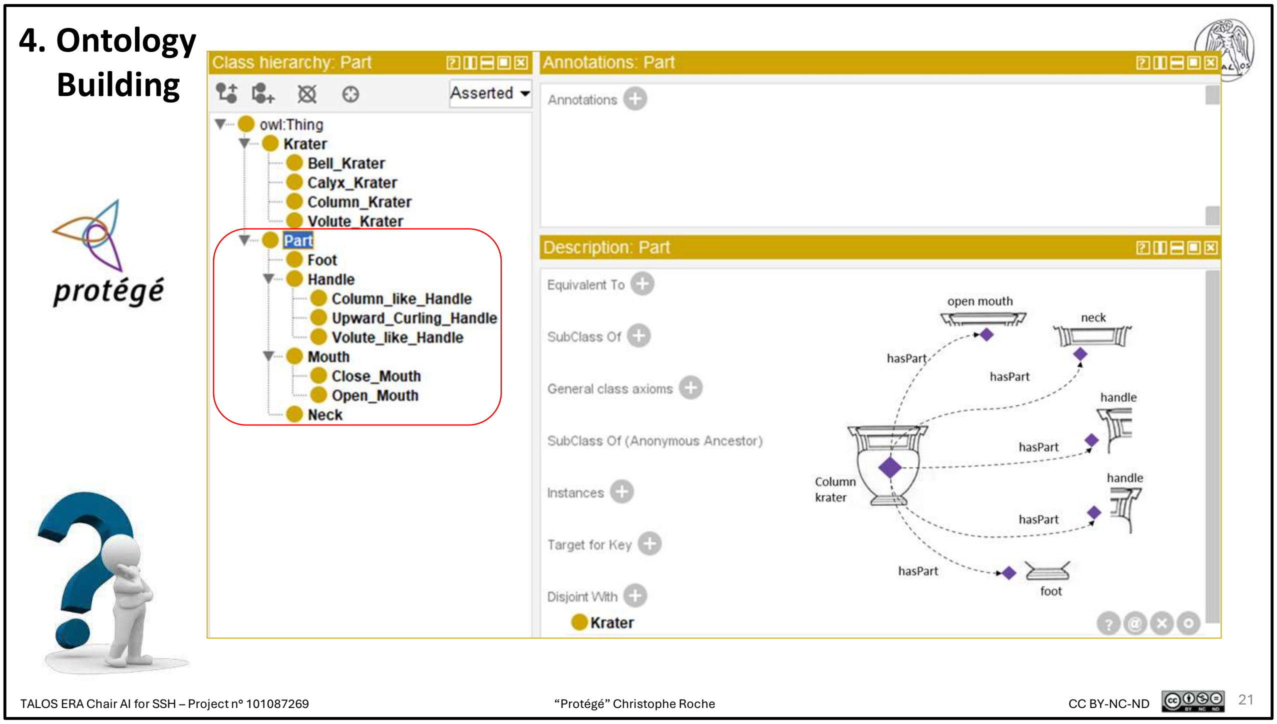
A first property is to declare the different classes of kraters as disjoint: a given krater can only be of one type and therefore belong to only one class. But that still doesn't define what a particular krater is.

Object Properties

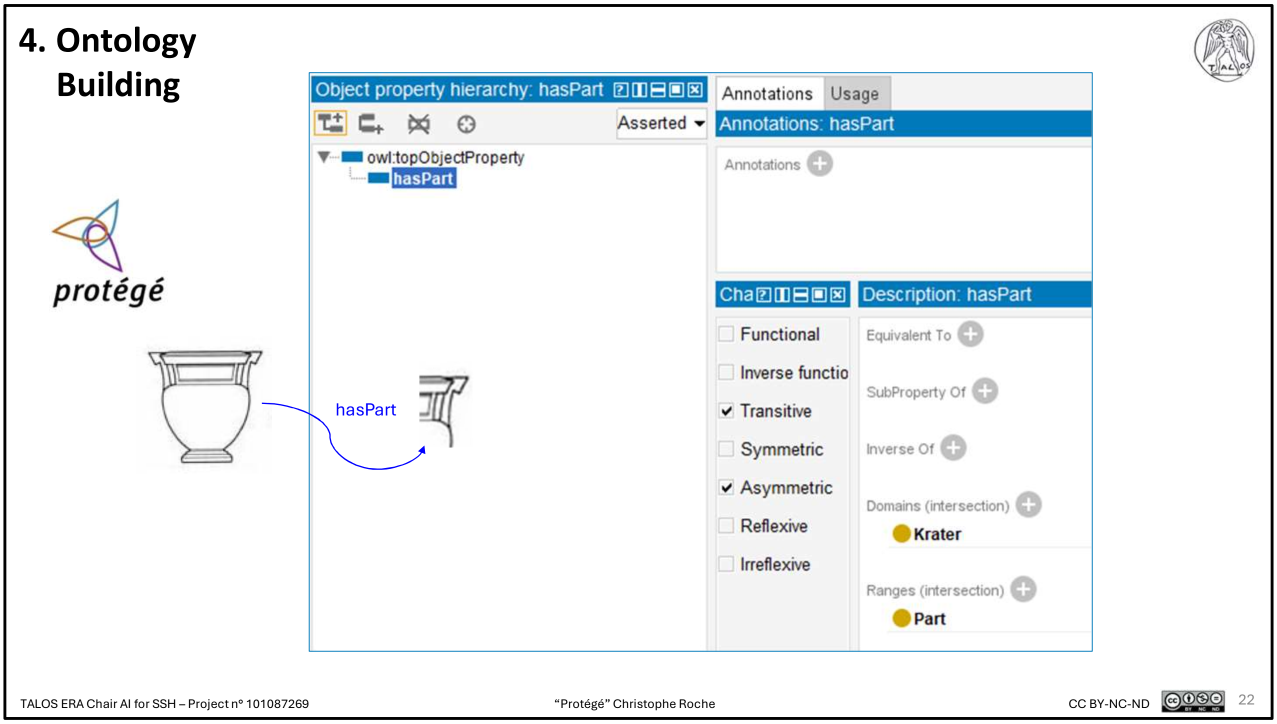
Relationships between individuals



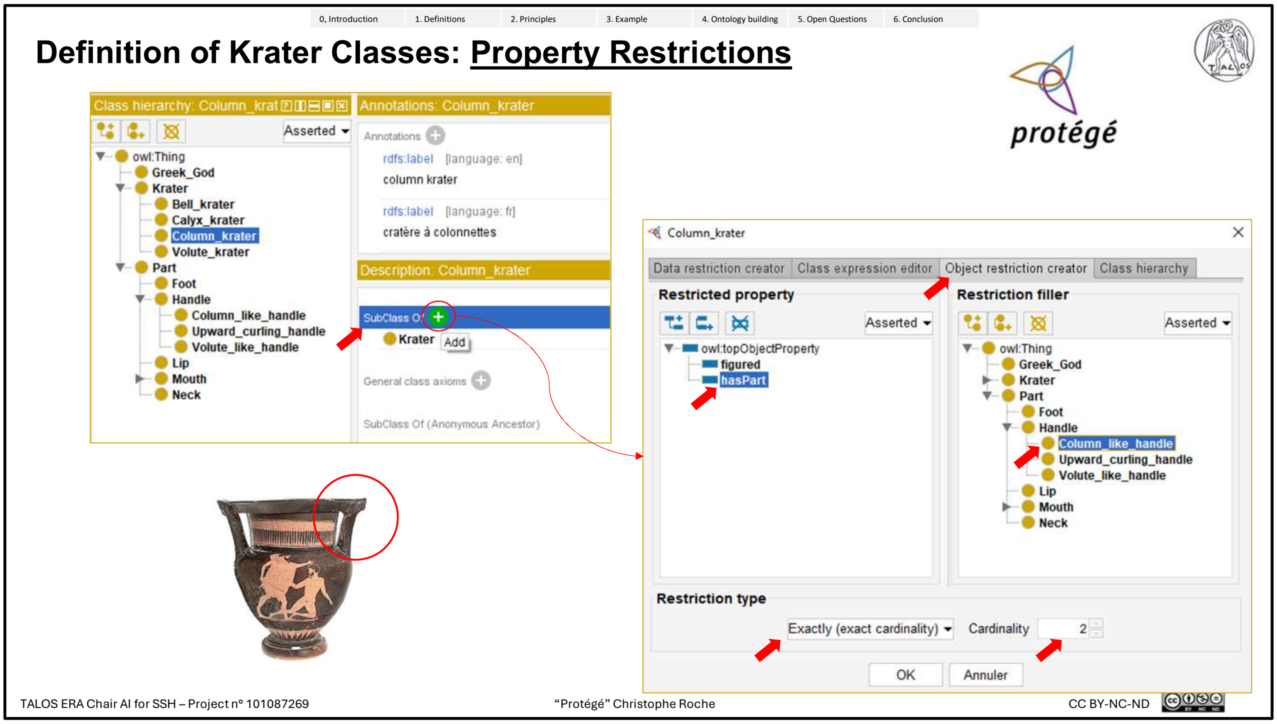
Since a class is defined on the basis of relations between individuals, a krater must be thought of not as a whole, but as a set of individuals, for example as the set of its parts: A column krater is linked by the property hasPart to a foot, a neck, two column-like handles, etc.



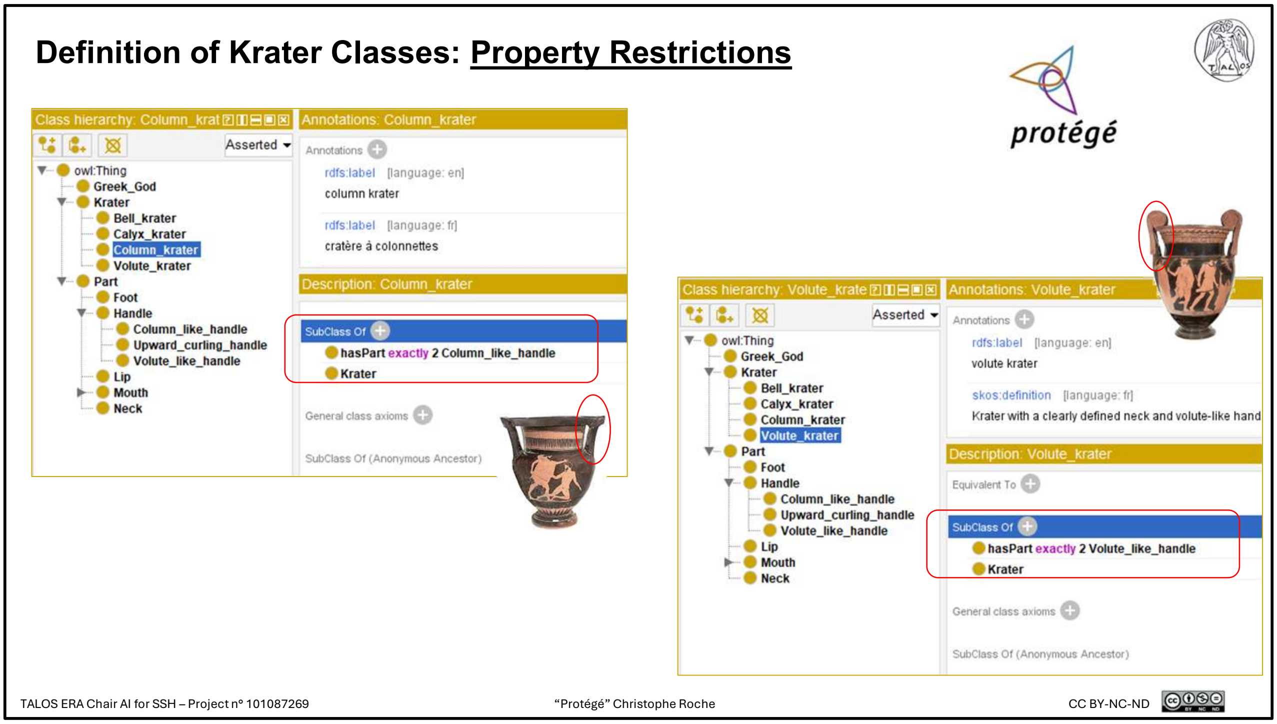
It is therefore necessary to define a new class, the class of parts, disjoint from the class of kraters. The different parts, Foot, Handle, Neck, etc., are subclasses of the class Part.



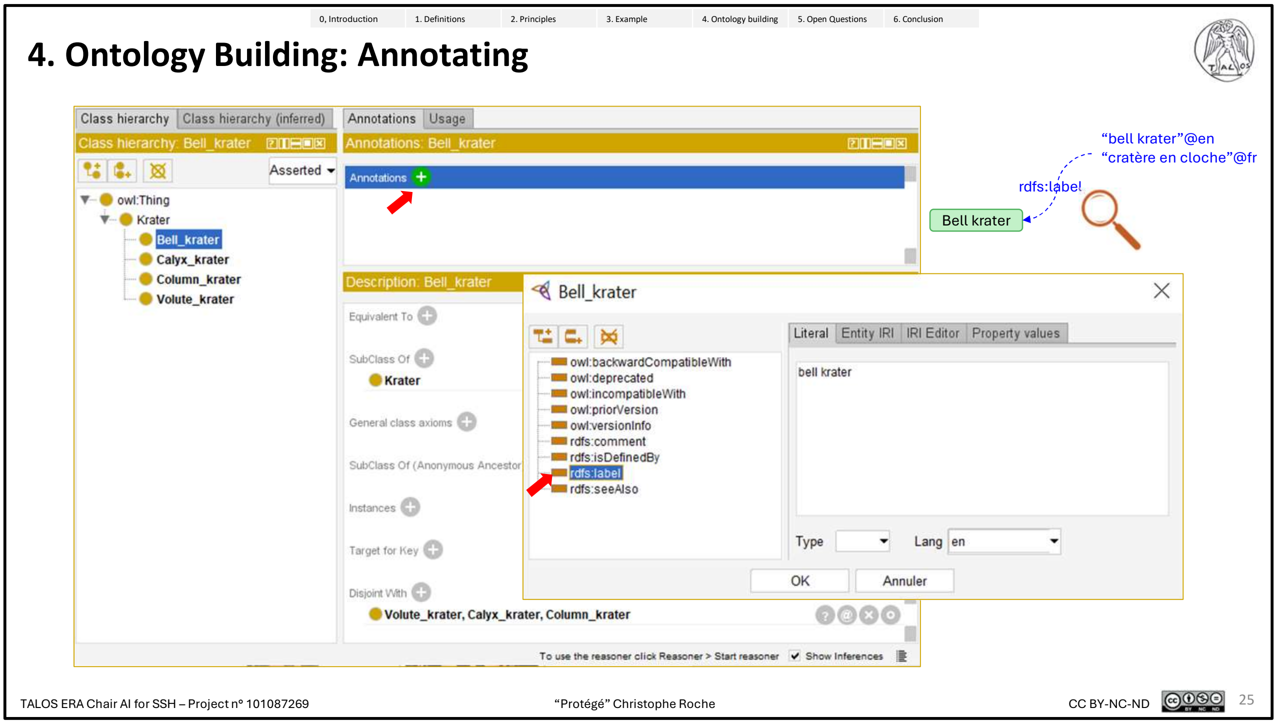
The next step is to set a new property between objects, the hasPart relationship. This relationship links individuals of the krater class, which is the domain of the relationship, to individuals of the class Part, the range of the relationship, which includes all the individuals of its subclasses: Foot, Handle, Neck, etc.



It then remains to specify how the individuals of a class are related to its different parts through the hasPart relation. Thus, the class of column kraters is a subclass of the class of objects linked to exactly 2 column-like handles.

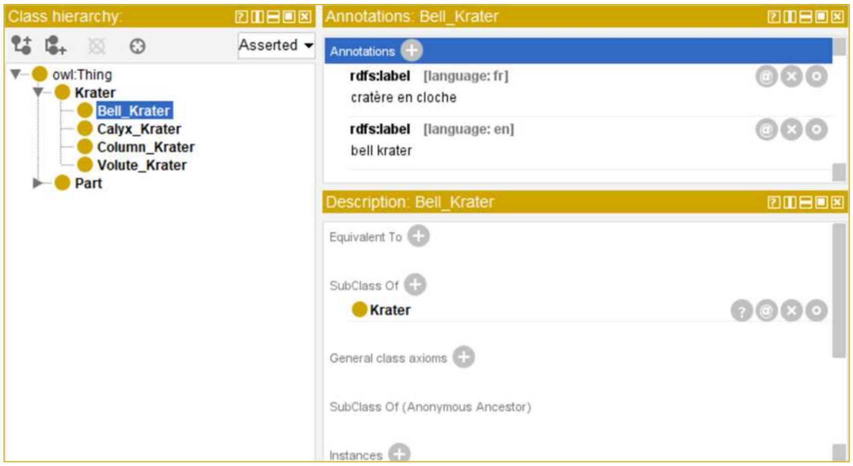


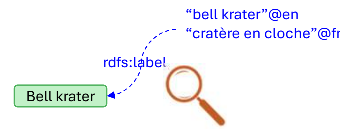
This slide illustrates the formal definitions of the column krater and the volute krater classes.



It is also possible to associate different information with classes in the form of annotations. For example, associating a term designating the class in different languages. The use of W3C vocabularies facilitates the exchange and interoperability of ontologies. In our example, the terms "bell krater" in English and "cratère en cloche" in French refer through the rdfs:label property to the class Bell_Krater

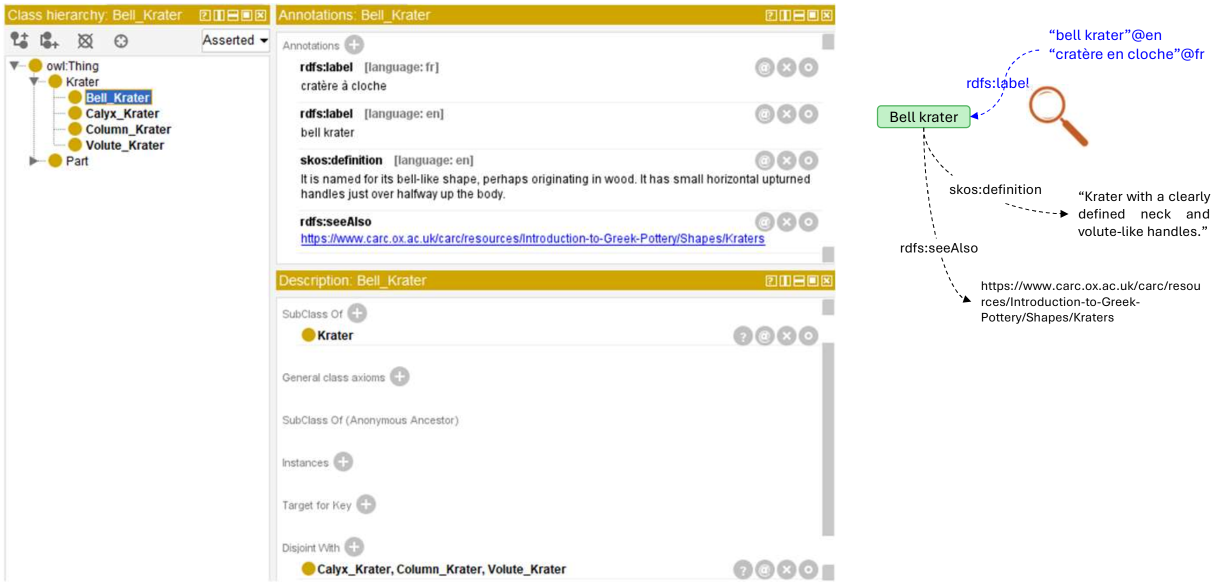
4. Ontology Building: Annotating





…

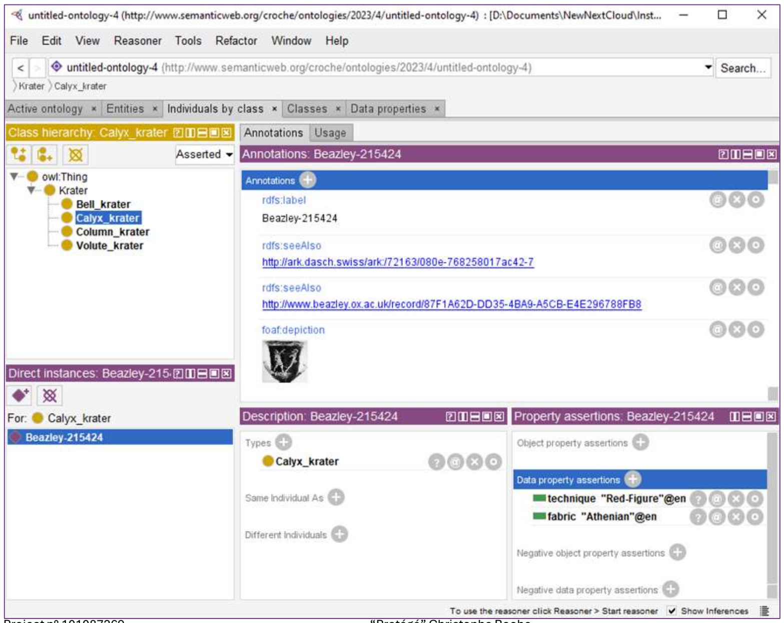
4. Ontology Building: Annotating



TALOS ERA Chair AI for SSH – Project n° 101087269 "Protégé" Christophe Roche 27

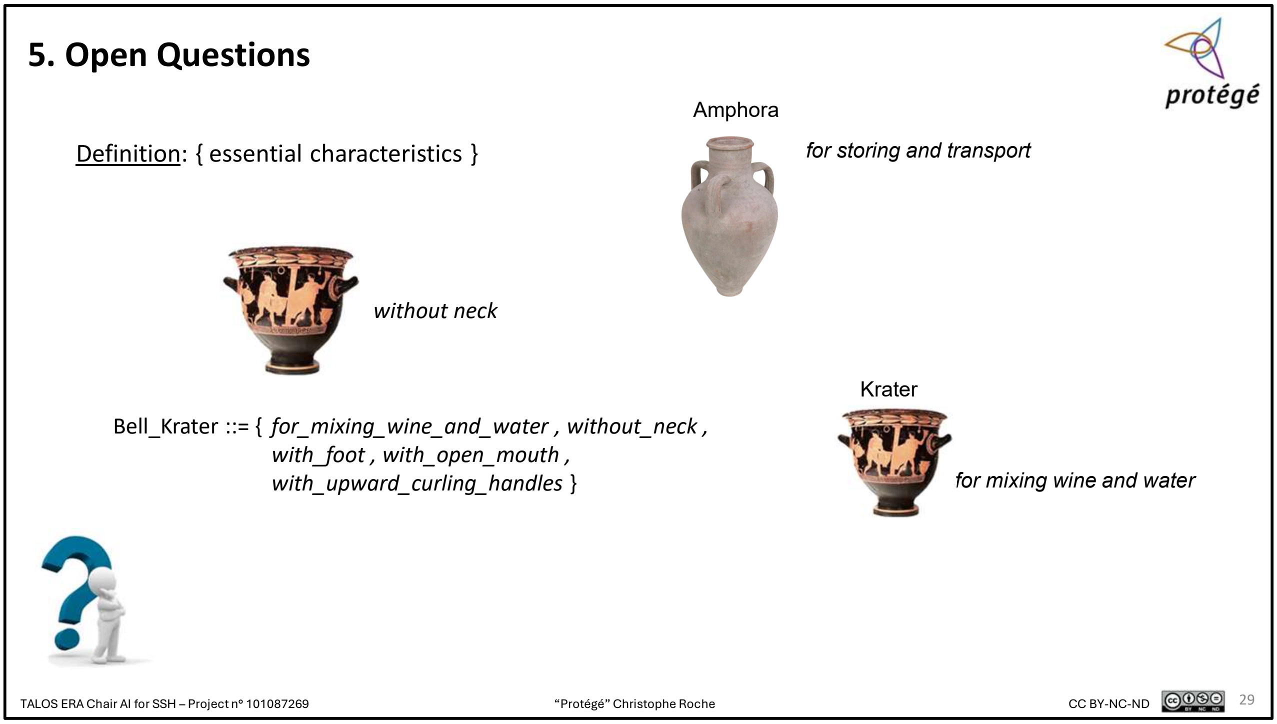
It is also possible, using annotations and standardized vocabularies such as SKOS, to associate a definition using the skos:definition property and to link a class to external resources using the rdfs:seeAlso property

4. Ontology Building: Populating

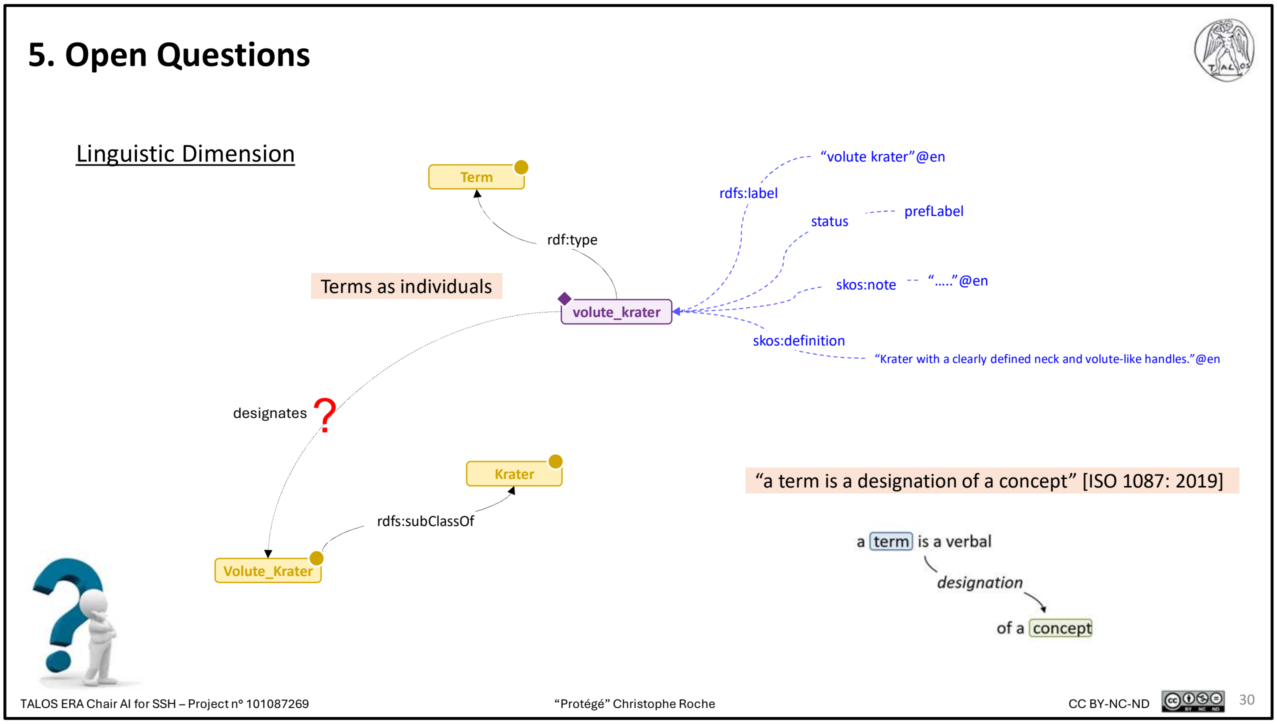


Vase Number: 215424
Fabric: ATHENIAN
Technique: RED-FIGURE
Shape Name: KRATER, CALYX
Date: -450 to -400

A particular vase, for example, the krater reference 215424 in the Beazley archive, will be represented by an individual instance of the calyx krater class. Its attributes, such as the technique used, are represented using data properties, linking the instance not to another object as with object properties, but to data such as a string or numeric value



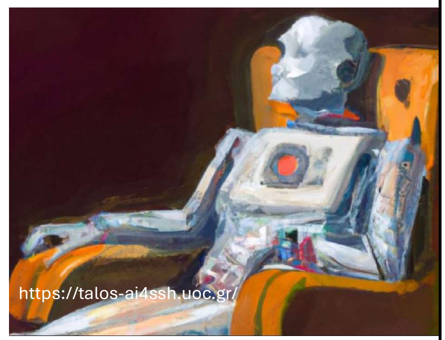
Protégé is an extremely powerful tool. However, some questions remain open concerning the notion of definition and the linguistic dimension associated with an ontology. Considering an object as a set of related entities does not represent all the knowledge that defines a class. If the absence of a part, a bell krater has no neck, can be expressed in the form of a logical property, how can the function of a vase be represented: kraters are for mixing water and wine, while amphorae are for storage and transport?
'for storage and transport' and 'for mixing water and wine' are essential characteristics, which contribute to the definition of these classes.
While the representation of essential characteristics in the form of individuals is logically correct, it is not satisfactory from the point of view of the domain knowledge.



Concerning the linguistic dimension, a term, defined as a verbal designation of a concept, cannot be reduced to a simple label attached to a class. The term requires explicit representation in the form of an individual in order to be able to attach information such as their grammatical category (part of speech) or status. Such a representation raises problems whose solutions complicate the ontology

6. Conclusion

- ✓ Free environment
- ✓ Large Community
- ✓ Definition based on relations between objects (Class)
- ✓ Description Logic Reasoners
- ✓ W3C Standards compliant



- How to represent essential characteristics (Concept)?
- How to represent the linguistic dimension?
- How to take into account the way of thinking of Experts?

To conclude. Protégé is an extremely powerful open-source tool with a large user community. Compliant with W3C standards, it is based on logical foundations that allow to verify several properties, including the consistency of the ontology. Protégé is based on the notion of class, not concept. A class is defined not according to the nature of its instances, but according to the relationships that its instances have with other objects. These principles are well-suited for organizing objects into a hierarchy of classes. However, they do not always align with the way experts think. Moreover, the essential characteristics underlying concept definitions and the linguistic dimension of ontology raise issues whose solutions in Protégé are not entirely satisfactory..