



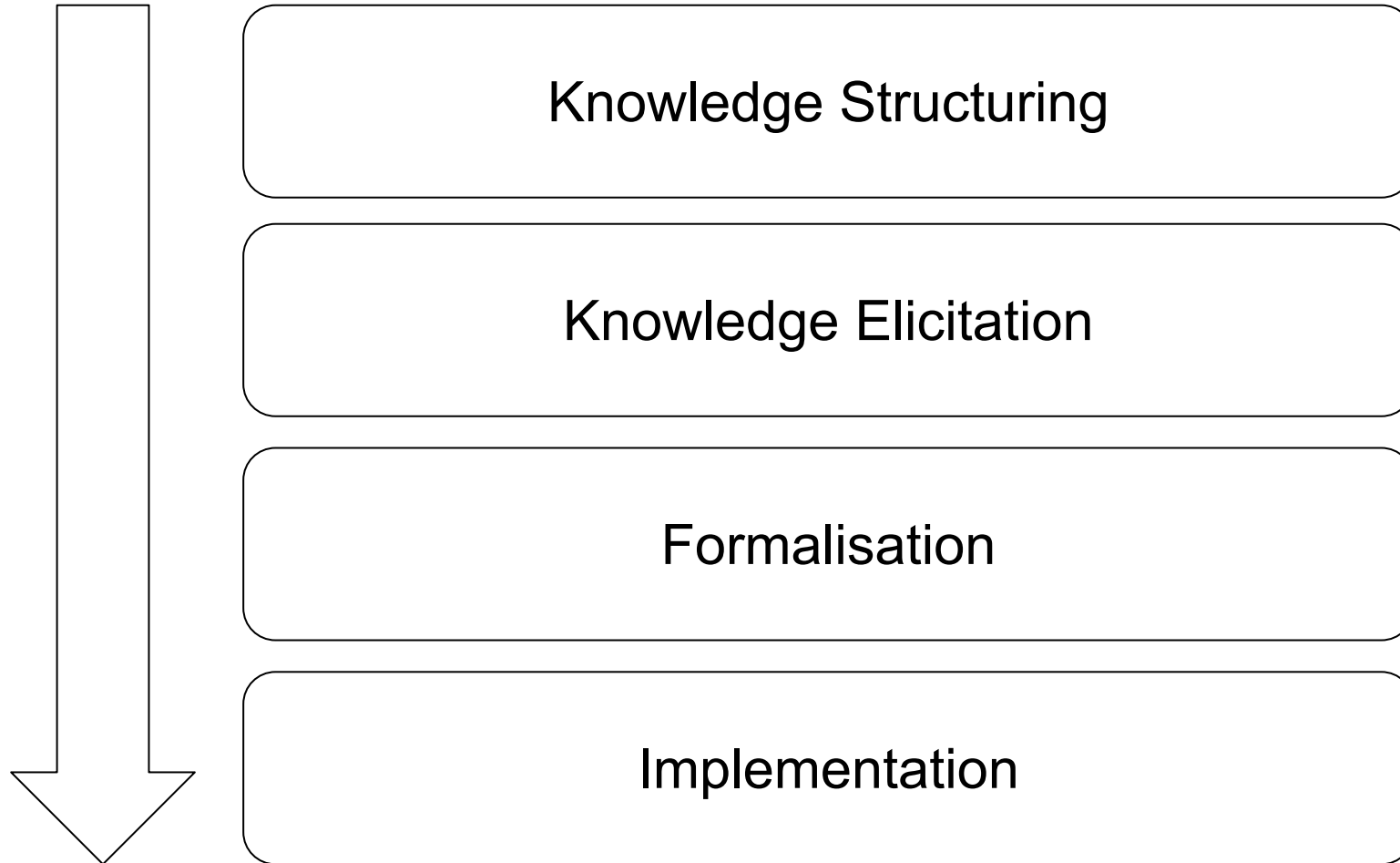
# Tools for Ontology Engineering

Demonstrator workshop - 10/03/2021

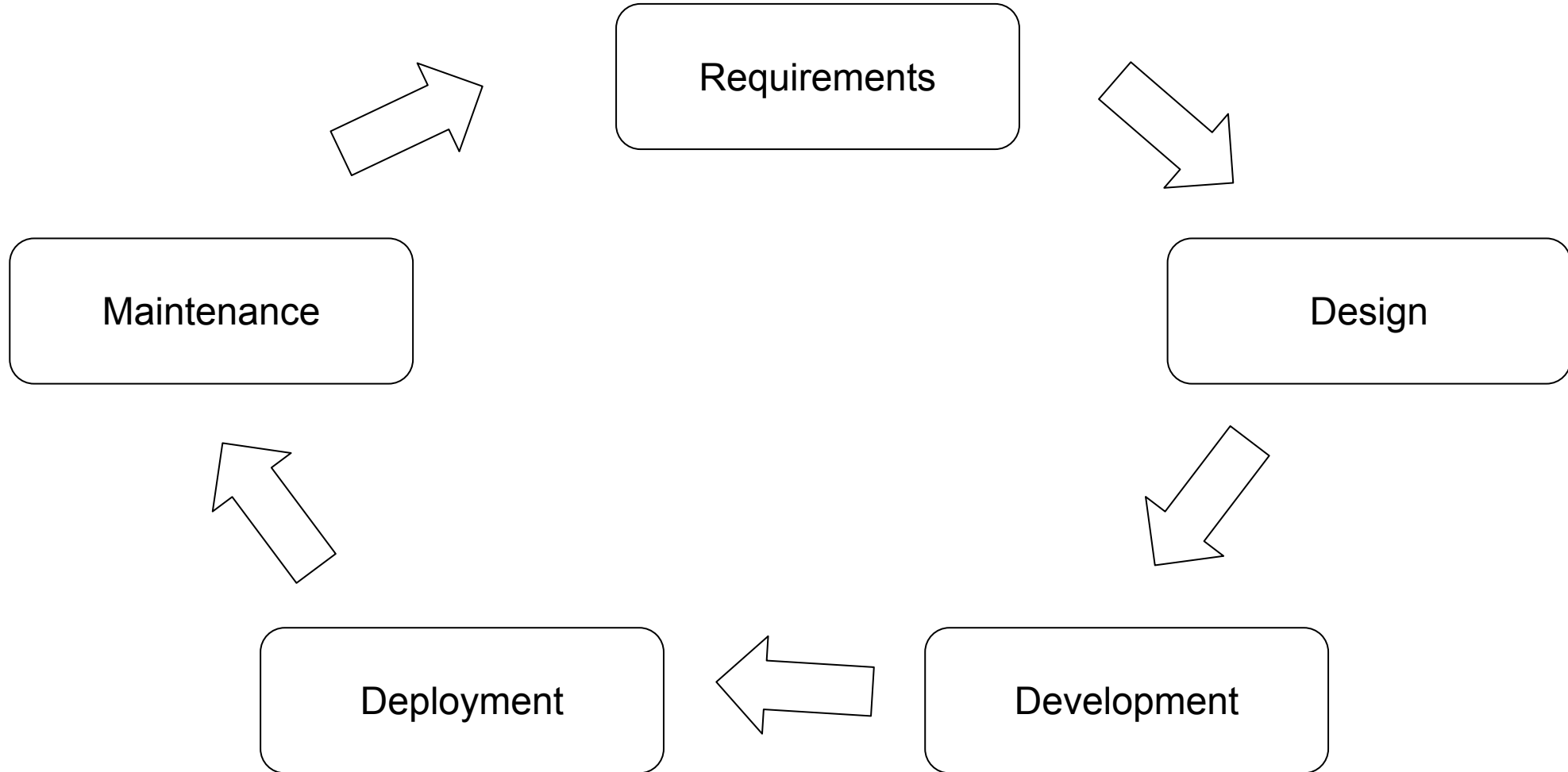
---

*Mathieu d'Aquin, NUI Galway*

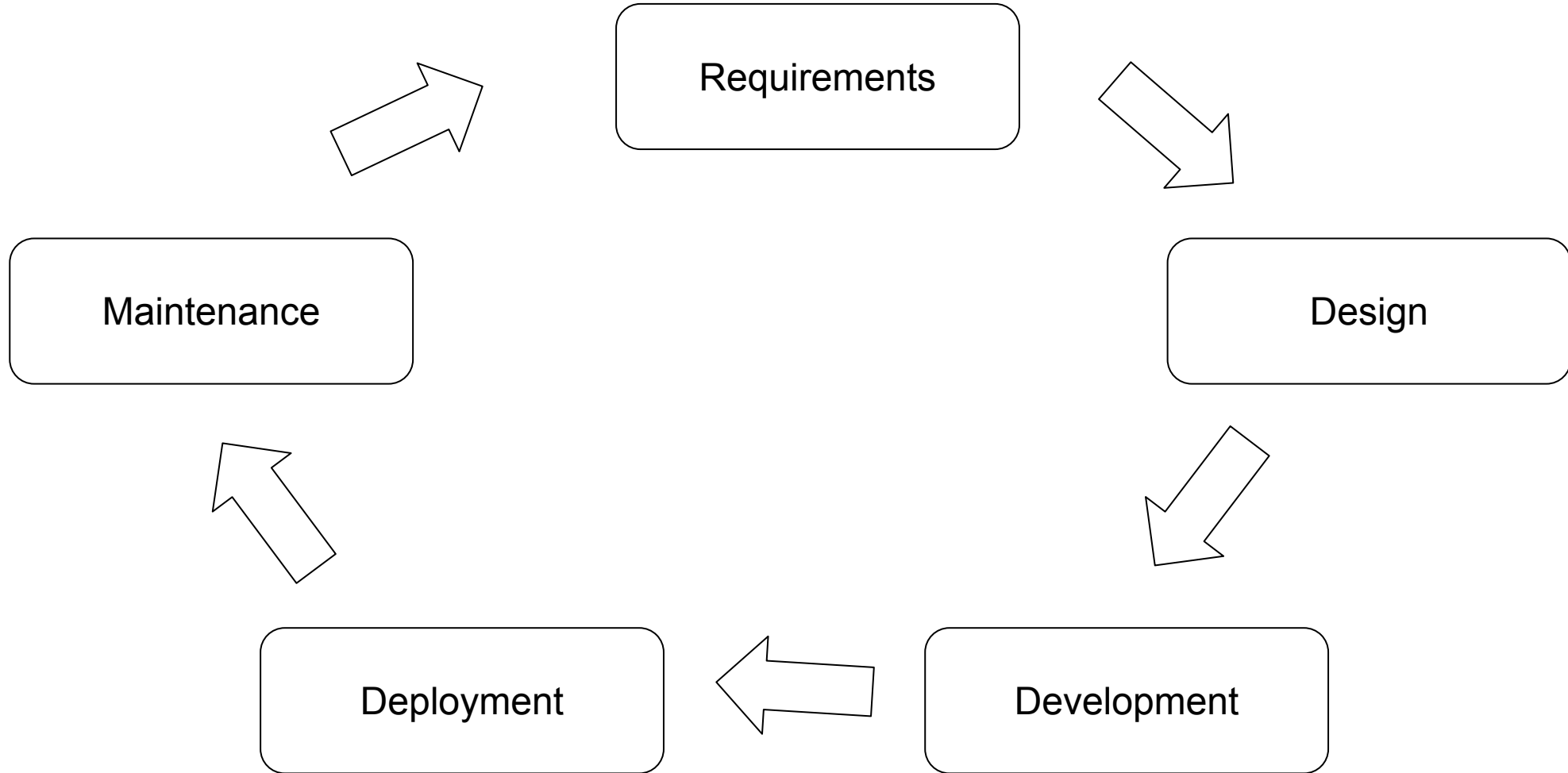
# The traditional knowledge engineering process



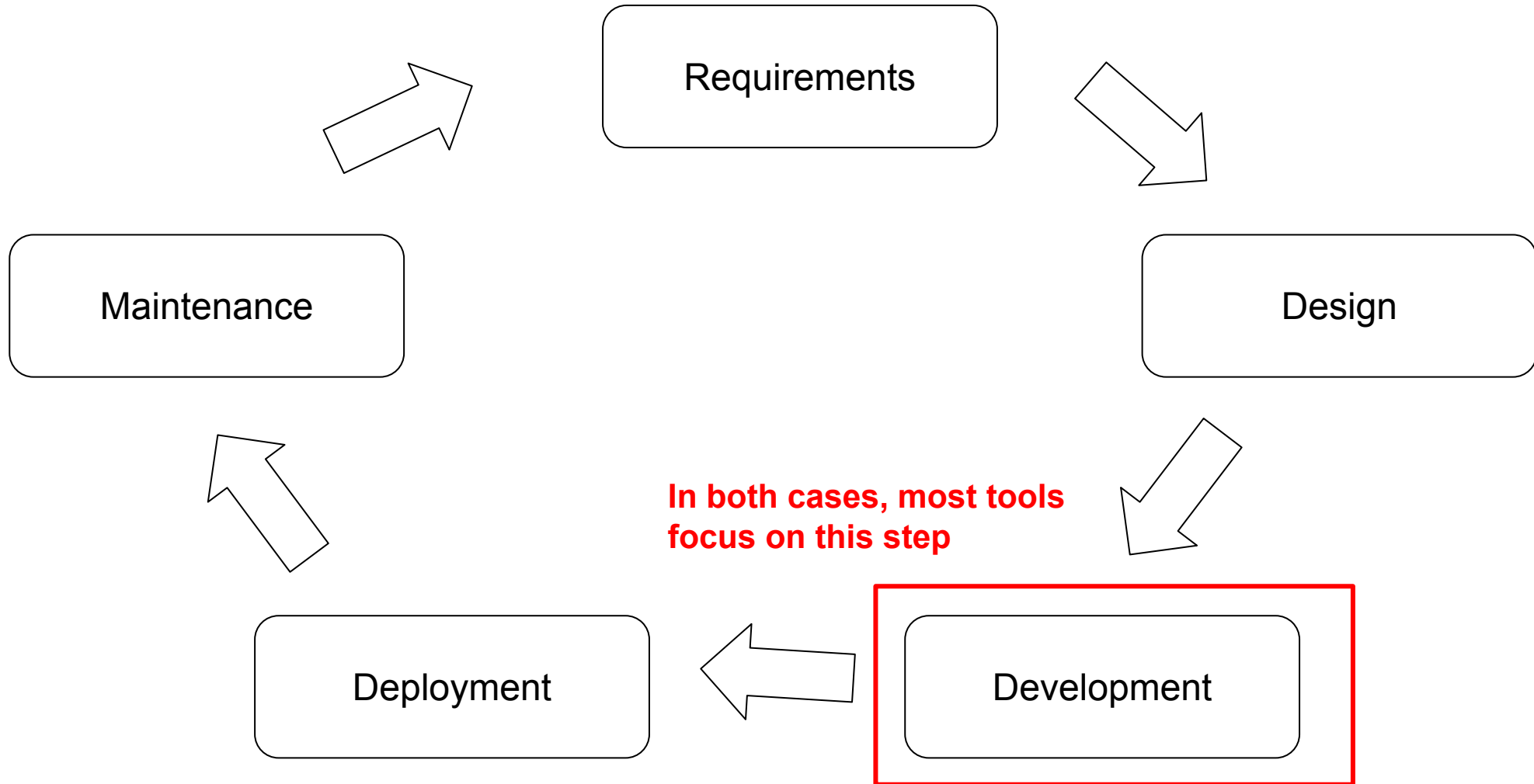
# The traditional *software* engineering process



# Ontology engineering process



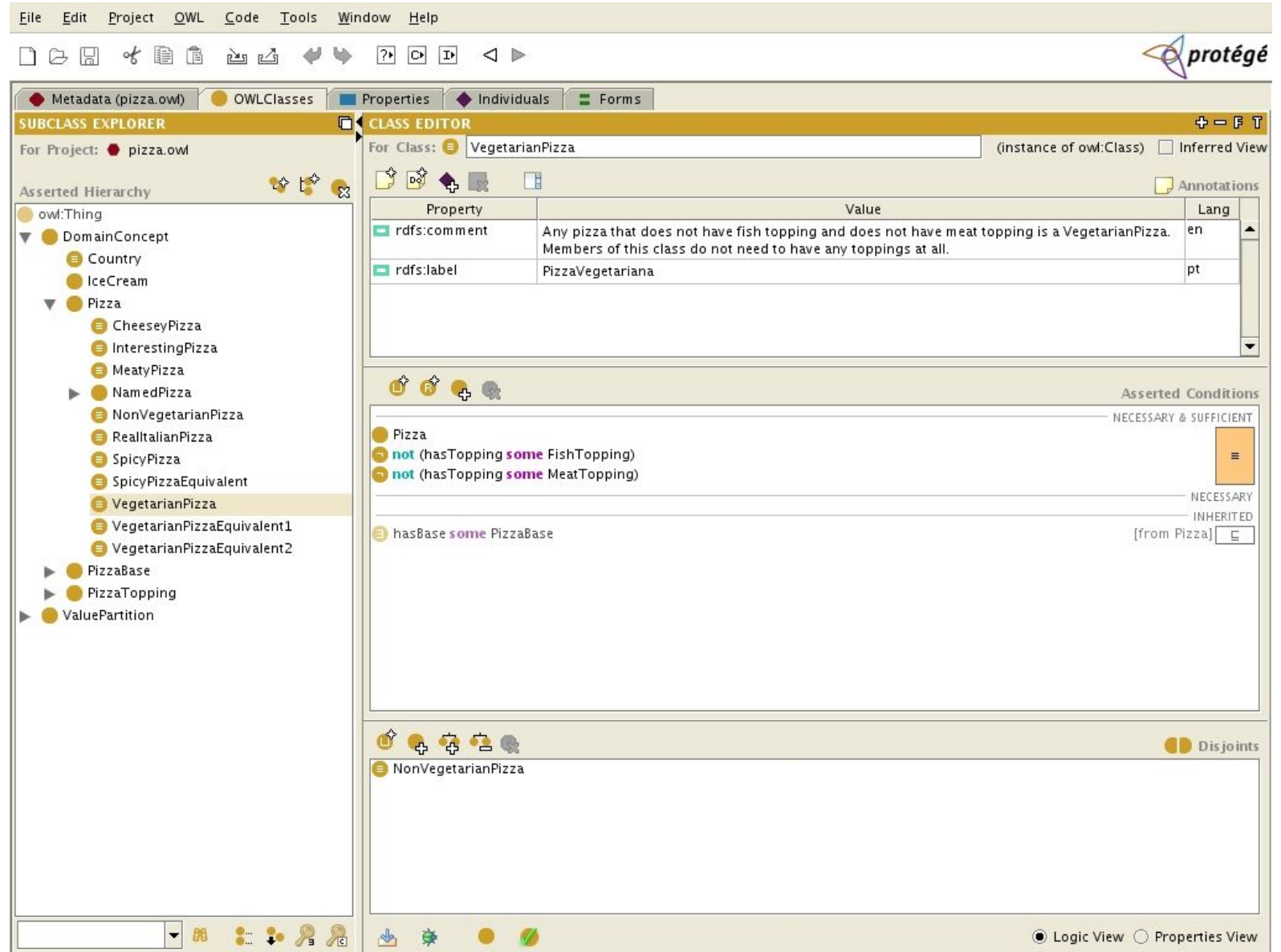
# Ontology engineering process



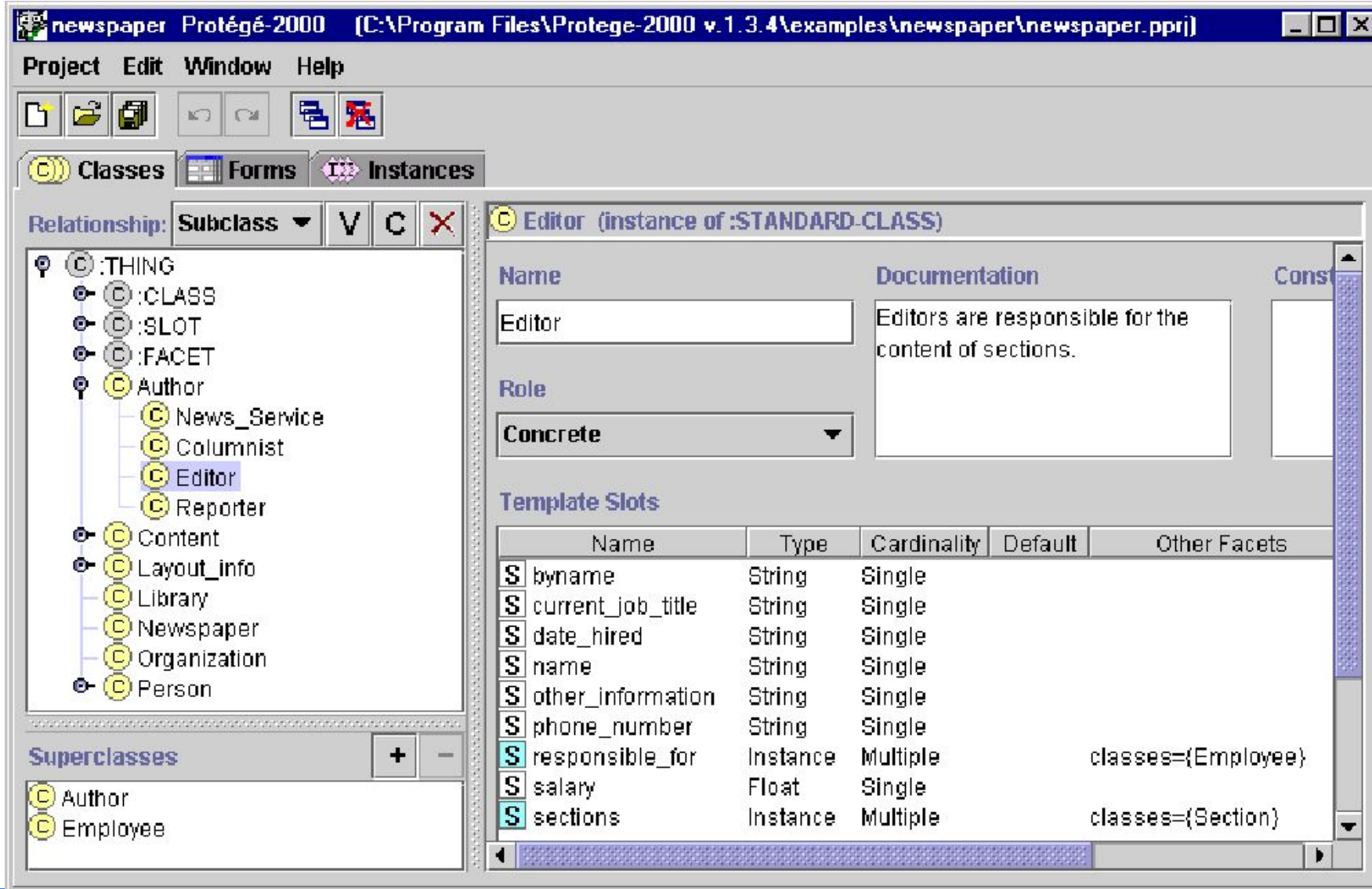
Best known and used ontology editor.

The “IDE” of ontology engineering.

Long history...



# Protégé-2000



The screenshot shows the Protégé-2000 interface with the 'Editor' class selected. The window title is 'newspaper Protégé-2000 [C:\Program Files\Protege-2000 v.1.3.4\examples\newspaper\newspaper.pprj]'. The menu bar includes 'Project', 'Edit', 'Window', and 'Help'. The toolbar contains icons for file operations and navigation. The 'Classes' tab is active, showing a hierarchy of classes. The 'Relationship' dropdown is set to 'Subclass'. The 'Editor' class is selected in the hierarchy, and its details are shown in the right pane.

**Relationship:** Subclass

**Class Hierarchy:**

- :THING
  - :CLASS
    - :SLOT
      - :FACET
        - Author
          - News\_Service
          - Columnist
          - Editor
          - Reporter
        - Content
        - Layout\_info
        - Library
        - Newspaper
        - Organization
        - Person

**Superclasses:** Author, Employee

**Editor (instance of :STANDARD-CLASS)**

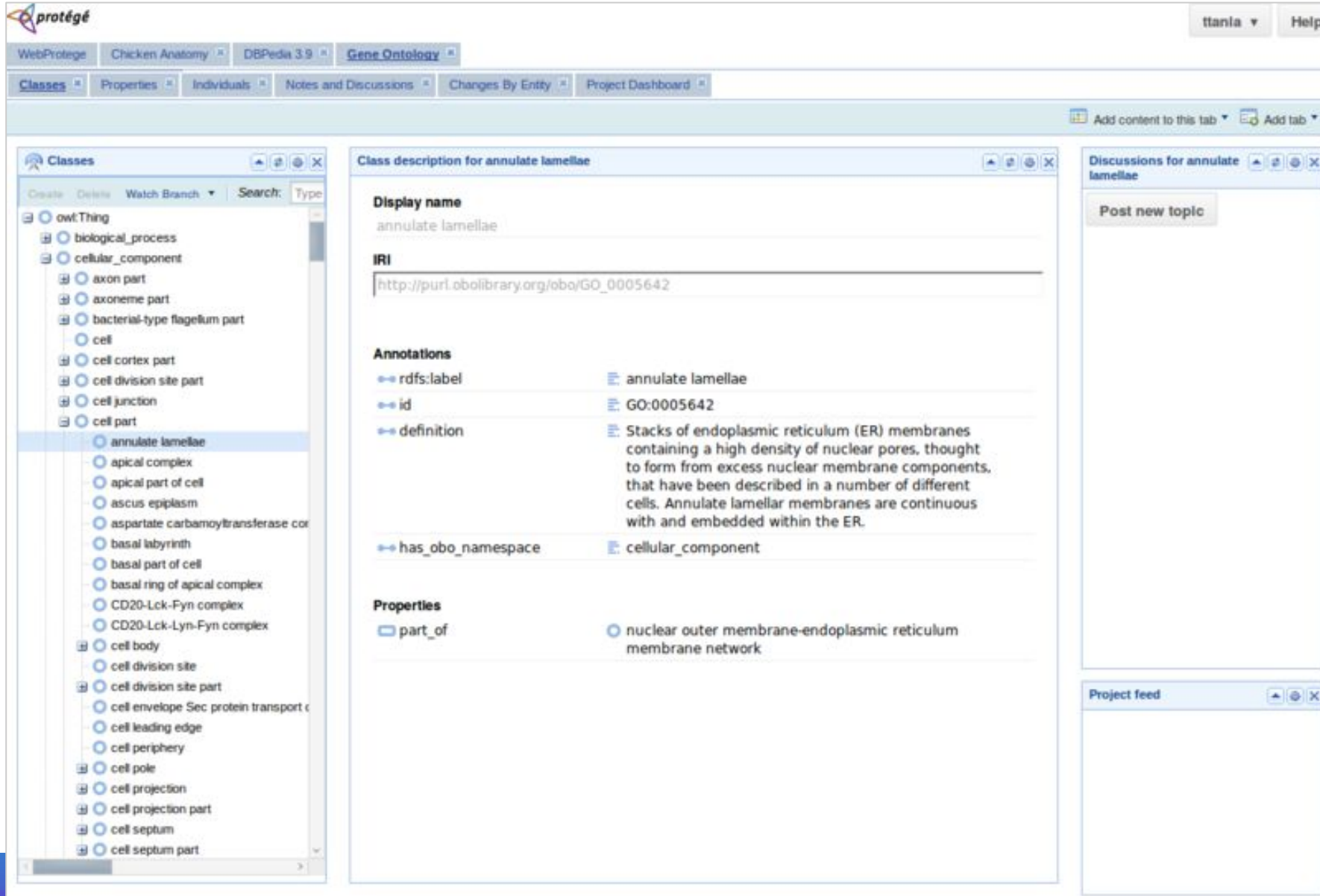
**Name:** Editor

**Documentation:** Editors are responsible for the content of sections.

**Role:** Concrete

**Template Slots:**

Name	Type	Cardinality	Default	Other Facets
byname	String	Single		
current_job_title	String	Single		
date_hired	String	Single		
name	String	Single		
other_information	String	Single		
phone_number	String	Single		
responsible_for	Instance	Multiple		classes={Employee}
salary	Float	Single		
sections	Instance	Multiple		classes={Section}

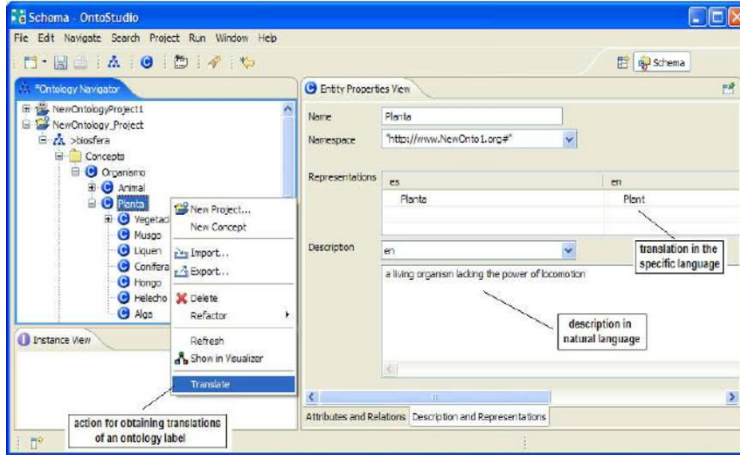


The screenshot displays the Web-Protégé web interface. At the top, there are navigation tabs for 'WebProtégé', 'Chicken Anatomy', 'DBPedia 3.9', and 'Gene Ontology'. Below these are tabs for 'Classes', 'Properties', 'Individuals', 'Notes and Discussions', 'Changes By Entry', and 'Project Dashboard'. The main interface is divided into three panels:

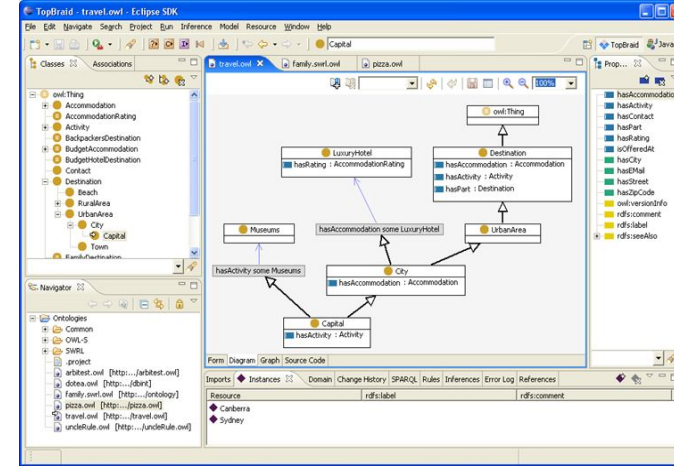
- Classes Panel (Left):** A tree view showing a hierarchy of classes. The 'cell part' class is expanded, and 'annulate lamellae' is selected.
- Class description for annulate lamellae (Center):**
  - Display name:** annulate lamellae
  - IRI:** [http://purl.obolibrary.org/obo/GO\\_0005642](http://purl.obolibrary.org/obo/GO_0005642)
  - Annotations:**
    - rdfs:label: annulate lamellae
    - id: GO:0005642
    - definition: Stacks of endoplasmic reticulum (ER) membranes containing a high density of nuclear pores, thought to form from excess nuclear membrane components, that have been described in a number of different cells. Annulate lamellar membranes are continuous with and embedded within the ER.
    - has\_obo\_namespace: cellular\_component
  - Properties:**
    - part\_of: nuclear outer membrane-endoplasmic reticulum membrane network
- Discussions for annulate lamellae (Right):** A section with a 'Post new topic' button.
- Project feed (Bottom Right):** A section for project updates.



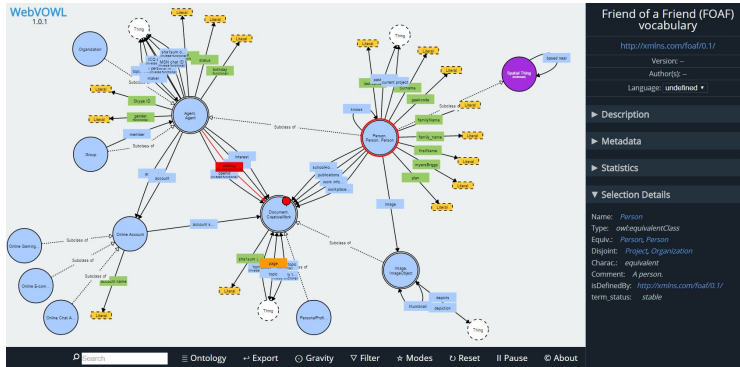
# Many other ontology editors *not all maintained*



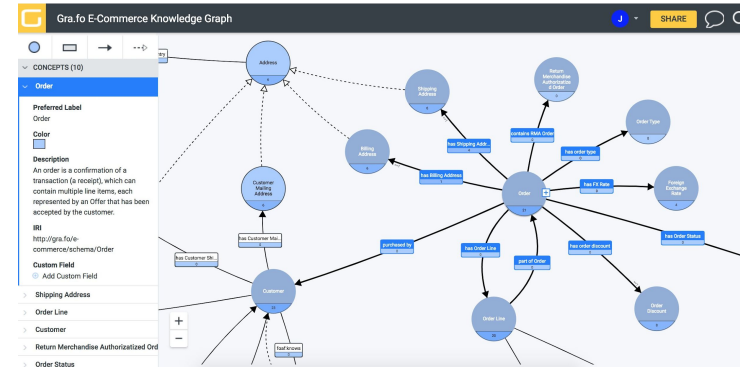
NeOn Toolkit



TopBraid Composer



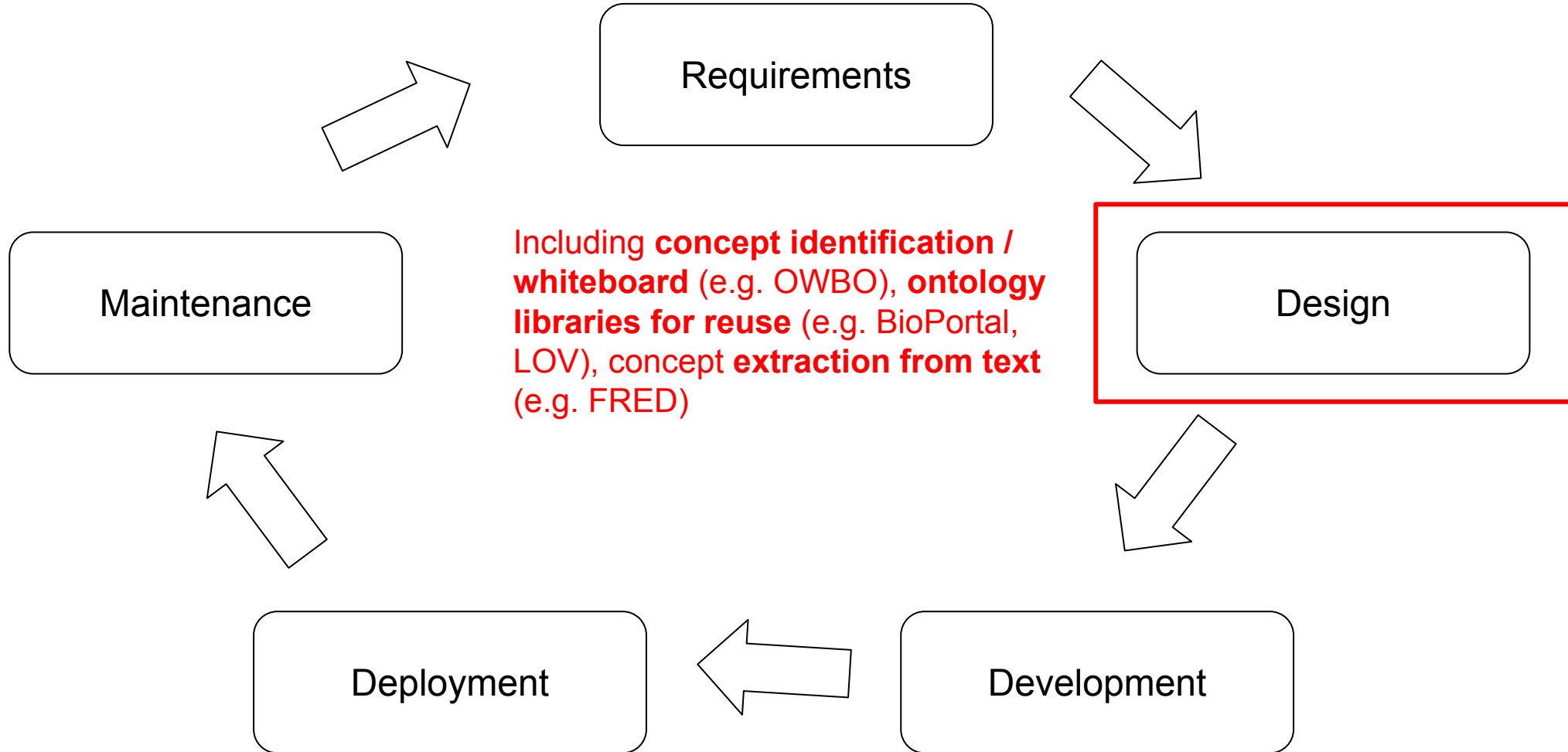
Web VOWL



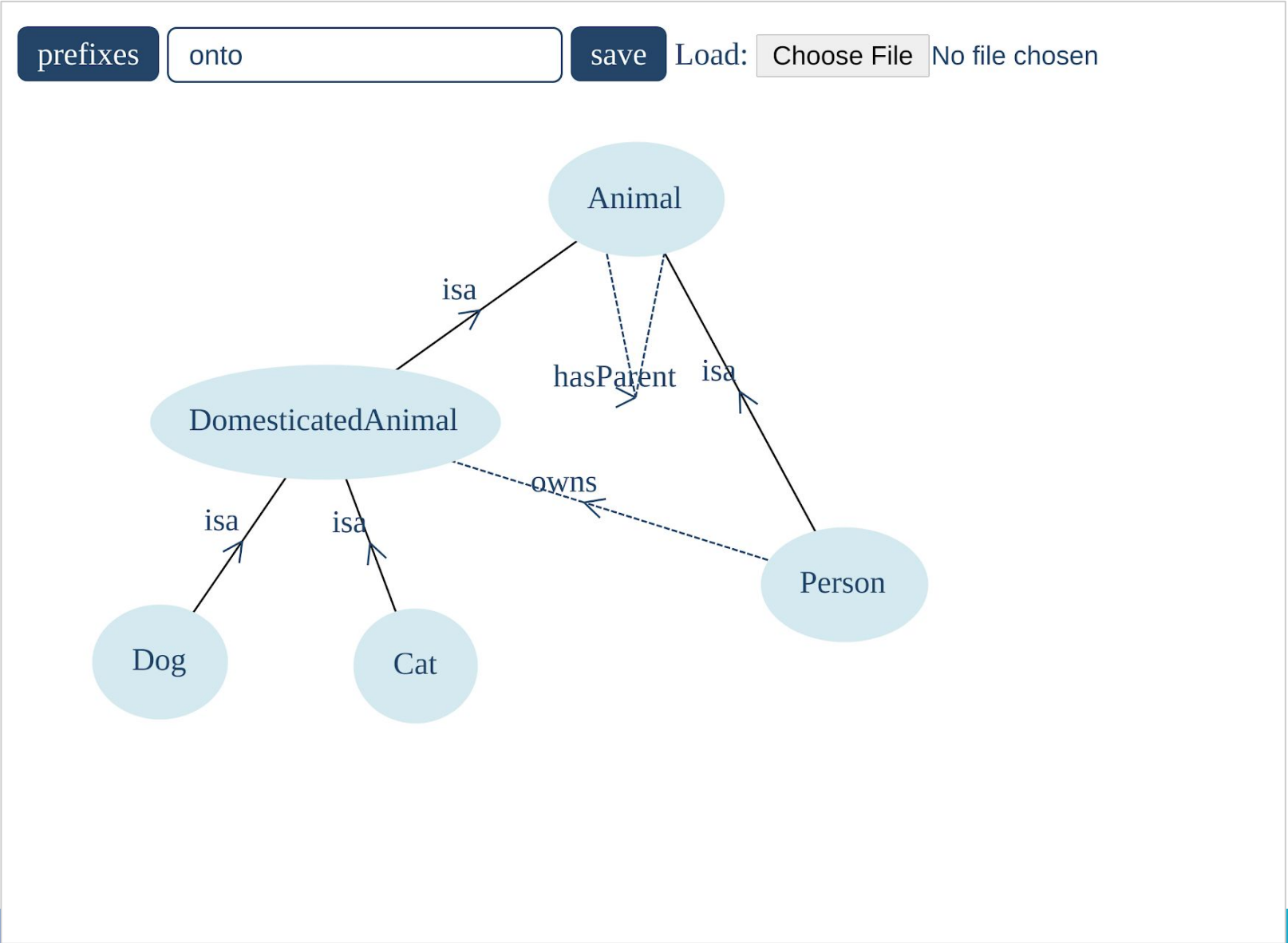
Gra.fo

# Ontology engineering process

## *Standalone tools and Protégé plugins*



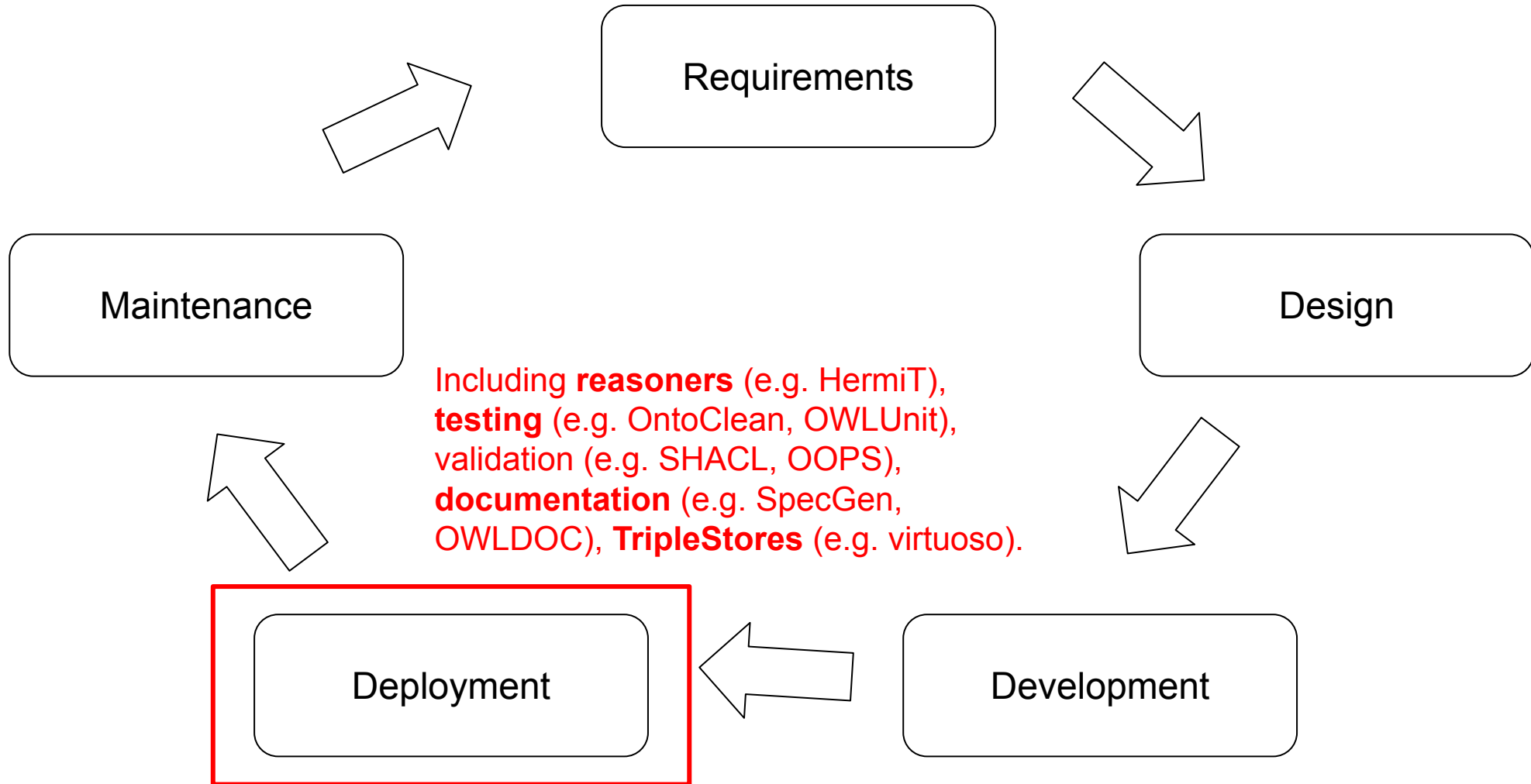
# Example: OWBO



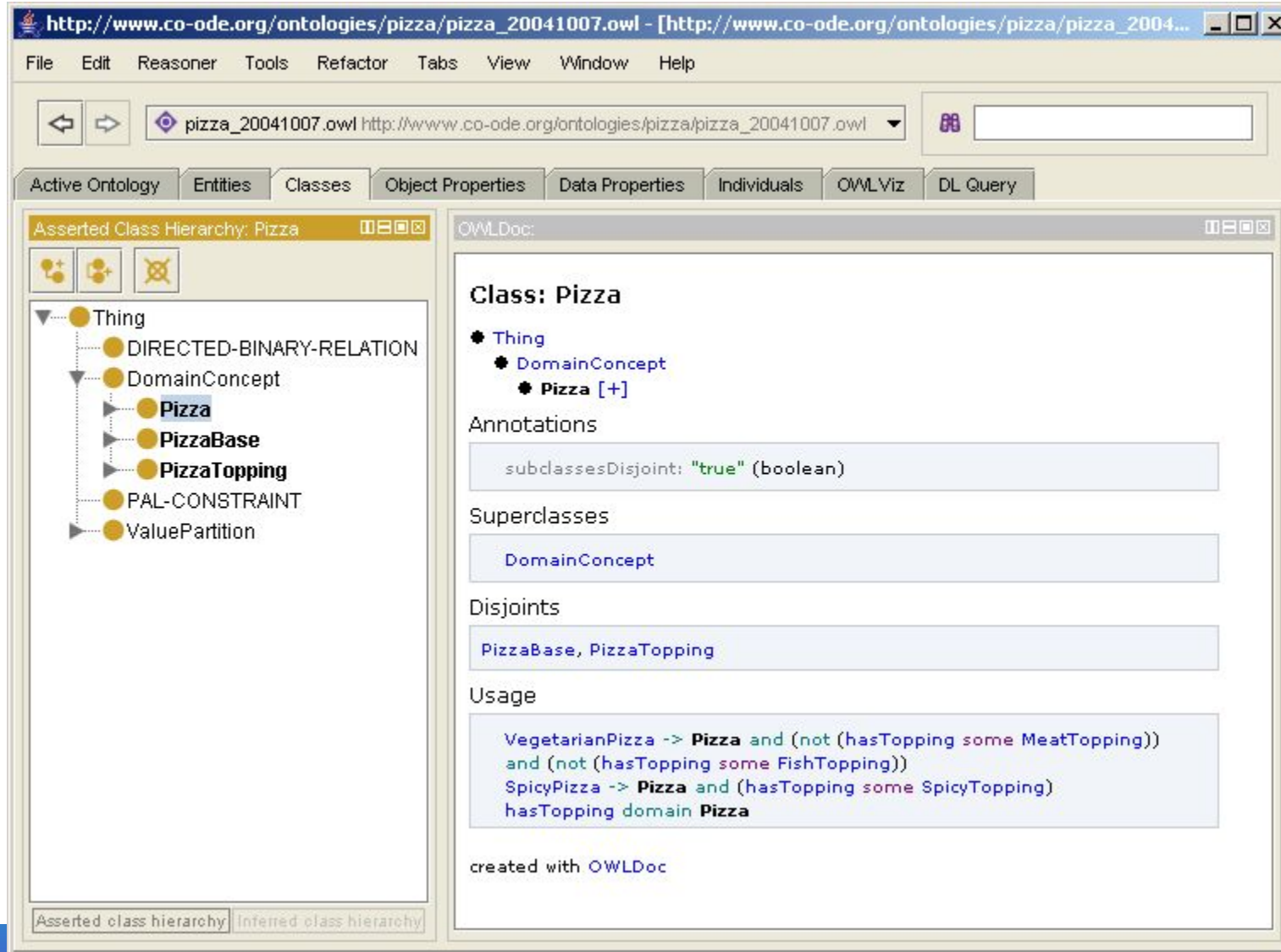


# Ontology engineering process

## *Standalone tools and Protégé plugins*



# Example: OWLDoc



The screenshot shows the OWLDoc application interface. The browser address bar displays the URL: `http://www.co-ode.org/ontologies/pizza/pizza_20041007.owl`. The application has a menu bar with options: File, Edit, Reasoner, Tools, Refactor, Tabs, View, Window, Help. Below the menu bar is a navigation area with a back button, a forward button, and a dropdown menu showing the current ontology: `pizza_20041007.owl`. The main interface is divided into two panes.

The left pane, titled "Asserted Class Hierarchy: Pizza", shows a tree view of the class hierarchy:

- Thing
  - DIRECTED-BINARY-RELATION
  - DomainConcept
    - Pizza**
    - PizzaBase
    - PizzaTopping
  - PAL-CONSTRAINT
  - ValuePartition

The right pane, titled "OWLDoc:", displays the details for the selected class, **Class: Pizza**. It shows the following information:

- Class: Pizza**
- Thing
  - DomainConcept
    - Pizza [+]
- Annotations**
  - subclassesDisjoint: "true" (boolean)
- Superclasses**
  - DomainConcept
- Disjoints**
  - PizzaBase, PizzaTopping
- Usage**
  - VegetarianPizza -> **Pizza** and (not (hasTopping some MeatTopping)) and (not (hasTopping some FishTopping))
  - SpicyPizza -> **Pizza** and (hasTopping some SpicyTopping)
  - hasTopping domain **Pizza**
- created with OWLDoc

At the bottom of the interface, there are two tabs: "Asserted class hierarchy" (selected) and "Inferred class hierarchy".

# Example: OOPS

**OOPS! (Ontology Pitfalls Scanner)** helps you to detect some of the most common pitfalls appearing when developing ontologies. To utilize it, enter a URI or paste an RDF/XML document into the text field above. A list of pitfalls and the elements of your ontology where they appear will be displayed.

Scanner by JRI:  Scanner by URI:

Scanner by direct input:  Scanner by RDF:

## Evaluation results

[Expand All] | [Collapse All]

Appearing pitfall name	Pitfall frequency
Results for P04: Creating unconnected ontology elements.	12 cases
Results for P05: Defining wrong inverse relationships.	2 cases
Results for P06: Including cycles in the hierarchy.	
<p>A cycle between two classes in the hierarchy is included in the ontology, although it is not intended to have such equivalent. That is, some class A has a subclass B and at the same time B is a superclass of A. An example of this type is represented by the class "Professor" as subclass of "Person", and the class "Person" as subclass of "Professor".</p> <p>This pitfall appears in the following elements:</p> <p><a href="http://swrc.ontoware.org/ontology#Proceedings">http://swrc.ontoware.org/ontology#Proceedings</a></p>	
Results for P08: Missing annotations.	45 cases
Results for P10: Missing disjointness.	1 cases
<p>The ontology lacks disjoint axioms between classes or between properties that should be defined as disjoint. For example, we can create the classes "Odd" and "Even" (or the classes "Prime" and "Composite") without being disjoint; such representation is not correct based on the definition of these types of numbers.</p> <p>This pitfall applies to the ontology in general instead of specific elements.</p>	
Results for P11: Missing domain or range in properties.	28 cases
Results for P12: Missing equivalent properties.	1 cases
Results for P13: Missing inverse relationships.	2 cases
Results for P19: Swapping intersection and union.	1 cases
Results for P22: Using different naming criteria in the ontology.	1 cases
SUGGESTION: symmetric or transitive object properties.	2 cases
<p>The domain and range axioms are equal for each of the following object properties. Could they be symmetric or transitive?</p> <p><a href="http://data.semanticweb.org/ns/swc/ontology#HasFert">http://data.semanticweb.org/ns/swc/ontology#HasFert</a></p> <p><a href="http://data.semanticweb.org/ns/swc/ontology#IsPartOf">http://data.semanticweb.org/ns/swc/ontology#IsPartOf</a></p>	
WARNING: the following classes do not have rdfs:type owl:Class or equivalent.	3 cases

Ontology elements affected by the pitfall

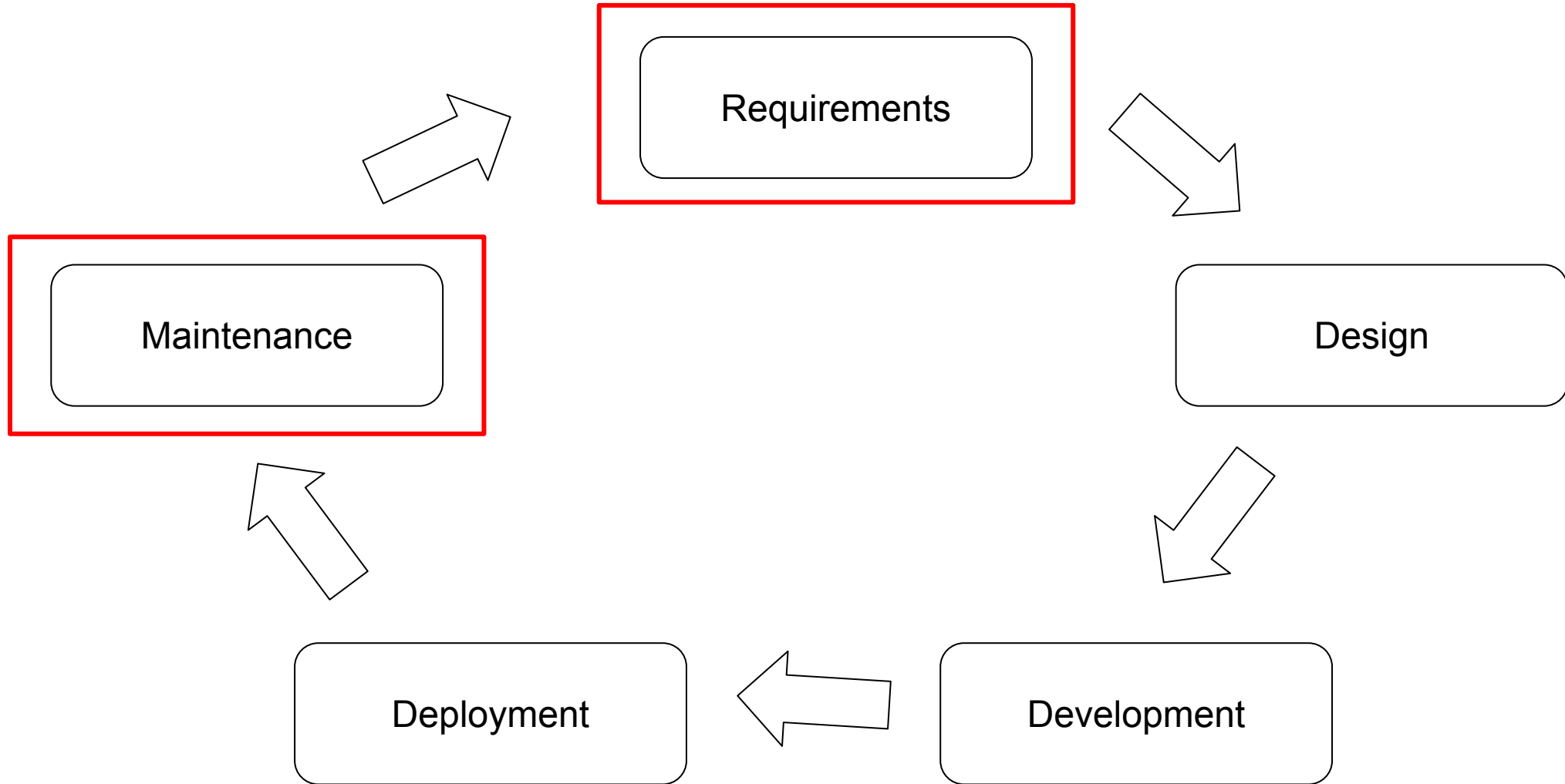
Pitfall description



Please, help us making OOPS! better. **Feedback** is more than welcome!

# Ontology engineering process

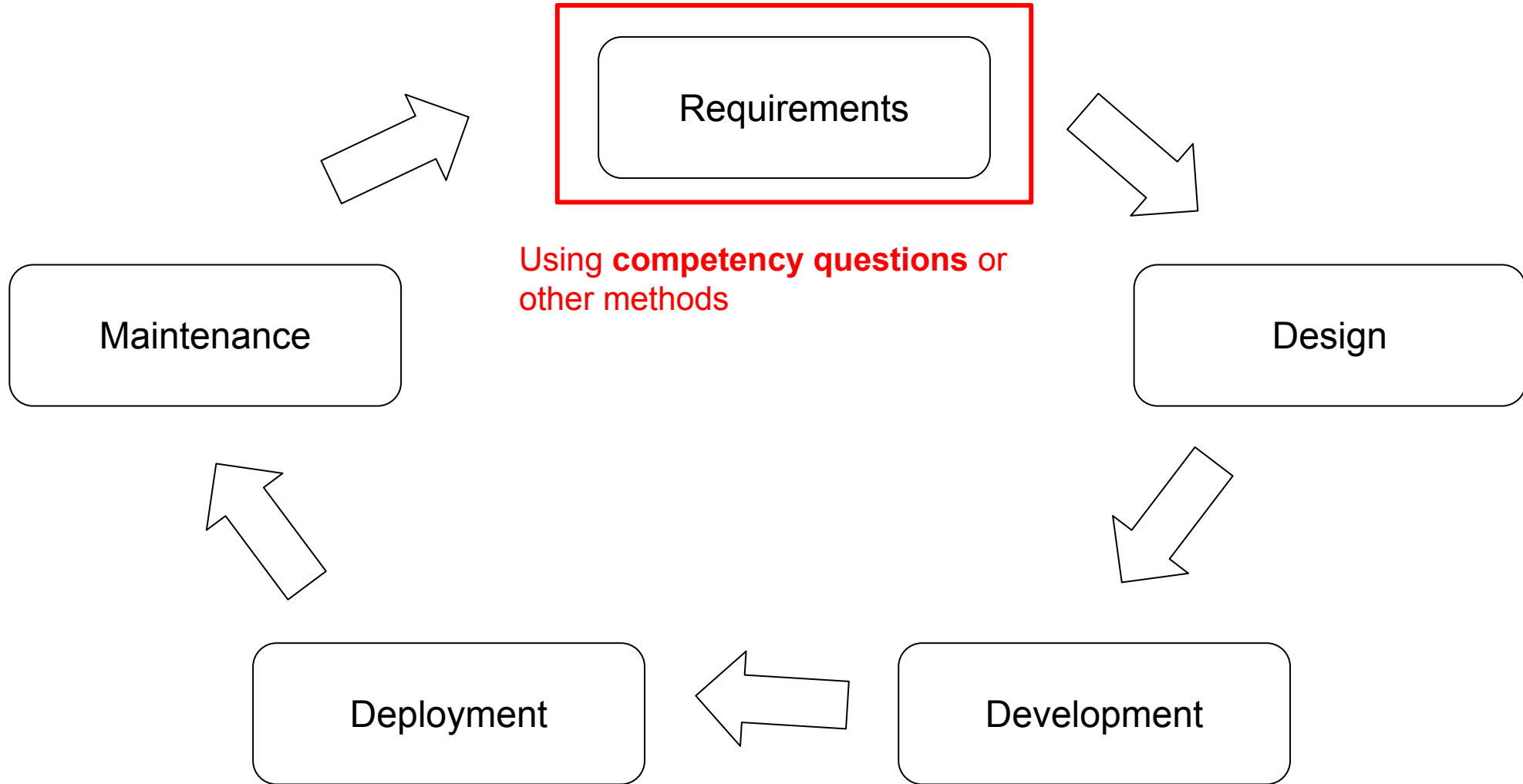
## *Standalone tools and Protégé plugins*





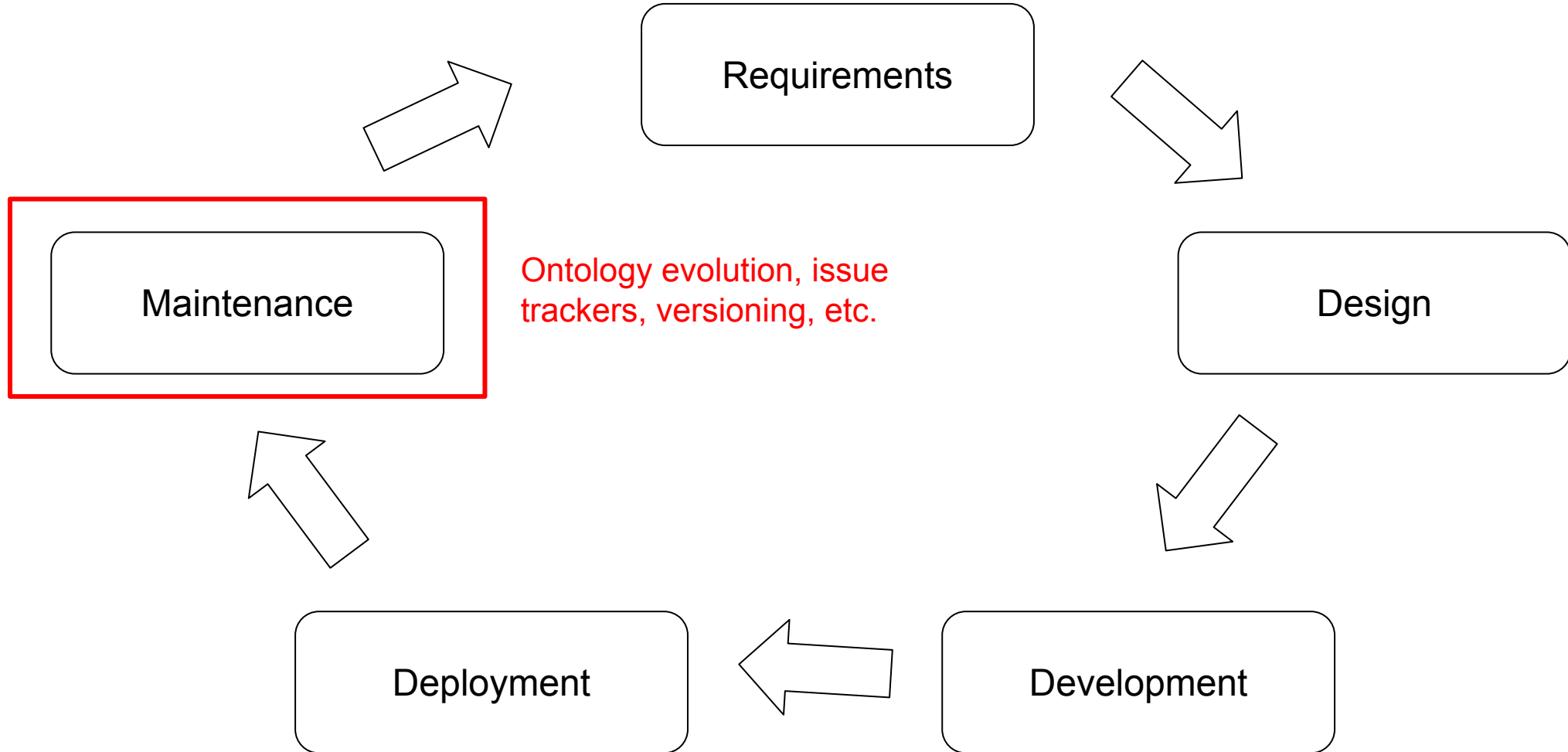
# Ontology engineering process

## *Standalone tools and Protégé plugins*



# Ontology engineering process

## *Standalone tools and Protégé plugins*



# Conclusion

As in software engineering, some aspects are more supported than others, and there is not one standard, common toolkit.

The tools are less commonly used than in software engineering however, and their learning curve are different.

They are also less integrated, making it more difficult to support the whole process.

What is lacking? How to make it easier? How to lower the entry barrier?

*You might want to attend the OntoCommons Workshop on Tools for*

*Ontology Engineering - <https://forms.gle/7LvJp3TZ8qNu1ZmHA>*

# Thank you for your attention

Questions?

<https://forms.gle/7LvJp3TZ8qNu1ZmHA>