Guidelines for the Creation of Semantic Models in the Internet of Production (IoP)

Author
Lina Teresa Molinas Comet

Reviewers
Patrick Sapel, M.Sc.
Iraklis Dimitriadis, M.Sc.

Contents

List of Figures

List of Tables

Version Control Table

Version	Author	Date	Changes
V 0.1	Lina Molinas Comet	28.07.2022	First draft
V 0.2	Lina Molinas Comet	01.08.2022	Add more catalogues
V 0.3	Lina Molinas Comet	03.08.2022	Modify methodology figures, add additional tools
V 0.3	Lina Molinas Comet	01.09.2022	Add hints in the workflow and additional examples

In case of questions regarding this guideline, please contact one of the following contact persons:

- Lina Teresa Molinas Comet
  - lina.molinas.comet@rwth-aachen.de or
  - linamolinascomet@protonmail.com
- Patrick Sapel Patrick.Sapel@ikv.rwth-aachen.de
- Iraklis Dimitriadis iraklis.dimitriadis@fit.fraunhofer.de

List of Abbreviations

CWA	Closed-World Assumption
DEs	Domain Experts
ER	Entity-relationship
KEs	Knowledge Engineers
I4.0	Industry 4.0
IoP	Internet of Production
ODPs	Ontology Design Patterns
OntoDevProcess	Ontology Development Process
ORSD	Ontology Requirements Specification
OWA	Open-World Assumption
OWL	Web Ontology Language
UNA	Unique Name Assumption
URI	Uniform Resource Identifier
W3C	The World Wide Web Consortium

1 Introduction

Industry 4.0 (I4.0) aims to enable autonomous coordination of processes and machines through interconnectivity and automation of processes [3], migrating from mass production to mass customization [17]. In this context, the Internet of Production (IoP) is considered to be the core of the visible rising I4.0. [22].

To achieve the goals of the I4.0, more specifically, to enable autonomous production, it is necessary to have a semantic model, which is a formal description of the knowledge related to, in this case, the production system, work-pieces, assets, the process involved, and the environment. Indeed, to obtain an optimal and valid knowledge modelling, both groups of actors, Domain Experts (DEs) and Knowledge Engineers (KEs), must work in cooperation complementing each other's work. In particular, this is necessary as DEs know the intricacies of the domain in the production system. However, they often lack knowledge about creating formal and syntactically correct semantic models. On the contrary, KEs understand how to formalize knowledge representation but do not have a deep understanding of the domain they want to model.

In Fig. **1**, we show a small example of how a semantic model represents the real world, in this case, part of the manufacturing process by modelling the objects, their attributes and the necessary constraints.

1.1 Motivation and Scope

The first requirement for the creation of semantic models helping to achieve the goals of the IoP is to have a guideline integrating enough information and directives on how to develop ontologies, which can be helpful for the actors involved in the process and in this way achieve coordination among them.

Even though there exist several methodologies for ontology development, there is no single guidebook or template that facilitates the modelling process and saves time for both groups of actors. Moreover, the approach of introducing existing methodologies is neither simple nor user-friendly.

The scope of this document is to provide an intuitive guideline in the form of a digital guide to help DEs to create semantic models, more specifically ontologies, mostly on their own but still collaborating with KEs. Also, we present an overview of existing domain-specific ontologies in the production and manufacturing domain and a catalogue of existing ontologies. Reusing existing ones instead of creating a new one from scratch will allow interoperability of the new ontology with existing ones. Our proposed guideline allows both DEs and KEs to work in a coordinated way to create and maintain ontologies efficiently.

To develop this guideline, we studied and unified best practices and existing methodologies used for ontology development. In this way, we support stakeholders who design ontologies to model knowledge on the Internet of Production (IoP).

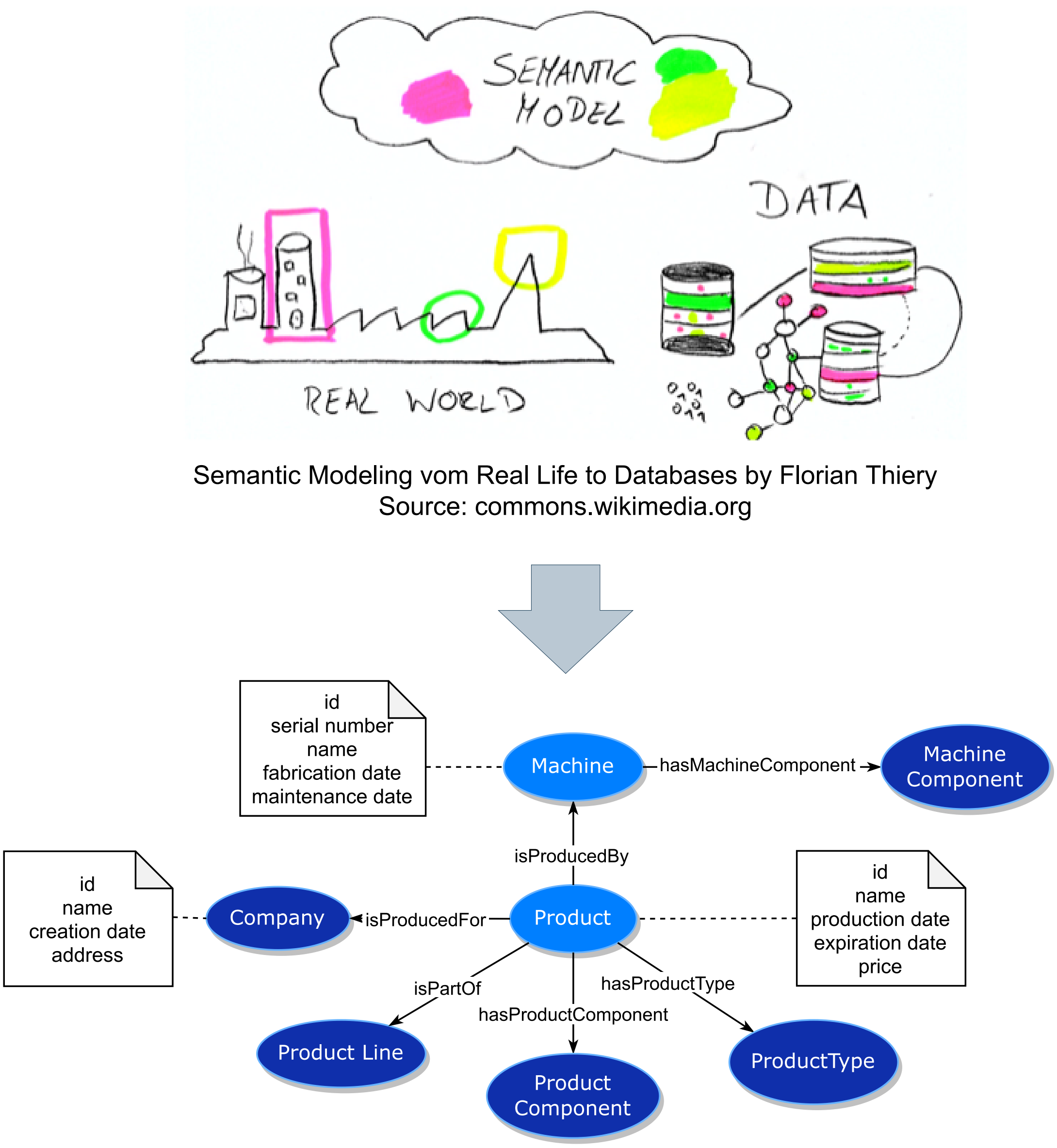


Figure 1. Semantics are needed to achieve the IoP

1.2 Audience and Use of this Guideline

The target audience of this document extends to those involved in the ontology development process, in particular:

- Domain Experts (DEs) in the manufacturing and production domains with little to no experience in Ontology Engineering. It will help them to create and maintain an ontology.
- Knowledge Experts (KEs) to support DEs in the ontology development process when needed.

We should remark that while this guideline aims to be a simple and basic resource, it is beneficial to gain some knowledge about basic concepts of ontologies. Therefore, we will explain those fundamental notions in Chapter 2.

1.3 Organization of this Guideline

Chapter **1** introduces the current problems with semantic modelling in the IoP, the target audience and the goal of this document. Chapter **2** presents the basic concepts and examples related to this guideline. We recommend that readers unfamiliar with the concepts of ontologies and semantic models first read the fundamental terminologies and explanations presented in this section because the rest of the document builds on top of it.

Chapter **3** presents the recommended methodology to create and maintain ontologies, a workflow to indicate the different stages of the process, and some additional resources to clarify concepts and how to proceed with the ontology creation. Then, in Chapter **4**, we enumerate some of the most common best practices to consider when building ontologies to assure interoperability and reusability of the ontology. In Chapter **5** we match the stages of the ontology development process with the corresponding tools. After that, in Chapter **6** we provide document templates that can be used to start the process and whose content later will translate into the ontology.

Finally, in Chapter **7** we suggest an architectural design on how the ontology would look by providing information about existing ontologies and a catalogue to localize some of these ontologies.

In this chapter, we introduce the topic, motivation and scope of this guideline. Also, we detailed the structure of this document. Additionally, for a better understanding, we present a summary of how to use this guideline in Fig. **2**.

One starts by identifying the use case we want to focus on and the need for using ontologies, then continues by learning the fundamental concepts about ontologies. After that, one focuses on the suggested methodology, best practices and tools to use and uses the proposed templates. Finally, using the information from this guideline, one builds a new ontology.

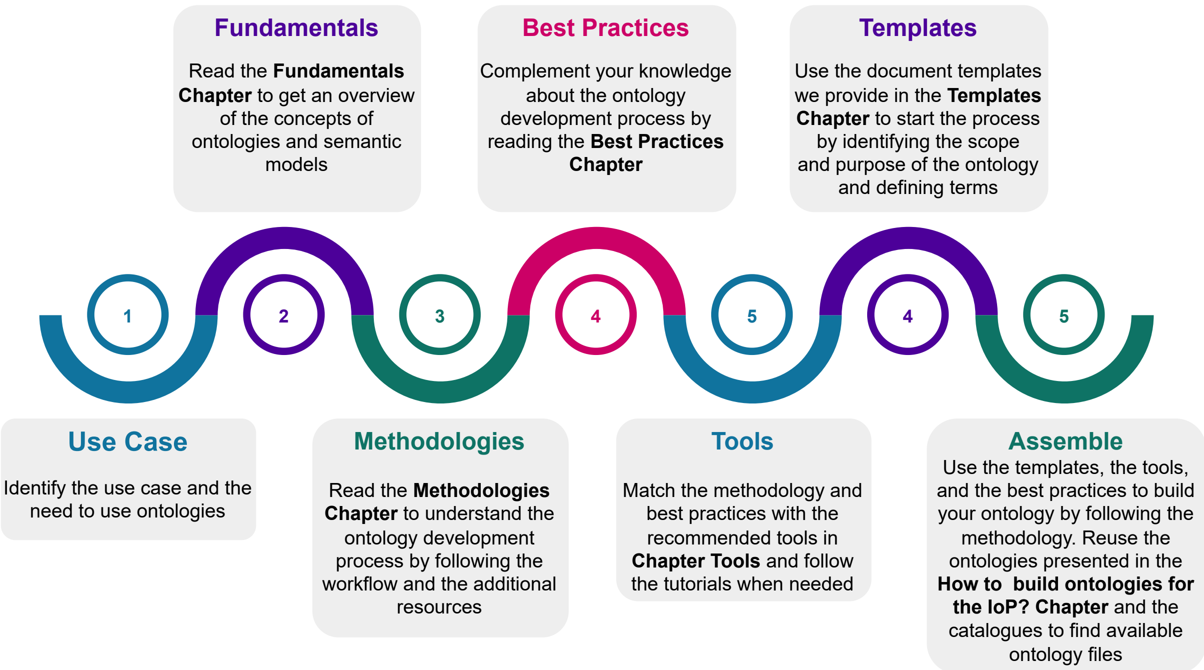


Figure 2. Our recommendation on how to use this guideline

1.4 Notation Conventions

We use the following formatting to express different parts of this document.

- We use figures, such as Fig. 1 to show examples graphically, and to introduce some concepts or structures.
- We use

> **Remark :** to express important considerations.

- We use

> **Note :** to express further comments and considerations.

- We use tables, such as Table 9 to present an overview of resources.
- We highlight concepts by using conceptual examples such as:

A machine has a machine component

- We highlight a word or phrase to give emphasis, for example, `a highlighted phrase`.
- We use references, such as Chapter 2 to refer and link to a specific resource, such as sections or figures in this document.

2 Fundamentals

2.1 Why are Semantic Models and Ontologies Important?

We motivated the creation of this guideline by indicating that semantic models are needed to achieve the goals of the I4.0 and IoP, such as autonomous production. Therefore, in this section, we first indicate why they are relevant in production and manufacturing. Later we explain their concepts and give concrete examples to support a better understanding of the concepts.

Nowadays, as a large amount of data in diverse formats is available from different and distributed sources, semantic models and ontologies are necessary for achieving data integration, and more specifically, semantic interoperability[1] [26]. It is feasible to attain this goal because ontologies help to specify a shared definition of concepts, and therefore a higher level of abstraction is achieved. Furthermore, it is possible to apply reasoning techniques to semantic models and extract knowledge, make predictions and support decision-making by implying knowledge [12, 17], as we will show in the examples in the upcoming sections.

2.2 How Can Ontologies Help in the Manufacturing and Production Domain?

As ontologies are a tool to define a shared definition of concepts, they assist different domains by allowing knowledge modelling and supporting big data analysis. In doing so, they improve flexibility, reusability and extensibility of the exchanged data [4, 13, 17]. The industry uses them to represent knowledge because they allow the description of entities and their relationship in an effective way. More specifically, in the domain of complex product design, ontologies contribute to innovation by successfully allowing knowledge sharing, internally and externally, in a company [27].

[1] Semantic interoperability refers to the ability of systems to share a common understanding of concepts

Then, using ontologies in the manufacturing and production domain can help with decision making, improving the quality of products and simulations, predicting machinery failures, etc. We will see in Section **7.3** that currently, different ontologies in these domains exist. They focus on various aspects and stages of the manufacturing process.

2.3 Semantic Models

As we have seen, the I4.0 aims to enable an interface connecting different components in the production system. To do so, it needs a semantic model describing and integrating the big data generated by that production or manufacturing system.

A semantic model is an abstraction of the real world in which one conceptually expresses objects, their properties and relationships. Semantic Models allow a common understanding of the modelled concepts by adding the necessary context (semantics) regarding their characteristics and environment.

In the example in Fig. **3**, we show how a small semantic model describes parts of the manufacturing process and its contextual information. We highlight in yellow the part of this figure we will focus on in the next part.

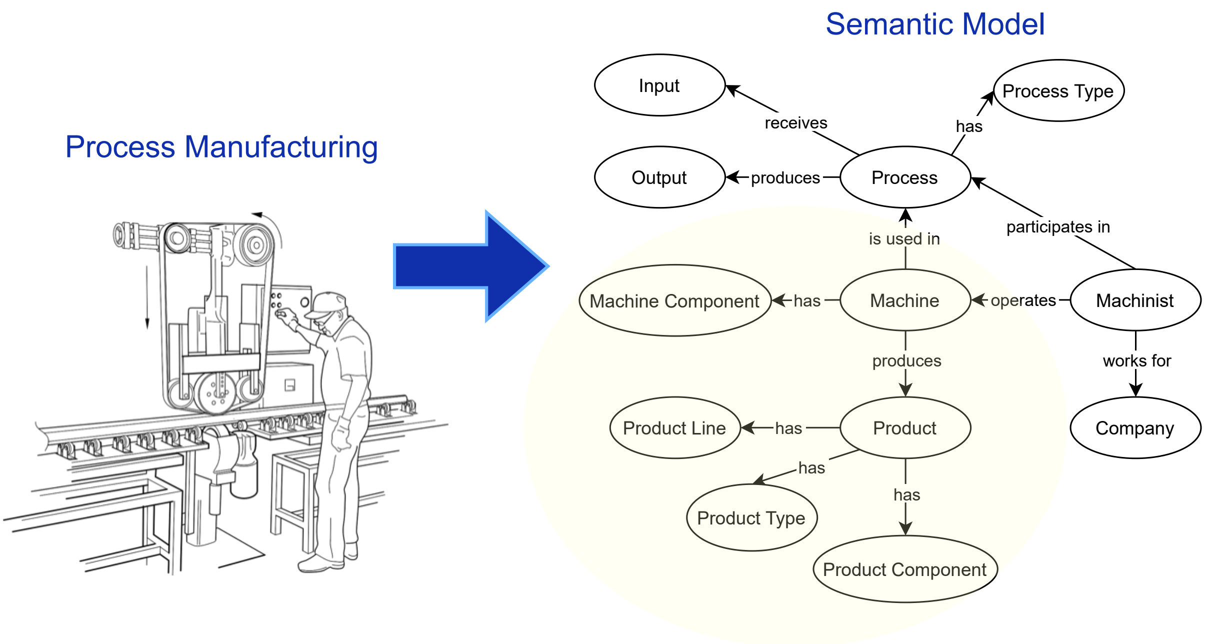


Figure 3. A semantic model describing the manufacturing process and its environment

In the example in Fig. **4**, we present a subset of this semantic model representing some concepts, also called classes of the manufacturing domain, in which we describe the following classes **Product, Product Line, Price, Machine, Machine Component** and **Company**.

Each class has a set of properties. For instance, the class **Machine** has the following properties: ***id, serial number, name, fabrication date, maintenance date***. Additionally, every two classes relate to each other through specific relations. Here we focus on two classes: **Machine** and **Product**.

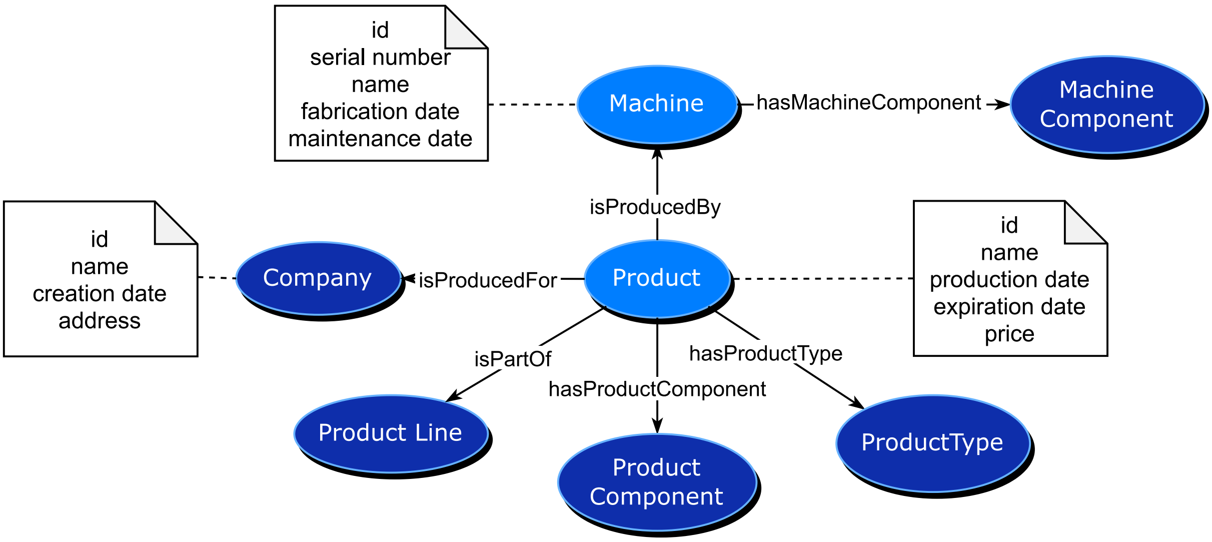


Figure 4. A semantic model as an abstraction of the concepts related to the manufacturing domain

Machine is linked to **MachineComponent** through the relation ***hasMachineComponent***, then it can be expressed that

A machine has a machine component

Also, **Machine** is connected to **Product** through the relation ***isProducedBy***. Then we know that:

A product is produced by a machine

If we contrast both semantic models presented in Figures **3** and **4** we can notice, for instance, that the relation between **Machine** and **Product** is expressed differently. In the first example the relation is called ***produces*** and it goes from **Machine** to **Product**, while in the second example it goes from **Product** to **Machine** and its called ***isProducedBy***. As we can see, it is an inverse relation expressing the same information. We want to achieve this kind of expressivity through a formal model, which can also help in inferring implicit knowledge. Therefore we need ontologies, which are then core components of semantic models.

Note : One should notice there is a difference between syntax and semantics when representing knowledge. We will see in the following sections, especially in **2.3**, how semantics add meanings on top of a structural model.

Remark : Semantic Web Technologies, which are standards defined by the The World Wide Web Consortium (W3C) are used to represent knowledge as semantic models. The main reason to use these standards, in our domain of interest is that, as reported in [10], they provide an effective semantic modelling solution to data integration and industrial diagnostics serving as an abstraction of industrial assets.

2.4 Ontologies

Even though the term *Ontology* has its origins in Philosophy, it is a well-known concept in the field of Knowledge Engineering and Information Science [13]. A short definition given by Gruber [5] is that "an ontology is a specification of a conceptualisation" and usually follows a hierarchy of the described concepts [7]. In another definition, ontologies are an explicit and formal description of terms in a given domain, the properties or roles of those concepts, and the existent restrictions (if they are) [12, 13].

2.4.1 What are Classes in an Ontology

Classes are the important terms one wants to describe. One uses classes in ontology to group similar resources and then to have a hierarchy. Let's consider the example in Fig. 5. Here the classes are **Machine**, **MachineComponent**, **Component** and **PhysicalObject**, **Product**, **ProductComponent**, and **ProductLine**.

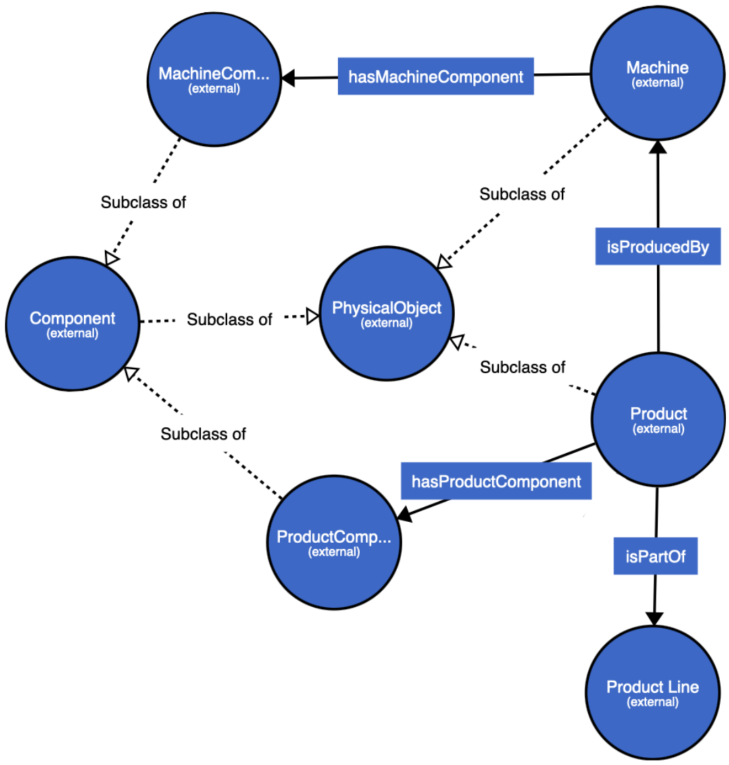


Figure 5. Examples of Classes

Here we also observe how to build a hierarchy, in this case consider:

> **Example 2.1.** A **Machine Component** is a subclass of a **Component**, and
> A **Machine** is a subclass of a **Physical Object**

Visually, the class hierarchy is as follows:

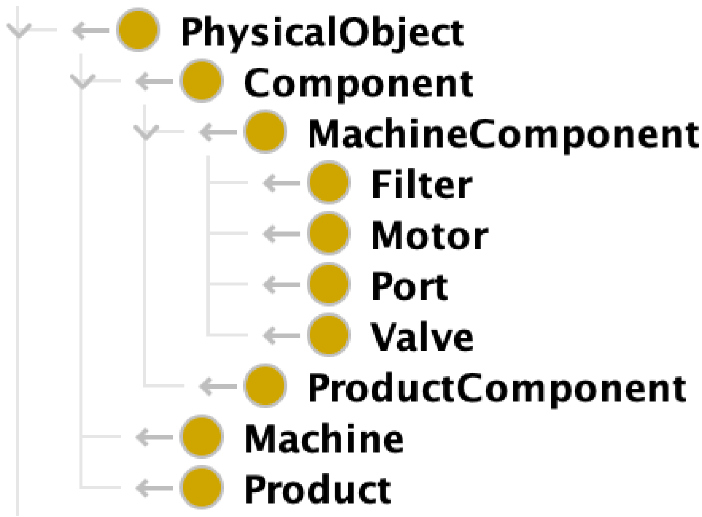


Figure 6. Class Hierarchy

The meaning here is that a Machine Component is a more specific type of Component. In this way, one can build a hierarchy of the classes in the ontology.

2.4.2 What are Properties in an Ontology

Properties, on the other hand are the linking attributes one wants to use to relate two classes (object properties) or a class and a value (data properties).

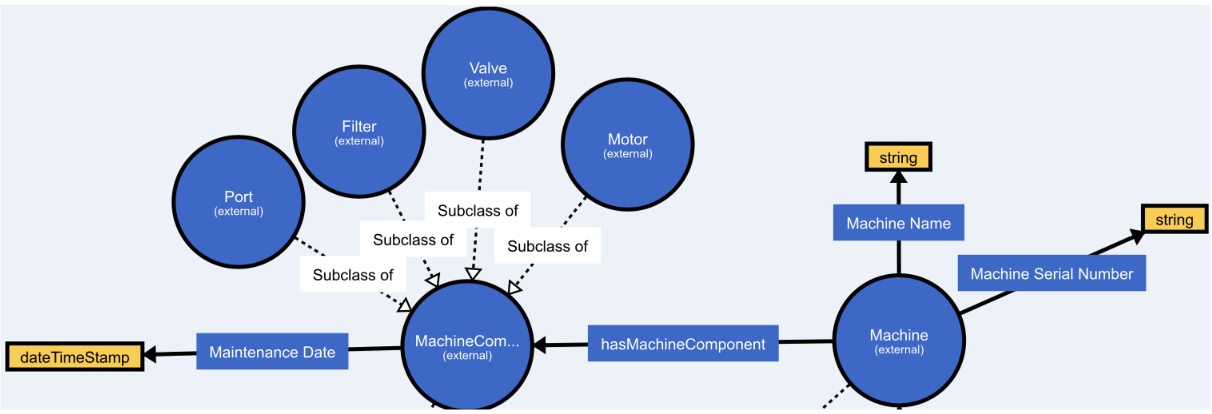


Figure 7. Examples of Data and Objects Properties

Now consider Fig. 7, which is a subset of the ontology we present in Fig. 10, one property here is *"hasMachineComponent"*. In this case, it is a object property as it is linking two classes in the ontology (Machine and MachineComponent).

Example 2.2. Machine *hasMachineComponent* MachineComponent

On the other hand, the property *"Maintenance Date"* is a data property, as it is linking a class (MachineComponent) with a value, in this case a data of type date/time. For example, consider the following:

Example 2.3. A Machine Component *hasMaintenanceDate* "2017-04-12T10:30:00"

2.4.3 What are Domains and Ranges in the Elements of an Ontology

The domain is the class that is the subject of what we are describing by using that property, and the range is the object. In other words, they better indicate the relation between the elements that use a specific property and allow inference.

Following the previous examples, consider Fig. 8. In this case, we can see that the domain of the property is **Machine**, and the range is **MachineComponent**. The meaning of such modelling is that a Machine has a machine component, or say differently, that a machine component belongs to a machine.

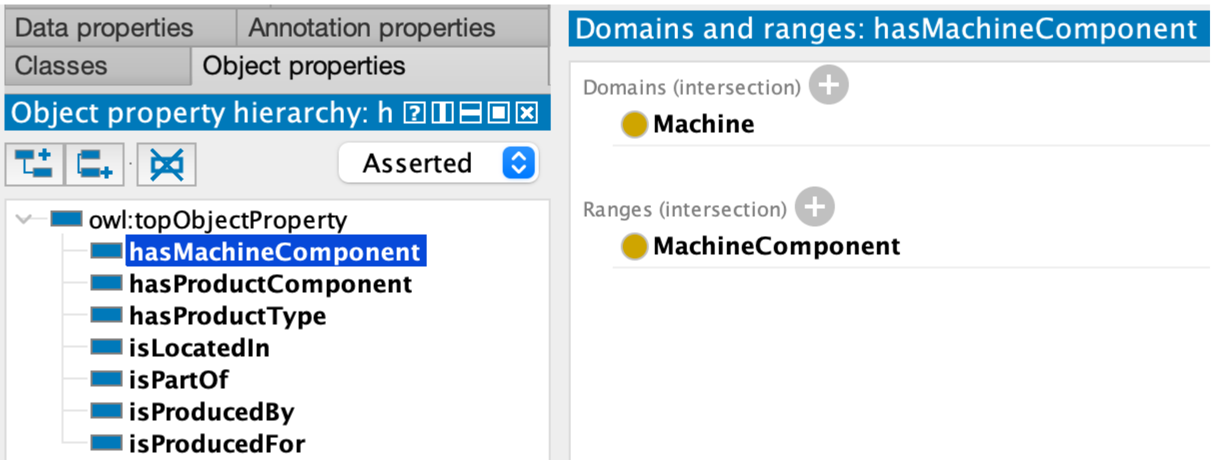


Figure 8. Examples of Domain and Range in an Object Property

In the case of a data property, consider again the example "Maintenance Date". We said, it is a data property, and as we see in Fig. 9, the domain now is a class (MachineComponent), but the range is a value, in this case of type datetime. Then, this modelling is expressing that a component of a machine undergoes maintenance on a given date.

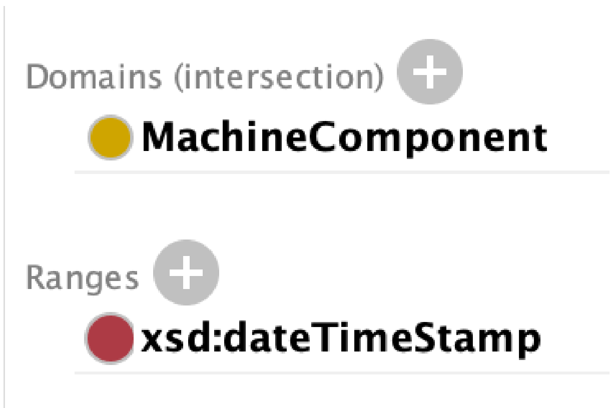


Figure 9. Examples of Domain and Range in a Data Property

Also, let's consider the following piece of knowledge:

Example 2.4. A Product is made of Polyethylene terephthalate (PET)

Here, the property would be *"isMadeOf"*, while **PET** and **PETBottle** are the classes. Now, by looking at the structure of the sentence we are using to express some knowledge, we see that **PETBottle** is the subject of the sentence, *isMadeOf* is the predicate, and **PET** is the object.

Likewise, we can say that the **domain** (subject), then is **PETBottle**, and the **range** (object) is **PET**.

2.4.4 What are Individuals in an Ontology

One can associate each class with a set of individuals. The individuals in the class are called the instances of the class and they are the most specialised or specific concepts one wants to model.

Example 2.5. A PET bottle is produced by an Injection Blow Moulding Machine

Here, PET bottle is an instance of the class Product, and Injection Blow Moulding Machine is an instance of the class Machine.

To illustrate the above-given definitions, we present an example in Fig. 10 in a graph form, in which we follow the previously introduced examples. Here we can notice two types of relationships between concepts. One of these types expresses hierarchical relations by indicating that a class is a subclass of (***SubclassOf***) another one. The other expresses the specific connection between concepts, for instance ***isProducedBy*** and ***hasMachineComponent***.

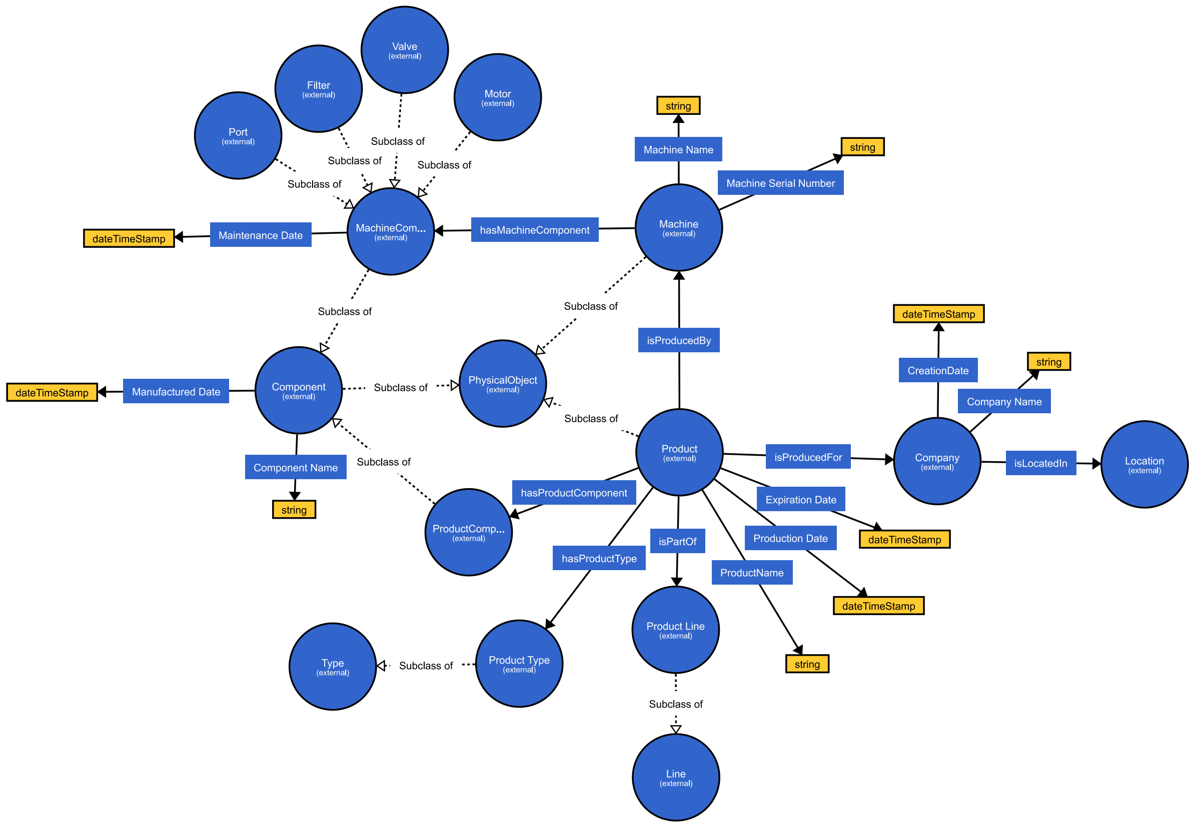


Figure 10. An ontology representing a subset of knowledge related to the manufacturing domain by definition of concepts and their relationships

> **Note :** We used the WebVOWL tool to generate the graphical representation of the small ontology presented in Fig. 10. We introduce WebVOWL in Chapter 5.

By following the example in Fig. 4, we can observe how the ontology in Fig. 10 also includes the representation of that knowledge. Now if we focus again on the relations ***isProducedBy*** and ***hasMachineComponent*** between the classes **Product** and **Machine**, and **Machine** and **MachineComponent** respectively, we can say that the same might be expressed by the inverse relations ***produces*** and ***belongsTo***. It is possible to reach this kind of conclusion by applying inference tools to an ontology. Then it is possible to infer the following knowledge from the ontology:

A machine produces a product

A machine component belongs to a machine

In this context, we need to remark that when modelling ontologies, it is essential to consider the concepts of Closed-World Assumption (CWA) and Open-World Assumption (OWA). CWA assumes complete information, i.e., it is the assumption that what is not known to be `True` is `False` [8]. On the other hand, OWA is the assumption that anything that is not known (to be true or false) is considered *unknown*, but not false [18]. Therefore, it assumes incomplete information [9], and the distinction between something being `False` and something being `Unknown` is possible [25].

We can argue that it is possible to infer knowledge from the ontology without the higher complexity needed in the Entity-relationship (ER) model to achieve similar results. In other words, it is possible to join several tables in a database (corresponding to the ER model) to get similar knowledge provided by ontologies, but it requires more effort. The reason is that ontologies add semantics by explicitly defining relations between classes, and in this way, allowing interoperability and more expressiveness [18]. Moreover, the main difference between ER models and ontologies is that while the former uses CWA and adds constraints, the latter benefits from OWA. In this context, there are some considerations, such as those presented in [25], to take into account when considering CWA or OWA for inferring knowledge.

To clarify the concepts of CWA and OWA, let's look at the two main differences between them when it comes to answering direct questions and inferring new knowledge [23]. For this, we will consider Fig. **11**, where we show a subset of the ontology introduced in Example **10**, and assume the following facts (axioms) are part of the model in our ontology and in a relational database:

1. A PET bottle is a product
2. A PET bottle is made of PET
3. PET is a recyclable plastic
4. HDPE is a recyclable plastic
5. A product is produced only by one machine
6. An extrusion blow moulding machine is a machine
7. An injection blow moulding machine is a machine
8. A PET bottle is produced by an injection blow moulding machine
9. A machine component is a component
10. A component is a physical object
11. Anything that is made of recyclable plastic is recyclable

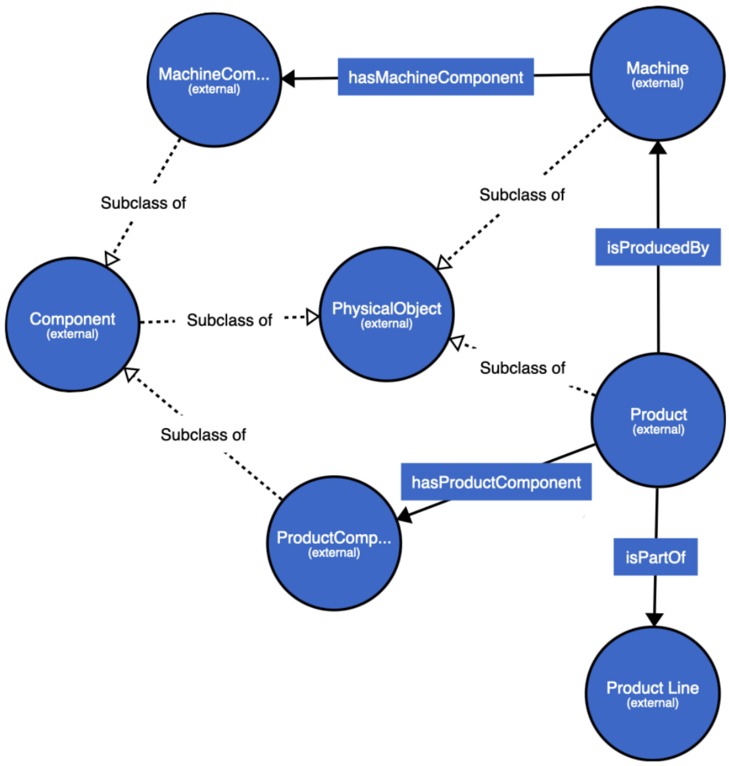
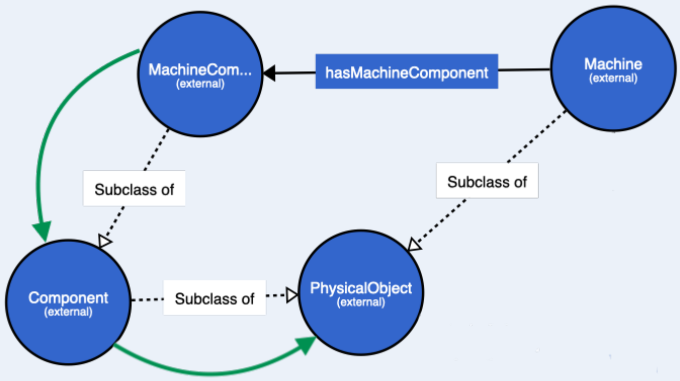


Figure 11. Subset of the Ontology presented in Example **10**

Regarding answering direct questions, consider we ask `"Is there a product called PET bottle?"`. The answer in both, relational databases (considering the CWA) and ontology (considering the OWA) will be `"Yes"` because such query will return the answer `True` considering the fact **1**. When we ask `"Are PET and HDPE recyclable plastics?"`, we will again get the answer `"Yes"`, as it is possible to answer this question directly by the facts **3** and **4**. However, if we now ask `"Is there a product called PP bottle?"`, we expect to get the answer `"No"` in relational databases because there is no information about such a product in our model (defined facts). That would be the case because the associated query will return `False`. On the other hand, in the ontology, instead of getting the answer `"No"`, we expect to get the answer `"Unknown"`. That is the correct behaviour in this scenario because the ontology would need more facts to disprove the assumption that such a product exists, i.e., to demonstrate that a product with that name does not exist, as it assumes the OWA.

Next, suppose we ask `"Is a machine component a physical object?"`. In the ontology (OWA), the answer will be `"Yes"`, and as shown in Example **2.6**, the explanation will result from the transitivity property of the relations between concepts, i.e., as a Machine Component is a subclass of Component (related to fact **9**), and Component is a subclass of Physical Object (related to fact **10**), it must be the case that a Machine Component is a Physical Object. We show these relations in the example by the green arrows. Likewise, it would be possible to reach that conclusion by a query joining the corresponding tables in relational databases, as there is no explicit meaning in the relationships.

Example 2.6. (MachineComponent subClassOf Component) *AND*
(Component subClassOf PhysicalObject) *IMPLIES THAT*
(MachineComponentsubClassOf PhysicalObject)



Now, regarding knowledge inference capabilities of an ontology, consider our model under the facts expressed by facts **5** and **8**. If we want to add a new fact *"A PET bottle is produced by an extrusion blow moulding machine"*, we would get two different scenarios, depending on if we consider the CWA or the OWA, as we describe below.

In the CWA as soon as we want to add the new fact, we would get an error by contradiction to our initial assumption that a product is produced only by one machine (fact **5**).

On the contrary, in the OWA instead of an error, this new statement would let the ontology infer a piece of new knowledge. That is, `"if a product is produced only by one machine, and if PET bottle is produced by an injection blow moulding machine and an extrusion blow moulding machine, then injection blow moulding machine and extrusion blow moulding machine must be referring to the same concept"`. The ontology reaches such a conclusion because the de facto language for ontology creation, Web Ontology Language (OWL), does not make the Unique Name Assumption (UNA)[2]. Thus, to avoid such behaviour, one should explicitly state that two resources with distinct Uniform Resource Identifier (URI)s [3] are different.

Now consider we ask: `"Is PET bottle recyclable?"` The answer in the relational database and the ontology would be `"Yes"` by the inference that is a PET Bottle is made of PET and PET is a recyclable plastic, then PET Bottle must be also a recyclable plastic.

However, if we ask `"Is HDPE bottle recyclable?"` in a relational database, the answer would be `"No"` because there is no information in the list of facts indicating such knowledge. On the other hand, in an ontology, the answer, by fact **11**, would be that there could be a model satisfying such conclusion, for example, by adding a new fact *"HDPE bottle is made of HDPE"*. Because we know, by fact **4** that HDPE is a recyclable plastic and anything made of recyclable plastic is recyclable.

[2]The Unique Name Assumption indicates that different names always refer to different entities in the world [21]

[3]An URI is a unique identifier that enables uniform identification of logical and physical resources [2]

2.5 Classification of Ontologies

An important consideration regarding ontologies is that they are classified depending on their coverage. Here, we describe the following:

- **Upper Ontologies:** also called "Top-Level" ontologies, are highly generic and represent the concepts, categories and relations common to various domains.
- **Domain Ontologies:** they represent concepts related to a specific domain, such as objects, tasks, activities, and processes in that domain.
- **New Ontology:** refers to the new ontology one is currently creating to model a specific use case.
- **Support Ontologies:** are the ontologies that can be used cross-sectionally, i.e., across different domains, for example to express organizational structures.

In Fig. 12 we illustrate the different ontology levels relevant for creating reusable and interoperable ontologies in our domain of interest. Upper ontologies are at the bottom of the figure, depicting that this type of ontologies is the base of the ontology construction system. Then, on top of that, one adds middle or domain ontologies. Finally, one adds the concepts of the new ontology. At the same time, supporting ontologies, which represent cross-domain concepts, should be considered. This representation means that one should reuse as much as possible the already defined concepts from other ontologies and newly define those specific for the use case one is modelling.

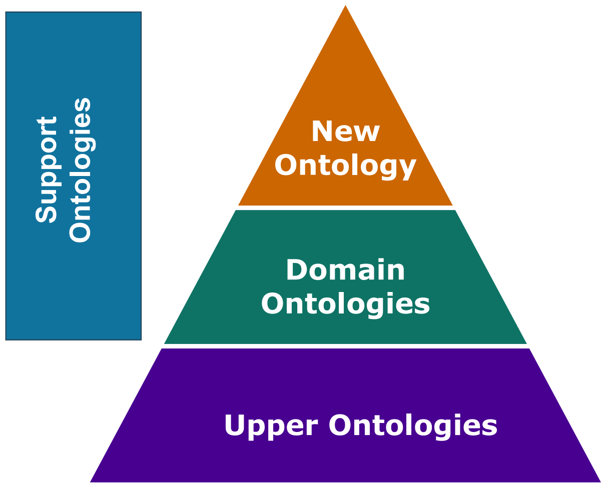


Figure 12. Ontology Composition

> **Remark :** It is crucial to consider reusing existing ontologies to ensure interoperability among ontologies. Therefore, it is relevant to understand their coverage and when to reuse them.

2.6 Ontology Development Process

To start the ontology development process, we first choose a methodology and adequate tools for the tasks at hand and adopt the best practices to tackle this task.

In Fig. 13, we show the most common steps to follow in the process, after choosing the methodology to follow and the tools to use.

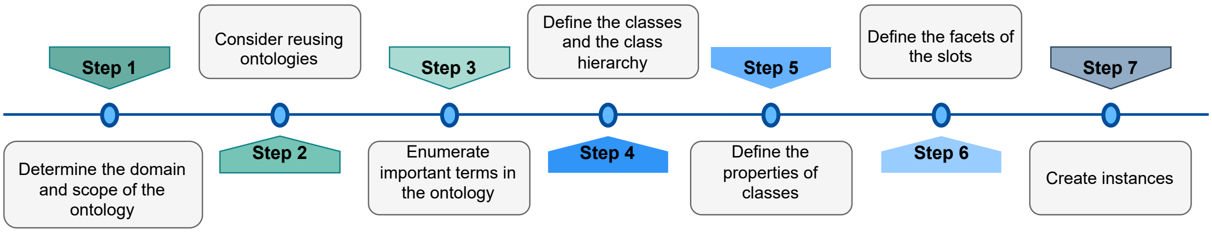


Figure 13. Iterative steps required to develop an ontology, based on [12]

3 Methodologies

There are several methodologies for ontology development, and as we explained in Chapter 1, one of the current problems in building an ontology is selecting an adequate methodology. Even though the steps presented in Fig. 13 are relevant to start building an ontology, a more comprehensive methodology is necessary.

3.1 Recommended Ontology Development Methodology

We recommend using the **Linked Open Terms (LOT)** methodology introduced in [15], which is available at: `https://lot.linkeddata.es`. We recommend it because it is a reasonably comprehensive methodology, and as it is an evolving approach, it covers several aspects to consider during the Ontology Development Process (OntoDevProcess). The methodology focuses on engineering aspects and offers valuable insights to engineers from different domains. Moreover, it recommends tools and best practices and covers the whole OntoDevProcess including the later stages, namely the evaluation, documentation and maintenance of the ontology.

The detailed documentation of the LOT methodology is available here: LOT: An industrial-oriented ontology engineering framework. Moreover, in the Appendix, in Section **A.1** we show the original LOT methodology views, defined by [15].

3.2 Workflow for the Ontology Development Process

For the creation and maintenance of ontologies, we recommend following the workflow presented in Fig. 14 for the creation and maintenance of the ontology, which covers the stages of the LOT methodology.

> **Remark :** We recommend to schedule an initial meeting between DEs and KEs before starting the Ontology Development Process to clarify ideas and expectations.

> **Remark :** When following the steps of the workflow, follow the best practices we name in Section 4 and the additional resources from Section 3.3.

3.2.1 Description of Each Step in the Workflow

1. **Identify and specify the use case:** start the process by identifying and defining the use case for which you want to build the ontology.
   Answer the questions:
   - What is the name of the use case?
   - What is this use case about?

   > **Hint:** Use the template *Template_OntologyRequirementsSpecification* provided in Section **6**.

2. **Identify and define scope and purpose:** determine the scope and purpose of the use case and the corresponding ontology.
   Answer the questions:
   - What are the goals of this use case?
   - What is the benefit of using ontologies for my use case?
   - Who will use this ontology and use case?
   - Who is responsible for defining the requirements of this use case?

   > **Hint:** Use the template *Template_OntologyRequirementsSpecification* together with *Template_CompetencyQuestions_Definition* provided in Section **6**.

3. **Define competency questions:** determine how the ontology will help to answer key questions.
   Answer the questions:
   - What are the relevant questions the ontology should answer?
   - What information needs to be expressed in the model to help us achieve the goals of this use case?

   > **Hint:** Use the template **"Template_CompetencyQuestions_Definition"** provided in Section **6**.

4. **Verify and refine competency questions:** check the list of competency questions and evaluate their consistency and completeness.
   Answer the questions:
   - Did we include all the essential questions the ontology should answer?
   - Did we precisely formulate the questions we are interested in?
   - Can we improve these competency questions?

   > **Hint:** Use the template *Template_CompetencyQuestions_Definition* provided in Section **6**.

   (a) If changes are necessary to the competency questions, modify and refine them on demand.
   (b) If no changes are needed, go to the next step.

5. **Conceptualize the ontology:** based on the terms one identifies as important for the model, define which of them are classes, properties, and individuals.
   Answer the questions:
   - What are the necessary terms we need to include?
   - What are the constraints and considerations in this use case?

> **Hint:** Consider the information we presented in Section **2.4** about Classes, Properties, and Individuals. Use the information you included on the documents based on templates *Template_OntologyRequirementsSpecification* and *Template_CompetencyQuestions_Definition* to identify the important terms you need to model in your ontology. Once you identified the terms, use the templates *Template_Class_Definition*, *Template_Property_Definition*, and *Template_Individuals_Definition* from Section **6**.

6. **Consider reusing existing ontologies:** once you identify the scope of the ontology and the required terms, look for information about existing ontologies covering a similar domain, and reuse them when needed. Also, consider reusing upper (top-level) ontologies.
   Answer the questions:
   - Is there an ontology covering our whole use case?
   - Is there an ontology partially covering our use case?

> **Hint:** Consider the information presented in Section **7** and identify which terms you want to model in your ontology are already included in exiting ontologies we list in Tables **7**, **8**, and in Section **7.3**. You can look at information about ontologies in the resources we provide in Table **9**.

7. **Formalize Ontology Requirements Specification (ORSD) and competency questions:** with the help of the editor tool, start creating the identified classes, properties, individuals, and the necessary relationships among them.
   Answer the questions:
   - Did we model all the identified entities?
   - Did we express all the needed relationships and constraints?

> **Hint:** Consider the information presented in Section **5** and use Protégé as the editor tool to start modelling your ontology. To visualise the ontology you are building, use WebVOWL.

8. **Evaluate the ontology:** with the help of the evaluation tool, verify the resulting model.
   Answer the questions:
   - Are there any mistakes in our modelling?
   - Did we forget to include some elements?
   - Does the model make sense regarding the use case?
   - Is the model able to answer our competency questions?

> **Hint:** Consider the information presented in Section **5** and use OOPS! as the tool to evaluate the resulting ontology.

   (a) If, based on the evaluation process, one needs to make changes to the ontology, there are two possibilities:
      - If one needs to change the ontology at the specification level, then one has to redefine the classes, properties, individuals, relationships or constraints (Go to step **5**) and proceed from there onwards.
      - If changes are only necessary at the knowledge formalization level, then one has to make the necessary changes to the ontology with the help of the editor (Go to step **7**) and proceed onwards.

   (b) If the evaluation process was successful, and no changes are needed, go to the next step.

9. **Document the ontology:** with the help of the recommended tool, one can automatically generate the documentation of the ontology as a HTML file.

> **Hint:** Consider the information presented in Section **5** and use Widoco as the tool to generate the documentation for the ontology.

10. **Publish the ontology:** one can take the HTML file generated in the previous step and publish it internally (in an Intranet) or make it public (on the Internet).
11. **Bugs or Improvements:** once the ontology is in use, it can be the case that the ontology needs changes. There are two reasons for changes at this stage:
    - One finds bugs, such as mistakes in the modelling or inconsistencies.
    - One detects improvements in the ontology, such as adding, modifying or deleting terms based on new requirements.

    The steps to follow to maintain the ontology at this point are the following:
    (a) In the first case, one detects bugs, go to step **5** if it is necessary to define some new classes, properties, individuals or relationships based on existing requirements and proceed onwards. Go to step **7** if the changes are only necessary at the formalization level, and proceed onwards.
    (b) If new requirements are detected, go to step **3** define new competency questions if needed, and proceed onwards. If no new competency questions are involved, go to step **5** and proceed onwards.

> **Remark :** Through steps **5** to **8**, the Knowledge Engineers offer support to Domain Experts to formalize the required knowledge. They also help in the quality evaluation of the resulting ontology by considering the use case and defined competency questions. The Knowledge Engineers also offer support at steps **9** and **10** for the documentation and publication of the ontology.

3.3 Additional Resources

For more detailed information and examples about the initial ontology development steps, we recommend reading the *Ontology 101* report. Moreover, it is relevant to consider the resources in the *NeOn Book* [14], which we present here to help to go more in-depth on a specific topic.

Additional Informational Resources
Ontology101 (How to start and examples)
Ontology Requirements Specification
Reusing general ontologies
Reusing domain ontologies
How to reuse ontology statements
How to model using Ontology Design Patterns (ODPs)
How to approach ontology evaluation
How to approach ontology modularization
How to approach ontology evolution
How to align ontologies in a network of ontologies
Complete NeOn Book with examples of large ontologies use cases

Table 1. Additional resources to consider when developing ontologies

Moreover, the LOT methodology provides helpful links in the **"Recommended reads"** section associated with each activity (ontology requirements specification, ontology implementation, ontology publication, and ontology maintenance) in `https://lot.linkeddata.es`.

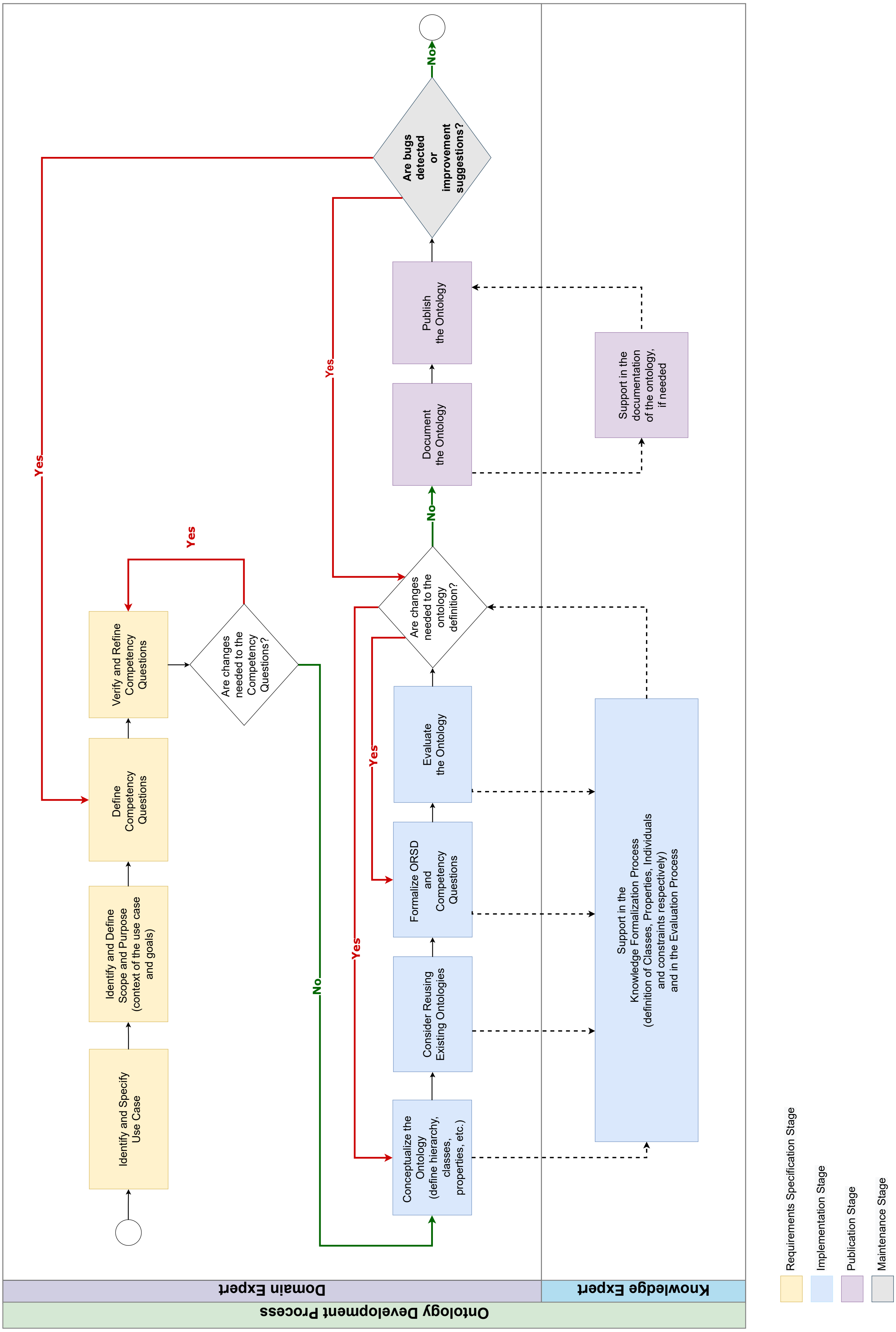


Figure 14. Ontology Development Workflow, our own definition based on the LOT methodology [15]

4 Best Practices

Here we enumerate the most common best practices to consider when building an ontology, based on the existing literature.

1. **Decide** the following:
   - Do we need to add a new class?
   - Do we need to model this entity as a new class or a property value?
   - Do we need to model a disjointness of classes?
   - Is our class hierarchy correct and reflecting our use case?
2. **Reuse ontologies** instead of building a new one entirely from scratch, reusing them results in time-saving and improvement in the interoperability of the new ontology with the existing ones.
3. **Consider using Ontology Design Patterns (ODPs)**, as they are small ontology building blocks.
4. **Consider the quality requirements the new ontology should meet**. For instance, the ontology should be concise, modular, adaptable to different granularities and perspectives, and highly reusable while focusing on modelling only one essential notion of a domain.
5. **Focus on upper (top-level) ontologies** to support consistent ontology development across domains.
6. **Adopt the *Top-Down* approach**, in which one defines first the most general concepts in the domain as a data schema and then applies gradual refinement of terms.
7. **Adopt a modular approach**, especially when designing large ontologies. It helps to facilitate reuse, extensibility and maintainability of the ontology.
8. **Specify disjointness of classes** to help in the validation process. Two classes are disjoint if they cannot have any instances in common.
9. **Consistency in classes name**, choose to use singular or plural when naming classes and keep it throughout the whole ontology creation.
10. **Avoid class cycles**, pay attention when creating a class B as a subclass of A and then defining B as a superclass of A.
11. **Follow name conventions**
    - Capitalize the first letter of the class name and use lower case for naming the properties, e.g. *Machine*, and *produces*.
    - When a concept name has more than one word, consider the following: either capitalize the first letter of each new word, or use an underscore, e.g. *MachineComponent*, and *Machine_Component*.
    - Avoid abbreviations in naming classes, properties, and individuals, e.g. *Machine* instead of *Mach*.
    - Use prefixes in the property names, e.g. *hasPart*.

> **Remark :** Regarding the class hierarchy in the ontology, presented in point **1**, recall that as *ontology* is a philosophical concept, it is fundamental to understand the intentional structure of the things one describes through an ontology as they help appropriately classify concepts. The intentional structures one should consider are: *1)* the intention of identifying things *(continuants)* and *2)* the intention of explaining things *(occurrents)* [11].

> **Remark :** Additionally, there are other specific modelling principles to consider when focusing on ontologies in the production and manufacturing domain, such as those presented in [1, 17, 27]. Also, there are more general best practices in [20] to help avoid the creation of heterogeneous and non-reusable ontologies.

How to use Ontology Design Patterns?

Regarding the best practice in point **3**, ODPs are a reusable solution for recurrent modelling problems [6, 24] and help to tackle the ontological commitment problem [4]. They should be flexible, easily understandable and generic to support similar problems.

To better understand the concept of ODPs, we present a small use case in Fig. **15**, based on the work presented in [16], in which we show how to use the *"part of"* pattern. This pattern corresponds to the Content ODPs.

In the scenario we describe here, **Filter**, **Valve**, **Motor** and **Port** are all components of the **Engine Stretch Blow Moulding**, which is expressed by the relation *owl:unionOf*[5] combined with the relation *partof:hasPart*. At the same time, **Engine Stretch Blow Moulding** is part of **Machine Stretch Moulding**, we know that by the relation *partof:hasPart* between those two entities. Therefore, by transitivity property of the relationships among entities, we see that it can be inferred that **Filter**, **Valve**, **Motor** and **Port** are also part of **Machine Stretch Moulding** (blue dotted lines).

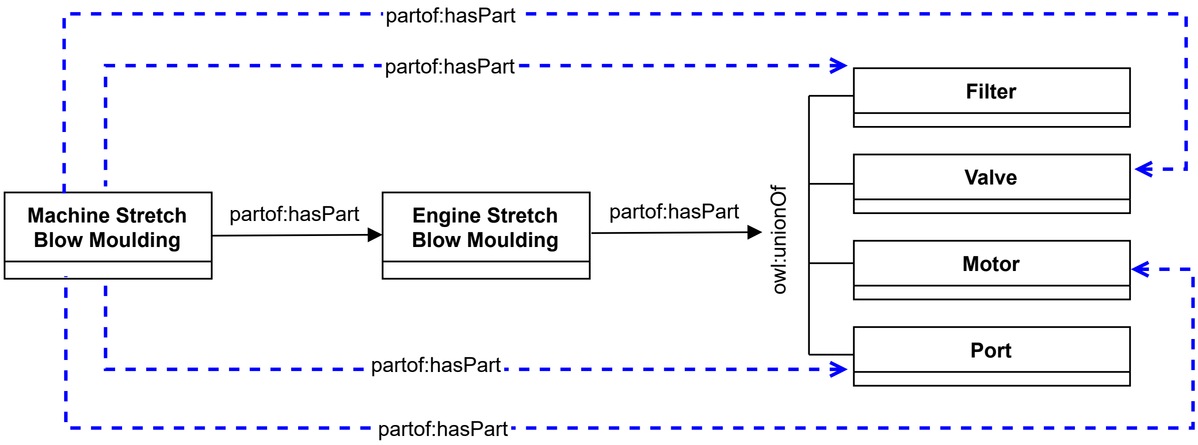


Figure 15. An example of the Content ODPs "part of"

Note : In Section **3.3** we provide a link to the document *How to model using ODPs*.

[4]Ontological commitment focuses on identifying the entities to be modelled into an ontology [19]
[5]https://www.w3.org/TR/owl-ref/#unionOf-def

5 Tools

There are several tools to be considered when developing ontologies. Here we recommend the required tools to create an ontology. In Fig. 16 we name these tools corresponding to each stage of the ontology development.

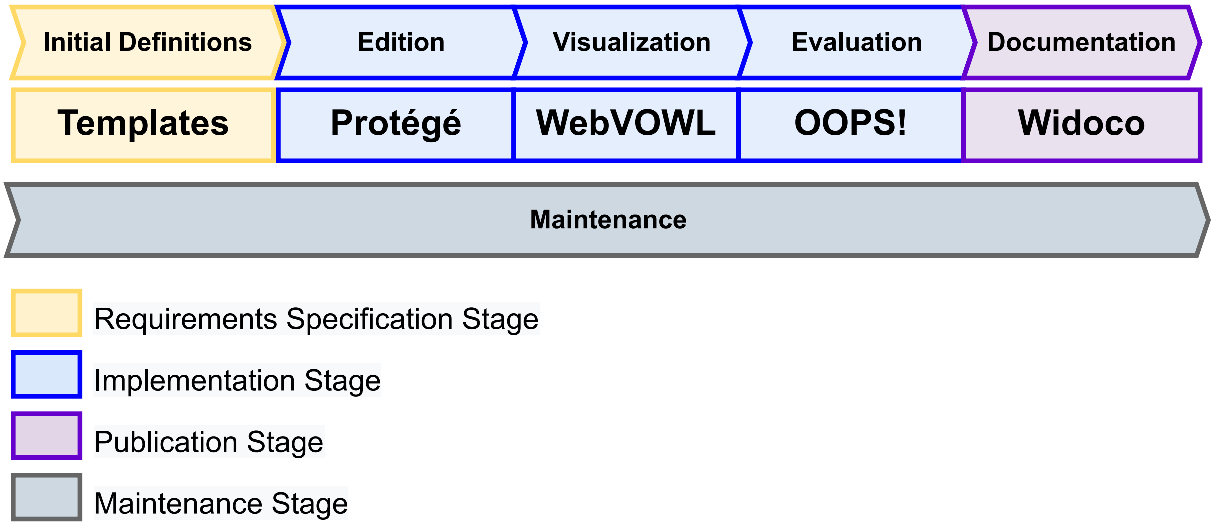


Figure 16. Tools needed at each stage of the ontology development process

> **Remark :** In Section 5.3 we present tools that one should consider when developing ontologies. They are not mandatory for starting the modelling process. However, knowing and understanding them brings additional benefits.

5.1 Tool Tutorials and Additional Resources

In Table 2 we provide the necessary links to tutorials and additional resources to reference when using these tools.

Tool Resources and Tutorials
Getting started with Protégé
Protégé Tutorial
How to import ontologies in Protégé?
How to import only some classes and properties in Protégé?
Documentation and Notations of VOWL, WebVOWL
OOPS! User Guide
Widoco Tutorial

Table 2. Tool resources and tutorials

5.2 Where to Find the Recommended Tools

Additionally, in Table 3, we provide the links for finding and downloading the tools, if necessary.

Where to find the tools
Templates
Protégé Desktop
Protégé Web
WebVOWL
OOPS!
Widoco

Table 3. Finding the tools

> **Remark :** Regarding *Protégé* as an ontology creation and editing tool, we recommend using the desktop version, as one can extend it with plugins. There is also a web version which is a simplified version of the tool.

5.3 Further Tools to Advance the Ontology Development Process

In this section, we present some additional tools, which we recommend to the reader for enhancing their knowledge regarding the ontology development process. Such tools are the underlying standard languages for ontologies construction, ontology querying languages tools, versioning control tools, and languages for modelling constraints to validate the quality of the resulting ontology.

In Table 4 we list these tools. The content in the first column is a clickable link to the tools documentation.

Advanced Tools	Description
Resource Description Framework (RDF)	A framework for representing information in the Web, which serve as a base for all RDF-based languages
RDF Schema (RDFS)	Data-modelling vocabulary for RDF data
Web Ontology Language (OWL 2)	An ontology language used to formally define the meaning of concepts
Shapes Constraint Language (SHACL)	A data validation language to validate RDF graphs against user-defined conditions
Shape Expressions Language (ShEx)	A data validation and generation language to describe RDF graphs and nodes structures
SPARQL Query Language	A graph querying language. The query results can be expressed as sets or RDF graphs
GitLab	Version control tool. It is an open source code repository and collaborative development tool
GitHub	Version control tool. It is a code repository and collaborative development tool

Table 4. Advanced tools - Documentation

To complement the information and help the reader, we offer a starting point by providing links in Table 5 to some relevant tutorials introducing these more advanced tools.

Advanced Tools - Tutorials
Introducing RDF
Introduction to RDFS
OWL Tutorial
SPARQL Book
Querying Semantic Data
Introduction to RDF and SPARQL
Introducing RDFS and OWL
Introduction to SHACL
ShEx by Example
Videotutorial - Git and GitHub for Beginners
GitHub Quickstart
A step-by-step guide to Git
Getting started with GitHub Desktop
Learn GitLab with tutorials

Table 5. Advanced Tools - Tutorials

In Table 6 we list a broader set of tools to support the ontology development process. We consider these tools relevant because they provide visual support to create queries or to generate ontologies from diagrams. They can used together with the main tools we introduced at the beginning of this section.

Additional Tools	Description
SPARQL Playground	A tool for learning and testing SPARQL
SparqlBlocks	A library to visually build SPARQL queries using blocks
Sparnatural	A visual tool to build SPARQL queries
Chowlk Converter	A tool to generate ontologies in OWL from a conceptualization diagram
Neologism 2.0	A visual tool for quick vocabulary creation
FrODO	A tool for automatically drafting ontologies from competency questions
CoModIDE	A plugin for Protégé for graphical composition of ontologies
ChImp	A plugin for Protégé to verify changes in the ontologies and visually analyse their impact

Table 6. Additional Tools

6 Templates

To download the templates, please refer to the following link to a Google Drive Folder: GuidelineForCreationSemanticModelsIoP, in which the following templates are available: ORSD, Competency Questions Definition, Class Definition, Property Definition, and Individuals Definition.

In the following, we briefly describe what each template is. Moreover, inside each available template, we provide additional links to resources to use when needed. These supplementary resources can help to clarify concepts and to observe examples.

6.1 Ontology Requirements Specification

One can use the template *Template_OntologyRequirementsSpecification* to define the purpose and scope of the ontology. It also helps to determine the intended uses and end-users of the ontology. It is complemented with the *Template_CompetencyQuestions_Definition* to determine the ontology requirements.

6.2 Competency Questions Definition

The *Template_CompetencyQuestions_Definition* helps to define the questions that the ontology should be able to answer once the modelling process is complete. They are also used to identify the important terms in the ontology. Those important terms then can be defined by using the *Template_Class_Definition*, *Template_Property_Definition* , and *Template_Individuals_Definition* templates.

Additionally, Competency Questions are commonly used to evaluate the newly created ontology, as one has to ensure that the ontology contains enough information to answer these questions.

6.3 Class Definition

The *Template_Class_Definition* helps to identify and define the classes in the ontologies. Classes are collections of objects, e.g. *Machine, Product* etc.

6.4 Property Definition

The *Template_Property_Definition* helps to identify and define the properties in the ontologies. Properties, also called slots, are the attributes describing one or more classes, e.g. *hasVolume, hasResponsible, hasPart* etc.

6.5 Individuals Definition

The *Template_Individuals_Definition* helps to identify and define the individuals in the ontologies. Individuals are instances of classes having an associated value, e.g. *Resistor_01* etc.

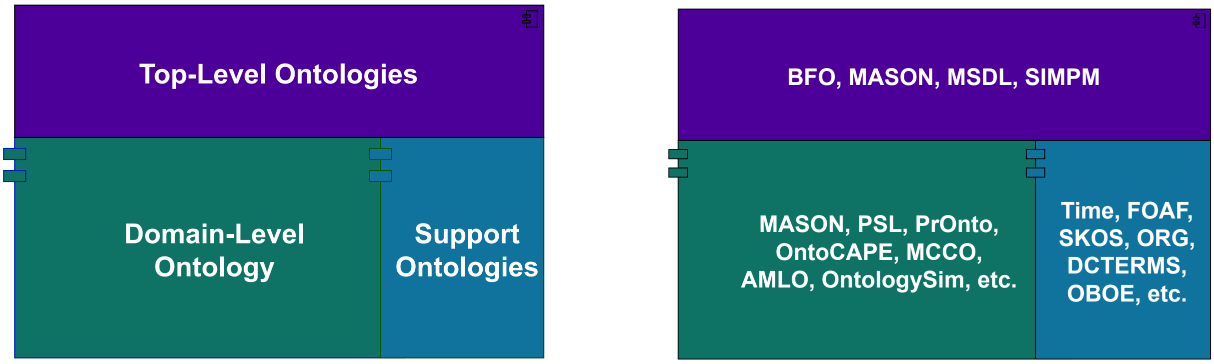
7 How to build ontologies for the IoP?

To start building the ontology by using the methodology presented in Chapter 3 and the tools indicated in Chapter 5, you need to consider the best practices we mentioned in Chapter 4.

Regarding the best practice in point 2, here we present a suggestion to build your new ontology. There are two possibilities:

- **New ontology uses support ontologies:** we recommended this style when modelling domain-specific ontologies, including non-technical concepts from the manufacturing and production domain, such as organizational and administrative knowledge.
- **New ontology does not need to use support ontologies:** we recommend this style when modelling only domain-specific knowledge.

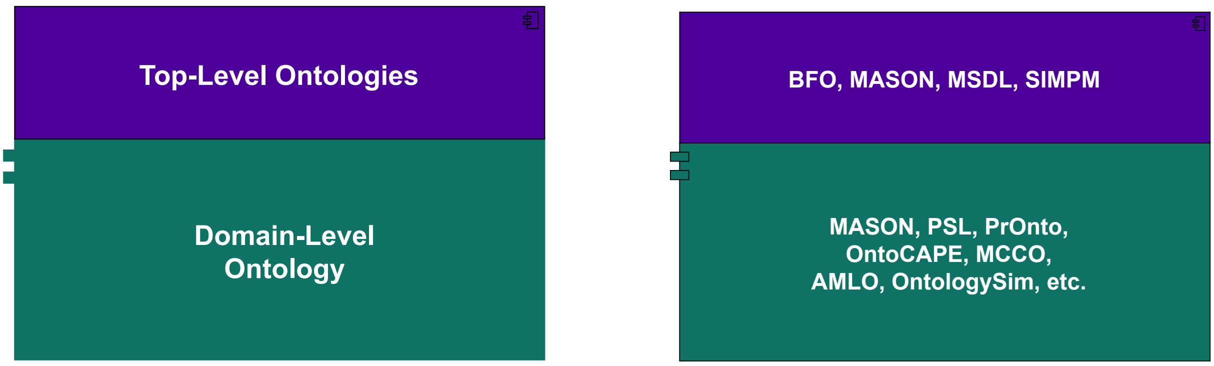
In Fig. 17 we show what the new ontology would look like by reusing existing ontologies. It is a composition of the upper (top-level) ontology, domain-specific ontologies, and support ontologies.



(a) Conceptual Ontology Architecture (b) Suggested Ontology Architecture

Figure 17. Architecture for the Ontology Development in IoP by using General Ontologies

In Fig. 18 we show how the new ontology would look by reusing existing ontologies but without using support ontologies because the model only covers domain-specific concepts. Then, it is a composition of the upper (top-level) ontology and domain-specific ontology or ontologies.



(a) Conceptual Ontology Architecture (b) Suggested Ontology Architecture

Figure 18. Architecture for the Ontology Development in IoP without using General Ontologies

7.1 Upper (Top-Level) Ontologies

As we explained in Section **2.5**, Upper or Top-Level Ontologies are generic and it is helpful to use them to start building a new ontology on top of them, as indicated in points **5** and **6**. Therefore, in Table **7** we present a list of some important ontologies in this category. Each resource name is a link to access them.

Resource	Features
OntoCape	Process engineering
BFO	Information retrieval, analysis and integration in various domains
GFO	Categories such as objects, processes, time and space, relations,roles, etc.
MASON	Automatic cost estimation and manufacturing simulation
SIMPM	Model constraints of manufacturing process planning: variety, time, and aggregation
MSDL	Represents conventional manufacturing processes
YAMATO	Quality and quantity, Objects, Processes and Events, etc.
COSMO	Broad semantic interoperability
gist	Maximum coverage of typical business ontology concepts

Table 7. Upper and Middle Ontologies

7.2 Support Ontologies: Ontologies Supporting integration with IoP

Here we present some valuable ontologies and standard vocabularies for modelling cross-domain concepts such as organization structure, time and measurement concepts.

In Table **8**, the column *"Resource"* provides the name and an URL to access the documentation of the ontology or vocabulary. In column *"Features"* we briefly describe the scope of the resource.

Resource	Features
ORG	Describe organizational structures
Time	Represent temporal concepts
vCard	Promote the use of vCard for the description of people and organisations
SKOS	Promote sharing and linking knowledge organization systems
FOAF	Describe set of concepts and properties linking people and information
QUDT	Represent various quantity and unit standards
DCMI-Terms	Specifies metadata terms
RDFDataCube	Publication of multi-dimensional data on the web
QB4ST	Extension of RDF Data Cube ontology to describe Spatio-temporal data
OM	Helps to model concepts and relations important to scientific research
Metadata4Ing	Helps to describe the generation of research data within a scientific activity
DCAT	It allows the description of data catalogs published on the Web
PROV-O	It allows integrating domains provenance in the information
VoID	It helps in describing metadata about RDF datasets
DQV	It allows to express concepts related to data quality

Table 8. Support Ontologies

7.3 Domain Ontologies: Ontologies in IoP

There exist various domain-specific ontologies in the manufacturing and production domain. Some of these and other ontologies can be found by following the links we present in Section **7.4**.

In the Appendix, in Section **A.2** we present the list of domain ontologies in the manufacturing and production domain by including the respective references in case you want to explore them more in-depth.

7.4 Catalogue

Here we present a catalogue of resources to localize some of the ontologies mentioned in the previous sections. Each resource name is a link to access them. We also added comments on how to use these resources.

Resource	Comments
NFDI4Ing Terminology Service	It has search option by terms and list of ontologies
Linked Open Data	One can filter by tag, for example, by tag "industry"
Ontohub	It has different filters such as ontology type
VoCol Service	It has a search option by term and an search by ontology
Bartoc	It has a search option and also a filtering system
Ontocommons	A list of ontologies in the manufacturing domain and other industries
Ontologies in GitHub	A search by keyword "ontology"
ODP ontologies	A catalogue of exemplary ontologies

Table 9. Ontologies and Vocabularies Catalogues

References

[1] Sadeer Beden, Qiushi Cao, and Arnold Beckmann. "Semantic Asset Administration Shells in Industry 4.0: A Survey". In: *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*. 2021, pp. 31–38. DOI: 10.1109/ICPS49255.2021.9468266.

[2] T. et al. Berners-Lee. *RFC3986 - Uniform Resource Identifier (URI): Generic Syntax*. https://datatracker.ietf.org/doc/html/rfc3986, Last accessed on 2022-06-06. 2005.

[3] Federal Ministry for Economic Affairs and Climate Action. *What is Industrie 4.0?* https://www.plattform-i40.de/IP/Navigation/EN/Industrie40/WhatIsIndustrie40/what-is-industrie40.html, Last accessed on 2021-12-12. 2022.

[4] Tsvetanka Georgieva-Trifonova and Miroslav Galabov. "Semantic Web Technologies for Big Data Modeling from Analytics Perspective: A Systematic Literature Review". In: *Balt. J. Mod. Comput.* 9 (2021).

[5] Thomas R. Gruber. "A Translation Approach to Portable Ontology Specifications". In: *Knowledge Acquisition* 5 (1993), pp. 199–220. URL: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

[6] Jason Jaskolka, Wendy MacCaull, and Ridha Khedri. "Towards an Ontology Design Architecture". In: *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE. 2015, pp. 132–135.

[7] Yannis Kalfoglou and Marco Schorlemmer. "Ontology Mapping: The State of the Art". In: *Semantic Interoperability and Integration*. Ed. by Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold. Dagstuhl Seminar Proceedings 04391. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2005. URL: http://drops.dagstuhl.de/opus/volltexte/2005/40.

[8] C. Maria Keet. "Closed World Assumption". In: *Encyclopedia of Systems Biology*. New York, NY: Springer New York, 2013, pp. 415–415. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_731. URL: https://doi.org/10.1007/978-1-4419-9863-7_731.

[9] C. Maria Keet. "Open World Assumption". In: *Encyclopedia of Systems Biology*. New York, NY: Springer New York, 2013, pp. 1567–1567. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_734. URL: https://doi.org/10.1007/978-1-4419-9863-7_734.

[10] Evgeny Kharlamov, Francisco Martin-Recuerda, Brandon Perry, David Cameron, Roar Fjellheim, and Arild Waaler. "Towards Semantically Enhanced Digital Twins". In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 4189–4193. DOI: 10.1109/BigData.2018.8622503.

[11] Stefanie Neuenschwander, Patricia Romao, Jürgen Holm, and Murat Sariyar. "Developing an Ontology for Documenting Adverse Events While Avoiding Pitfalls". In: *Studies in health technology and informatics* 289 (2022), pp. 166–169. DOI: 10.3233/SHTI210885.

[12] Natalya F. Noy and Deborah L. Mcguinness. *Ontology Development 101: A Guide to Creating Your First Ontology*. Tech. rep. Stanford, CA: Stanford University, 2001. URL: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf.

[13] Li Peng and Yuan Man. "A Novel Data Exchange Model Based on P2P and Ontology". In: *Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*. 2013, pp. 1693–1696. DOI: 10.1109/MEC.2013.6885329.

[14] Asunción Pérez Gómez, Mcs De Figueroa Baonza, and Boris Villazón. "Neon Methodology for Building Ontology Networks: Ontology Specification". In: *Methodology* (2008), pp. 1–18.

[15] María Poveda-Villalón, Alba Fernández-Izquierdo, Mariano Fernández-López, and Raúl García-Castro. "LOT: An industrial oriented ontology engineering framework". In: *Engineering Applications of Artificial Intelligence* 111 (2022), p. 104755.

[16] Valentina Presutti, Aldo Gangemi, Stefano David, G Aguado de Cea, MC Surez-Figueroa, Elena Montiel-Ponsoda, and M Poveda. "NeOn Deliverable D2. 5.1. A Library of Ontology Design Patterns: Reusable Solutions for Collaborative Design of Networked Ontologies". In: *NeOn Project. http://www. neon-project. org* (2008).

[17] Foivos Psarommatis, Paul Arthur Dreyfus, and Dimitris Kiritsis. "Chapter 9 - The role of big data analytics in the context of modeling design and operation of manufacturing systems". In: *Design and Operation of Production Networks for Mass Personalization in the Era of Cloud Technology*. Ed. by Dimitris Mourtzis. Elsevier, 2022, pp. 243–275. ISBN: 978-0-12-823657-4. DOI: https://doi.org/10.1016/B978-0-12-823657-4.00012-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128236574000129.

[18] Borja Ramis Ferrer, Wael M. Mohammed, Mussawar Ahmad, Sergii Iarovyi, Jiayi Zhang, Robert Harrison, and Jose Luis Martinez Lastra. "Comparing Ontologies and Databases: A Critical Review of Lifecycle Engineering Models in Manufacturing". In: *Knowledge and Information Systems* (2021). DOI: https://doi.org/10.1007/s10115-021-01558-4.

[19] Víctor Rodríguez-Doncel, Adila Krisnadhi, Pascal Hitzler, Michelle Cheatham, Nazifa Karima, and Reihaneh Amini. "Pattern-based Linked Data Publication: The Linked Chess Dataset Case". In: *Proceedings of the 6th International Workshop on Consuming Linked Data co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, Pennsylvania, US, October 12th, 2015*. Ed. by Olaf Hartig, Juan Sequeda, and Aidan Hogan. Vol. 1426. CEUR-WS.org. CEUR-WS.org, 2015. URL: http://dase.cs.wright.edu/publications/pattern-based-linked-data-publication-linked-chess-dataset-case.

[20] Ron Rudnicki, Barry Smith, Tanya Malyuta, and William Mandrick. *Best Practices of Ontology Development*. Tech. rep. CUBRC, Inc., Oct. 2016.

[21] Stuart Russell and Peter Norvig. *Artificial intelligence (Third edition, Global edition)*. 3rd ed. Prentice Hall, 2016.

[22] RWTH Aachen University. *Internet of Production*. https://www.rwth-aachen.de/go/id/bktz/lidx/1, Last accessed on 2022-01-14. 2022.

[23] Juan Sequeda. *Introduction to: Open World Assumption vs Closed World Assumption*. https://www.dataversity.net/introduction-to-open-world-assumption-vs-closed-world-assumption, Last accessed on 2022-06-04. 2012.

[24] Cogan Shimizu, Quinn Hirt, and Pascal Hitzler. *MODL: A Modular Ontology Design Library*. 2019. arXiv: 1904.05405 [cs.AI].

[25] Michael Uschold. "Ontology and Database Schema: What?s the Difference?" In: *Applied Ontology* 10.3-4 (2015), pp. 243–258. DOI: 10.3233/ao-150158.

[26] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. "Ontology-based Integration of Information - A Survey of Existing Approaches". In: *Ois@ ijcai*. 2001, pp. 108–117.

[27] Xinjie Zhou, Guyue Gao, Xinguo Ming, Liya Wang, Dao Yin, and Xiaohong Ma. "Task-oriented Knowledge Representation and Ontology Modeling for Complex Product Design". In: *2021 International Conference on Service Science (ICSS)*. 2021, pp. 30–37. DOI: 10.1109/ICSS53362.2021.00013.

A Appendix

A.1 LOT Methodology Figures

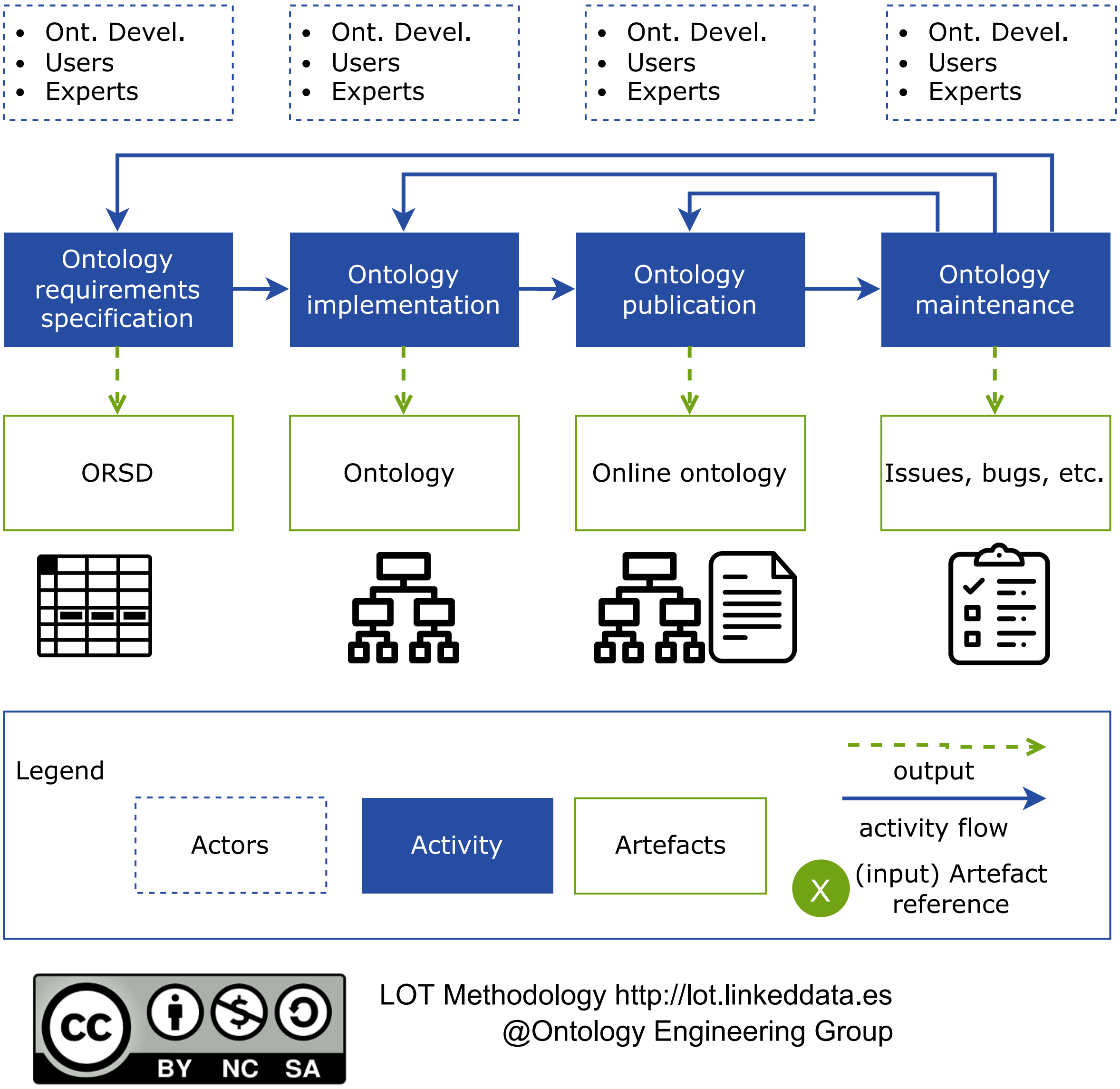


Figure 19. LOT Methodology Overview
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

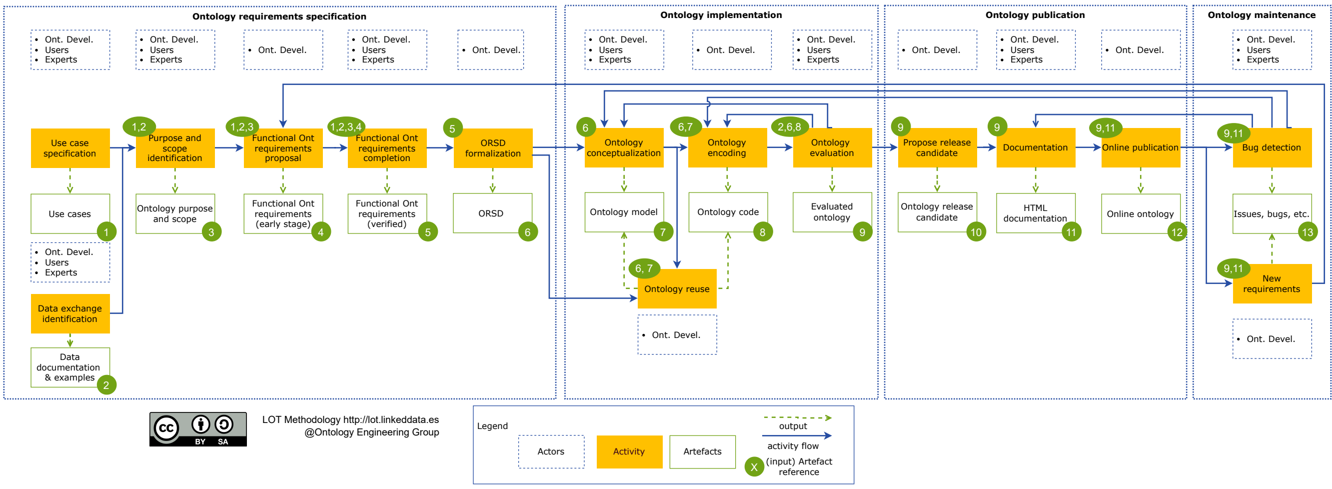


Figure 20. LOT Methodology Detailed View
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

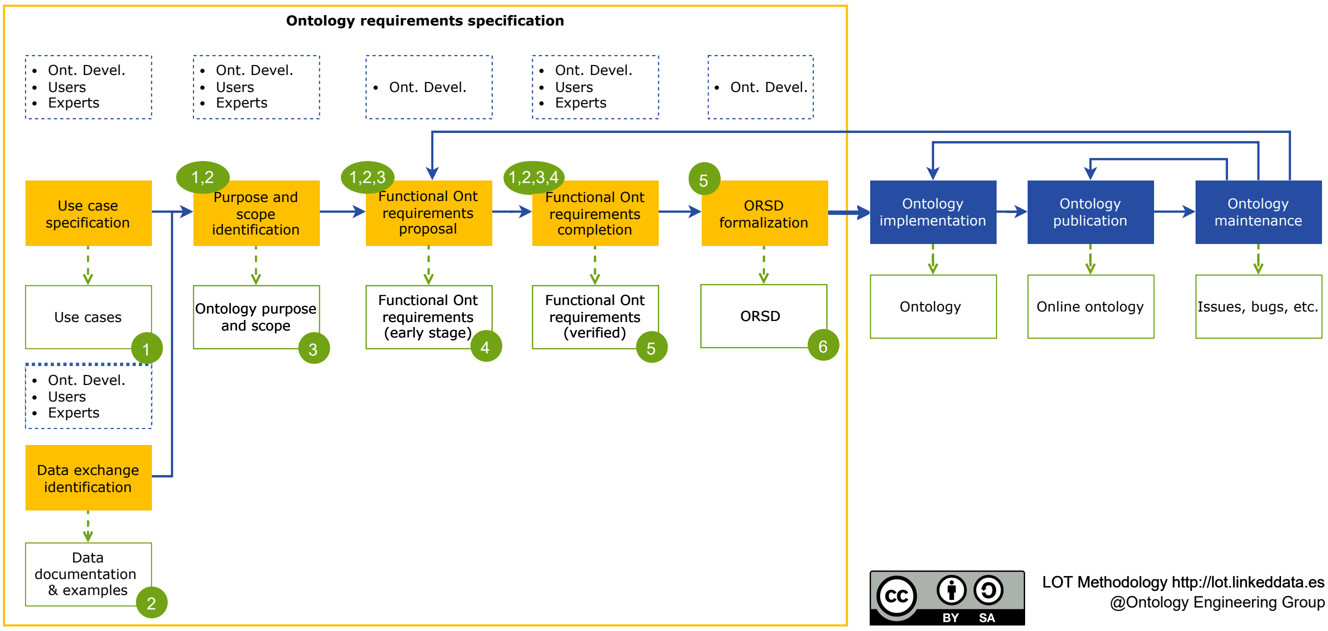


Figure 21. LOT Methodology - Stage: Ontology Requirements Specification
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

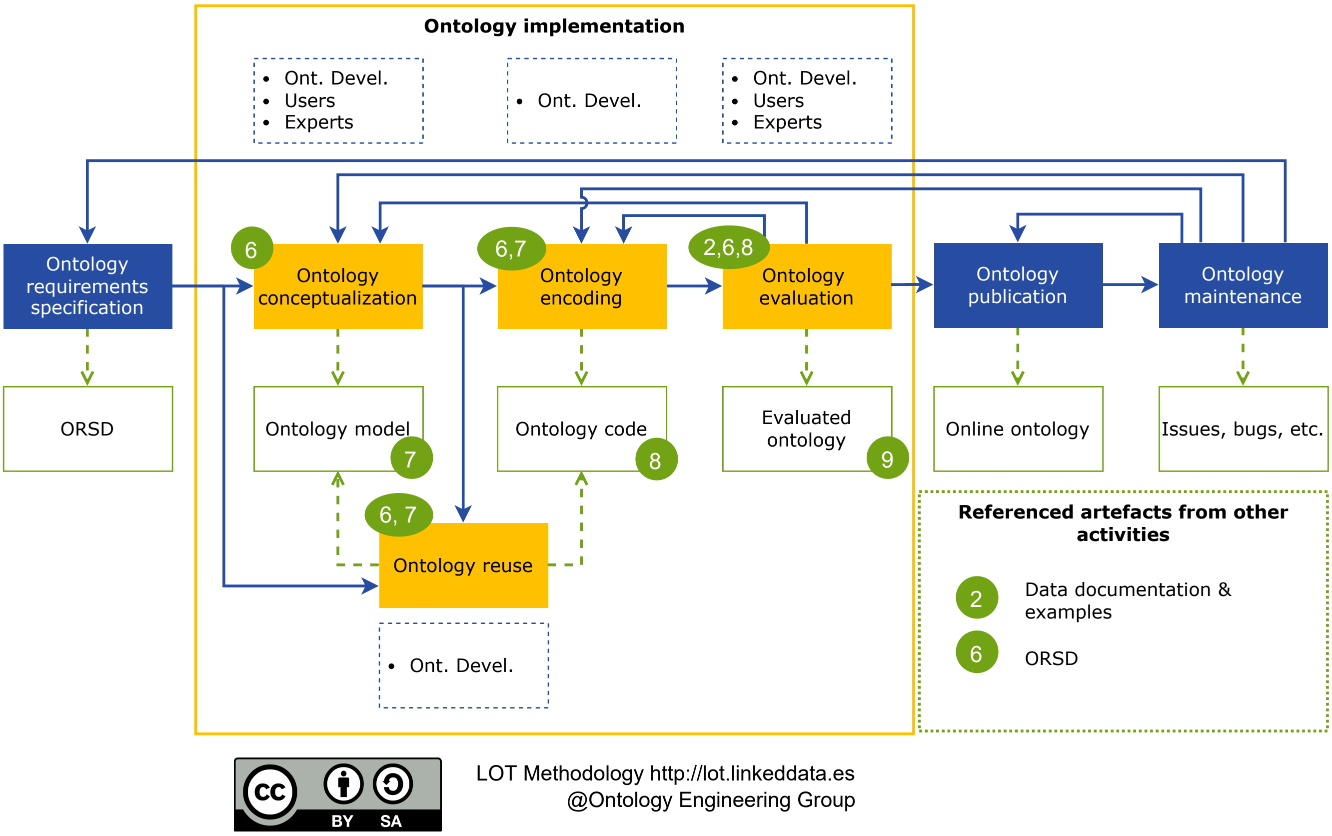


Figure 22. LOT Methodology - Stage: Ontology Implementation
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

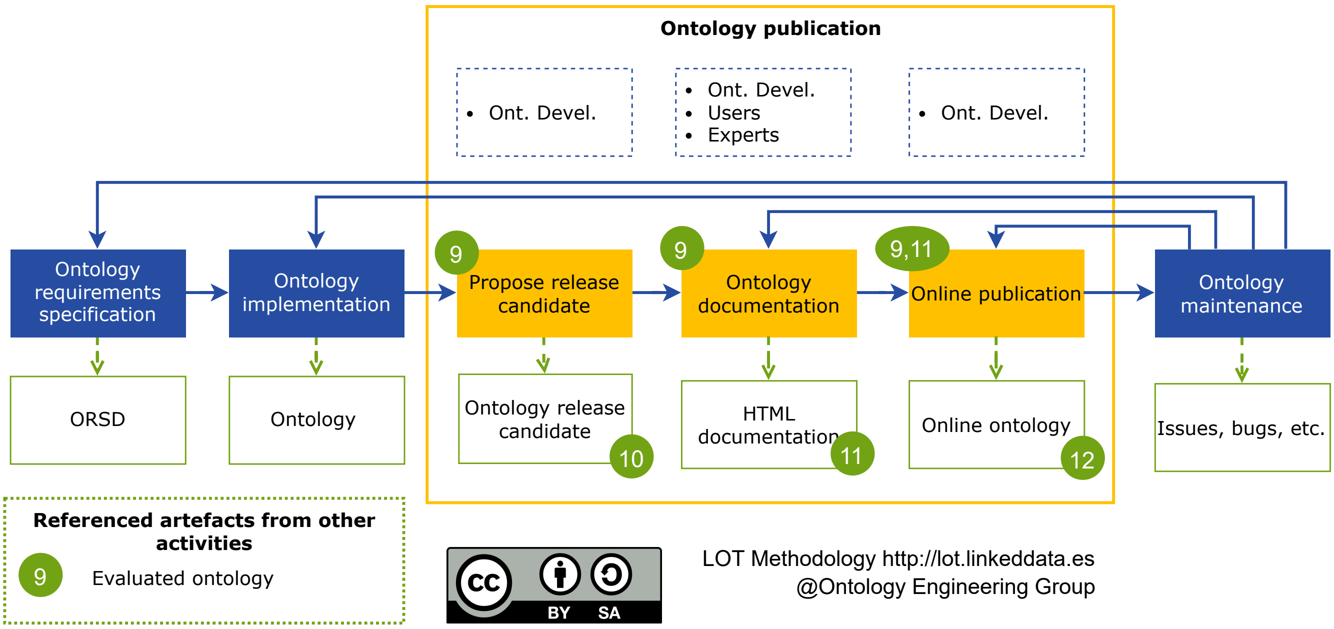


Figure 23. LOT Methodology - Stage: Ontology Publication
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

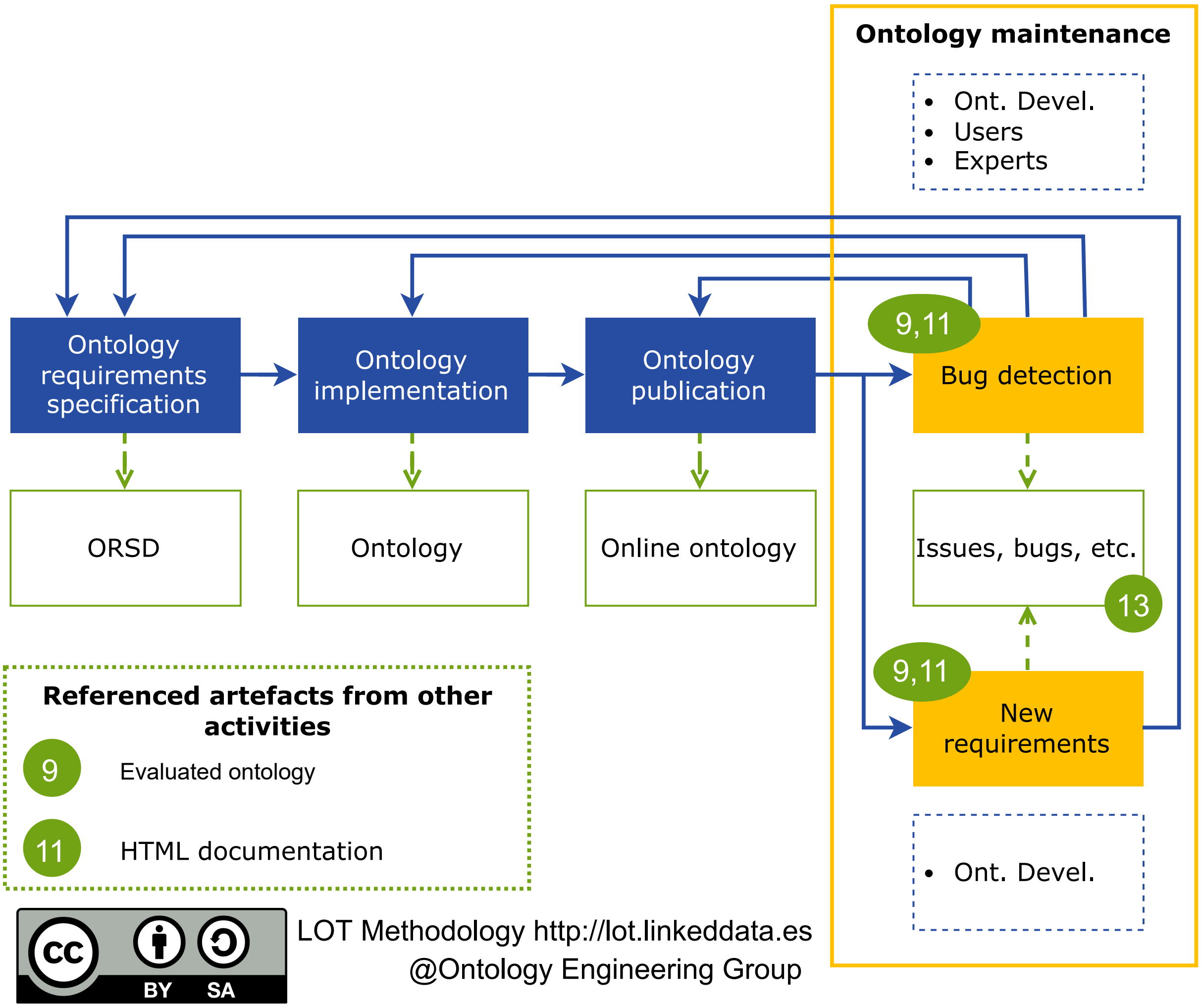


Figure 24. LOT Methodology - Stage: Ontology Maintenance
Ontological Engineering Group (OEG). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

A.2 Detailed Information about Domain Ontologies

The following content corresponds to domain-ontologies in the manufacturing and production domain, with their respective references and classification. **SM** indicates that the resource is a Semantic Model, and **O** means that it is an ontology.

Legend: **MFG:** Manufacturing, **PROD:** Products, **PROC:** Processes, **RES:** Resources, **P.COMP:** Plant Components, **ACTV:** Activities, **SHED:** Scheduling, **MAINT:** Maintenance, **SENS:** Sensors, **ROBT:** robotics, **ENG:** Engineering, **BATCH:** Batch Processing, **MSMT:** Measurements, **STD:** Standards, **SIM:** Simulation

	MFG	PROD	PROC	RES	P.COMP	ACTV	SHED	MAINT	SENS	ROBT	ENG	BATCH	MSMT	STD	SIM	CLASSIF	REF
SOIL	✓	✓	✓	✓									✓			SM	[1]
SemAnz4.0	✓	✓	✓													SM	[2]
iFAB	✓															SM	[2]
ADACOR	✓		✓		✓		✓									SM	[2]
MASON	✓	✓	✓	✓												O	[3, 4]
MaSDeM	✓	✓	✓	✓							✓					O	[2]
SIMPM	✓		✓													O	[5]
PSL	✓		✓			✓	✓								✓	O	[3]
OntoCAPE			✓								✓					O, SM	[2]
BaPrOn			✓				✓					✓				O	[6]
FABMAS	✓	✓		✓		✓	✓									O	[7]
PrOnto	✓	✓	✓		✓											O	[2, 8]
ARUM	✓	✓	✓	✓			✓									O	[9]
RGOM	✓	✓	✓	✓			✓	✓						✓		SM	[8]
ONTO-PDM	✓	✓	✓	✓			✓							✓		O	[10]
MPMO	✓		✓	✓						✓						O	[11]
MCCO	✓	✓														O	[2]
SCRO	✓		✓	✓	✓	✓										O	[12]
MSDL	✓	✓	✓	✓												O	[2]
AMU	✓	✓	✓	✓	✓											O	[13]
MTM	✓			✓												O	[14]
MPO	✓		✓	✓							✓					O	[15]
IMAMO	✓			✓	✓			✓	✓							O	[13]
IEO	✓	✓	✓	✓	✓							✓				O	[2]
COMPOSITION	✓	✓	✓	✓												O	[16]
MaRCO	✓			✓												O	[17]
ExtruOnt	✓		✓	✓					✓							O	[17]
OntoProg	✓			✓				✓								O	[18]
Onto-ICMS	✓		✓	✓	✓	✓		✓								O	[19]
AMLO	✓	✓	✓	✓	✓				✓		✓			✓		O	[20]
WSSN								✓								O	[21]
M3									✓	✓			✓	✓		O	[22]
CSIRO-SO			✓						✓				✓			O	[23]
OPW	✓	✓	✓	✓		✓										O	[24]
CDM-Core	✓		✓				✓			✓			✓	✓		O	[25, 13]
ROMAIN			✓	✓				✓								O	[26, 13]
STO	✓													✓		O	[27]
SSN				✓					✓							O	[28]
newSSN				✓					✓							O	[28]
AWARE				✓					✓	✓						O	[28]

Continued on next page . . .

…Ontologies in the Production and Processes Domains (Continued)

	MFG	PROD	PROC	RES	P.COMP	ACTV	SHED	MAINT	SENS	ROBT	ENG	BATCH	MSMT	STD	SIM	CLASSIF	REF
OntologySim	✓	✓	✓	✓		✓									✓	O	[29]
CMSD	✓	✓	✓	✓											✓	O	[30]

References

[1] Amon Göppert, Lea Grahn, Jonas Rachner, Dennis Grunert, Simon Hort, and Robert H Schmitt. 2021. Pipeline for ontology-based modeling and automated deployment of digital twins for planning and control of manufacturing systems. *Journal of Intelligent Manufacturing*, 1–20.

[2] Felix Ocker, Birgit Vogel-Heuser, and Christiaan J. J. Paredis. 2019. Applying Semantic Web Technologies to Provide Feasibility Feedback in Early Design Phases. *Journal of Computing and Information Science in Engineering*, 19, 4, (July 2019). 041016. ISSN: 1530-9827. DOI: 10.1115/1.4043795. eprint: https://asmedigitalcollection.asme.org/computingengineering/article-pdf/19/4/041016/6415224/jcise_19_4_041016.pdf. https://doi.org/10.1115/1.4043795.

[3] Sadeer Beden, Qiushi Cao, and Arnold Beckmann. 2021. Semantic Asset Administration Shells in Industry 4.0: A Survey. In *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, 31–38. DOI: 10.1109/ICPS49255.2021.9468266.

[4] S. Lemaignan, A. Siadat, J.-Y. Dantan, and A. Semenenko. 2006. MASON: A Proposal For An Ontology Of Manufacturing Domain. In *IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)*, 195–200. DOI: 10.1109/DIS.2006.48.

[5] Dušan Šormaz and Arkopaul Sarkar. 2019. Simpm – upper-level ontology for manufacturing process plan network generation. *Robotics and Computer-Integrated Manufacturing*, 55, 183–198. Extended Papers Selected from FAIM2016. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2018.04.002. https://www.sciencedirect.com/science/article/pii/S0736584517302119.

[6] E. Muñoz, G.M. Kopanos, A. Espuña, and L. Puigjaner. 2009. Towards an Ontological Infrastructure for Chemical Batch Process Management. In *19th European Symposium on Computer Aided Process Engineering*. Computer Aided Chemical Engineering. Volume 26. Jacek Jeżowski and Jan Thullie, editors. Elsevier, 883–888. DOI: https://doi.org/10.1016/S1570-7946(09)70147-6. https://www.sciencedirect.com/science/article/pii/S1570794609701476.

[7] Lars Mönch and Marcel Stehli. 2003. An ontology for production control of semiconductor manufacturing processes. In *German Conference on Multiagent System Technologies*. Springer, 156–167.

[8] Muhammad Yahya, John G. Breslin, and Muhammad Intizar Ali. 2021. Semantic Web and Knowledge Graphs for Industry 4.0. *Applied Sciences*, 11, 11. ISSN: 2076-3417. DOI: 10.3390/app11115110. https://www.mdpi.com/2076-3417/11/11/5110.

[9] Ondřej Harcuba and Pavel Vrba. 2015. Ontologies for flexible production systems. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 1–8.

[10] Hervé Panetto, Michele Dassisti, and Angela Tursi. 2012. Onto-pdm: product-driven ontology for product data management interoperability within manufacturing process environment. *Advanced Engineering Informatics*, 26, 2, 334–348.

[11] Qiushi Cao, Ahmed Samet, Cecilia Zanni-Merk, François de Bertrand de Beuvron, and Christoph Reich. 2020. Combining Chronicle Mining and Semantics for Predictive Maintenance in Manufacturing Processes. *Semantic Web*, 11, 6, 927–948.

[12] Sadeer Beden, Qiushi Cao, and Arnold Beckmann. 2021. SCRO: A Domain Ontology for Describing Steel Cold Rolling Processes towards Industry 4.0. *Information*, 12, 8. ISSN: 2078-2489. DOI: 10.3390/info12080304. https://www.mdpi.com/2078-2489/12/8/304.

[13] Marco Kainzner, Christoph Klösch, Dominik Filipiak, Tek Raj Chhetri, and Anna Fensel. 2021. Poster: Towards Reusable Ontology Alignment for Manufacturing Maintenance. In *CEUR workshop proceedings series (Vol-2941)*. SEMANTiCS 2021 EU, Amsterdam, the Netherlands. http://ceur-ws.org/Vol-2941/paper9.pdf.

[14] T. Kjellberg, A. von Euler-Chelpin, M. Hedlind, M. Lundgren, G. Sivard, and D. Chen. 2009. The Machine Tool Model — A Core Part of the Digital Factory. *CIRP Annals*, 58, 1, 425–428. ISSN: 0007-8506. DOI: https://doi.org/10.1016/j.cirp.2009.03.035. https://www.sciencedirect.com/science/article/pii/S0007850609001073.

[15] Luis Ramos, Richard Gil, Dimitra Anastasiou, and Maria J. Martin-Bautista. 2014. Towards a Machine of a Process (MOP) Ontology to Facilitate e-Commerce of Industrial Machinery. *Comput. Ind.*, 65, 1, 108–115. ISSN: 0166-3615. DOI: 10.1016/j.compind.2013.07.012. https://doi.org/10.1016/j.compind.2013.07.012.

[16] COMPOSITION Consortium. 2021. D6.8 Collaborative Manufacturing Services Ontology and Language II. https://www.composition-project.eu/wp-content/uploads/2019/03/D6.8_Collaborative_Manufacturing_Services_Ontology_and_Language_II.pdf, Last accessed on 2022-02-15. (2021).

[17] Víctor Julio Ramírez-Durán, Idoia Berges, and Arantza Illarramendi. 2020. ExtruOnt: An Ontology for Describing a Type of Manufacturing Machine for Industry 4.0 Systems. *Semantic Web*, 11, 887–909.

[18] Qiushi Cao, Cecilia Zanni-Merk, Ahmed Samet, Christoph Reich, de Bertrand de Beuvron. François, Arnold Beckmann, and Giannetti. Cinzia. 2022. KSPMI: A Knowledge-based System for Predictive Maintenance in Industry 4.0. *Robotics and Computer-Integrated Manufacturing*, 74, 102281. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2021.102281. https://www.sciencedirect.com/science/article/pii/S0736584521001617.

[19] Qiushi Cao, Franco Giustozzi, Cecilia Zanni-Merk, François de Bertrand de Beuvron, and Christoph Reich. 2019. Smart condition monitoring for industry 4.0 manufacturing processes: an ontology-based approach. *Cybernetics and Systems*, 50, 2, 82–96. DOI: 10.1080/01969722.2019.1565118. eprint: https://doi.org/10.1080/01969722.2019.1565118. https://doi.org/10.1080/01969722.2019.1565118.

[20] Olga Kovalenko, Irlán Grangel-González, Marta Sabou, Arndt Lüder, Stefan Biffl, Sören Auer, and Maria-Esther Vidal. 2018. Automationml ontology: modeling cyber-physical systems for industry 4.0. *IOS Press Journal*, 1.

[21] Rimel Bendadouche, Catherine Roussey, Gil De Sousa, Jean-Pierre Chanet, and Kun Mean Hou. 2012. Extension of the semantic sensor network ontology for wireless sensor networks: the stimulus-wsnnode-communication pattern. In *5th International Workshop on Semantic Sensor Networks in conjunction with the 11th International Semantic Web Conference (ISWC)*, 16–p.

[22] Amelie Gyrard, Soumya Kanti Datta, Christian Bonnet, and Karima Boudaoud. 2015. Integrating machine-to-machine measurement framework into onem2m architecture. In *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 364–367.

[23] Holger Neuhaus and Michael Compton. 2009. The semantic sensor network ontology. In *AGILE workshop on challenges in geospatial data harmonisation, Hannover, Germany*, 1–33.

[24] Kudirat Ayinla, Edlira Vakaj, Franco Cheung, and Abdel-Rahman H. Tawil. 2021. A Semantic Offsite Construction Digital Twin-Offsite Manufacturing Production Workflow (OPW) Ontology. In.

[25] Luca Mazzola, Patrick Kapahnke, Marko Vujic, and Matthias Klusch. 2016. CDM-Core: A Manufacturing Domain Ontology in OWL2 for Production and Maintenance. In *KEOD*.

[26] Mohamed Hedi Karray, Farhad Ameri, Melinda Hodkiewicz, and Thierry Louge. 2019. ROMAIN: Towards a BFO Compliant Reference Ontology for Industrial Maintenance. *Applied Ontology*, 14, 2, 155–177.

[27] Irlán Grangel-González, Paul Baptista, Lavdim Halilaj, Steffen Lohmann, Maria-Esther Vidal, Christian Mader, and Sören Auer. 2017. The industry 4.0 standards landscape from a semantic integration perspective. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–8. DOI: 10.1109/ETFA.2017.8247584.

[28] Boulos El Asmar, Syrine Chelly, and Michael Färber. 2020. AWARE: An Ontology for Situational Awareness of Autonomous Vehicles in Manufacturing. (2020).

[29] M. C. May, L. Kiefer, A. Kuhnle, and G. Lanza. 2022. Ontology-based production simulation with ontologysim. *Applied Sciences (Switzerland)*, 12, 3, Art.–Nr.: 1608. ISSN: 2076-3417. DOI: 10.3390/app12031608.

[30] Christian Block, Dominik Lins, and Bernd Kuhlenkötter. 2018. Approach for a simulation-based and event-driven production planning and control in decentralized manufacturing execution systems. *Procedia CIRP*, 72, 1351–1356. 51st CIRP Conference on Manufacturing Systems. ISSN: 2212-8271. DOI: https://doi.org/10.1016/j.procir.2018.03.204. https://www.sciencedirect.com/science/article/pii/S2212827118303639.