

# COGITO

CONSTRUCTION PHASE  
DIGITAL TWIN MODEL

[cogito-project.eu](http://cogito-project.eu)

COGITO Data  
Model and  
Ontology  
Definition and  
Interoperability  
Design



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958310.

## D3.2 – COGITO Data Model and Ontology Definition and Interoperability Design

Dissemination Level: Public

Deliverable Type: Report

Lead Partner: UPM

Contributing Partners: Hypertech, UCL, AU, UEDIN, CERTH, BOC, QUE, NT

Due date: 30-09-2021

Actual submission date: 01-10-2021

### Authors

Name	Beneficiary	Email
<b>Socorro Bernardos</b>	UPM	sbernardos@fi.upm.es
<b>Alba Fernández-Izquierdo</b>	UPM	aizquierdo@fi.upm.es
<b>Raúl García-Castro</b>	UPM	rgarcia@fi.upm.es
<b>Giorgos Giannakis</b>	Hypertech	g.giannakis@hypertech.gr
<b>Georgios Lilis</b>	UCL	g.lilis@ucl.ac.uk
<b>Jochen Teizer</b>	AU	teizer@eng.au.dk
<b>Frédéric Bosché</b>	UEDIN	f.bosche@ed.ac.uk
<b>Vasilios Karkanis</b>	CERTH	vkarkanis@iti.gr
<b>Damiano Falcioni</b>	BOC	damiano.falcioni@boc-eu.com
<b>Panagiotis Moraitis</b>	QUE	p.moraitis@que-tech.com
<b>Ján Varga</b>	NT	varga@novitechgroup.sk

### Reviewers

Name	Beneficiary	Email
<b>Tobias Hanel</b>	FER	thanel@ferrovial.com
<b>Georgios Lilis</b>	UCL	g.lilis@ucl.ac.uk

### Version History

Version	Editors	Date	Comment
<b>0.1</b>	Alba Fernández-Izquierdo	27.07.2021	Methodology and infrastructure sections
<b>0.2</b>	Socorro Bernardos	02.09.2021	Requirements and ontology network sections
<b>0.3</b>	Raúl García-Castro	03.09.2021	General review
<b>0.4</b>	Raúl García-Castro	07.09.2021	Coverage section; introduction & conclusions
<b>0.5</b>	Socorro Bernardos	08.09.2021	Executive summary, some revisions
<b>0.6</b>	Raúl García-Castro	09.09.2021	General review & revision. Version sent to reviewers
<b>0.7</b>	Socorro Bernardos	16.09.2021	Revisions according to FER review
<b>0.8</b>	Socorro Bernardos	17.09.2021	Revisions based on Hypertech, UCL, UEDIN reviews
<b>1.0</b>	Raúl García-Castro	09.09.2021	First version

## Disclaimer

©COGITO Consortium Partners. All rights reserved. COGITO is a HORIZON2020 Project supported by the European Commission under Grant Agreement No. 958310. The document is proprietary of the COGITO consortium members. No copying or distributing, in any form or by any means, is allowed without the prior written agreement of the owner of the property rights. The information in this document is subject to change without notice. Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies. The information and views presented in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Communities. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use, which may be made, of the information contained therein.

## Executive Summary

The COGITO Digital Twin platform aims to enable interoperability with existing and emerging standards and data models covering different domains. Semantic interoperability in the COGITO platform relies on ontologies that will be exploited throughout the COGITO infrastructure and will be used to semantically represent the information exchanged.

This document details the methodology and technological infrastructure used to develop the COGITO ontology network, as well as the first version of the ontologies that make up such an ontology network.

The followed methodology is called LOT [1] and includes four activities: 1) ontology requirements specification, 2) ontology implementation, 3) ontology publication, and 3) ontology maintenance.

The current version of the COGITO ontology network (see Figure 7) consists of three modules corresponding to the construction itself, the construction process, and the construction resources (see Table 9). These modules have already been conceptualised, implemented, and published and will be modified and complemented with new modules as the project progresses.

Some of the ontologies reviewed in previous steps of the project (in particular, in Task 3.1) have been reused according to the results of a requirement coverage analysis, which allowed us to identify that some of the terms that appear in the requirements also appear in the ontologies to a greater or lower extent. Likewise, it has allowed us to identify potential extensions currently not covered by the analysed ontologies such as: description of the construction as a whole, quality of construction elements, of relevant resources (workers, equipment, and materials) and their relationship with the construction process, and of metrics on the construction process such as the quality or the cost of tasks.

The next steps will be devoted to continuing to collect more ontological requirements from the pilots and from the architecture to iteratively evolve the COGITO ontology network. The COGITO ontology portal will serve as a living repository for the different ontologies and will contain the latest version of the developed ontologies and of all the related artifacts.

## Table of Contents

Executive Summary .....	3
Table of Contents .....	4
List of Figures .....	6
List of Tables.....	7
List of Acronyms and Abbreviations.....	8
1 Introduction .....	9
1.1 Scope and Objectives of the Deliverable .....	9
1.2 Relation to other Tasks and Deliverables.....	9
1.3 Structure of the Deliverable .....	9
2 Ontology Development Methodology .....	10
2.1 Ontological Requirements Specification.....	10
2.1.1 Identification of Purpose and Scope .....	11
2.1.2 Data Exchange Identification and Use Case Specification.....	11
2.1.3 Ontological Requirement Proposal .....	11
2.1.4 Completion and Validation of Ontology Requirements .....	12
2.1.5 Prioritization of Ontological Requirements.....	12
2.2 Ontology Implementation .....	12
2.2.1 Ontology Conceptualization.....	12
2.2.2 Ontology Encoding .....	13
2.2.3 Ontology Evaluation .....	13
2.3 Ontology Publication .....	13
2.3.1 Propose Release Candidate.....	14
2.3.2 Ontology Documentation.....	14
2.3.3 Online Publication .....	14
2.4 Ontology Maintenance .....	14
2.4.1 Bug Detection.....	15
2.4.2 New Requirements Proposal.....	15
3 Ontology Requirements Specification.....	16
4 Coverage Analysis .....	18
4.1 Methodology .....	18
4.2 Results.....	19
4.2.1 Construction .....	20
4.2.2 Resource.....	21
4.2.3 Construction Process.....	23
5 Overview of the COGITO Ontology Network .....	26

5.1	Construction Module .....	28
5.2	Process Module .....	29
5.3	Resource Module .....	30
6	Ontology Development Infrastructure.....	32
6.1	Infrastructure to Support the Requirements Specification Activity .....	32
6.2	Infrastructure to Support the Ontology Implementation Activity .....	33
6.3	Infrastructure to Support the Ontology Publication Activity .....	33
6.4	Infrastructure to Support the Ontology Maintenance Activity.....	34
7	Conclusions .....	36
	References .....	37

## List of Figures

Figure 1 - Ontology development methodology followed in COGITO .....	10
Figure 2 - Workflow for ontology requirements specification .....	11
Figure 3 - Workflow for the ontology implementation activity .....	12
Figure 4 - Workflow for the ontology publication activity .....	14
Figure 5 - Workflow for the ontology maintenance activity .....	15
Figure 6 - Methodology for the coverage analysis .....	19
Figure 7 – COGITO ontology network .....	26
Figure 8 – General overview of the COGITO construction ontology .....	29
Figure 9 - General overview of the COGITO process ontology .....	30
Figure 10 – General overview of the COGITO resource ontology .....	31
Figure 11 - COGITO ontology requirements for PMS .....	32
Figure 12 - HTML requirements for the construction process .....	32
Figure 13 - Ontology section in the COGITO ontology portal .....	34
Figure 14 - Example of the GitHub issue tracker .....	35

## List of Tables

Table 1 – General information about the ontology requirements specification .....	16
Table 2 - List of requirements per domain .....	16
Table 3 – Term coverage results for the Construction domain.....	20
Table 4 – Requirement coverage results for the Construction domain .....	20
Table 5 – Term coverage results for the Resource domain .....	22
Table 6 – Requirement coverage results for the Resource domain .....	23
Table 7 – Term coverage results for the Construction Process domain .....	24
Table 8 – Requirement coverage results for the Construction Process domain .....	24
Table 9 – List of the ontologies created in the COGITO project.....	27
Table 10 – List of the ontologies reused in the COGITO project .....	27



## List of Acronyms and Abbreviations

Term	Description
<b>BBO</b>	BPMN Based Ontology
<b>BOT</b>	Building Topology Ontology
<b>BPMN</b>	Business Process Modelling Notation
<b>BPO</b>	Building Product Ontology
<b>DICO</b>	Digital Construction
<b>ETSI</b>	European Telecommunications Standards Institute
<b>IoT</b>	Internet of Things
<b>LOT</b>	Linked Open Terms
<b>ORSO</b>	Ontology Requirements Specification Document
<b>OWL</b>	Web Ontology Language
<b>PMS</b>	Process Modelling and Simulation
<b>SAREF</b>	Smart Applications REFERENCE ontology
<b>SAREF4BLDG</b>	SAREF for Building
<b>SSN</b>	Semantic Sensor Network
<b>SOSA</b>	Sensor, Observation, Sample and Actuator
<b>WODM</b>	Work Order Definition and Monitoring tool

# 1 Introduction

## 1.1 Scope and Objectives of the Deliverable

The COGITO Digital Twin platform aims to enable interoperability with existing and emerging standards and data models covering different domains. Semantic interoperability in the COGITO platform relies on ontologies that will be exploited throughout the COGITO infrastructure and will be used to semantically represent the information exchanged.

In computer science, ontologies are defined as “formal, explicit specifications of a shared conceptualization”. The COGITO ontologies will be developed using the W3C Web Ontology Language standard (OWL)<sup>1</sup> reusing existing resources and standards whenever possible. This development is based on the technical specification of the COGITO architecture and is based on a set of requirements extracted from its different components.

This deliverable describes the environment that supports the development of the COGITO ontologies. To do so, on the one hand, it describes the methodology to be followed to develop the ontologies and the ontology development infrastructure deployed to support such development. On the other hand, it presents the outcomes of the first ontology development sprints. It summarises all the requirements that were extracted from the COGITO use cases and architecture, which are used to analyse the coverage of existing ontologies to the project requirements. Finally, this document presents an overview of the current version of the COGITO ontologies that are available online in the COGITO ontology portal.<sup>2</sup>

The version of the ontology presented in this document is the version produced on the date of writing of this document. Two other versions of this deliverable will be delivered in months 18 (April 2022) and 24 (October 2022) and will present the updated versions of the ontologies, as well as a more complete coverage analysis. In any case, the COGITO ontology portal will contain the latest version of the ontologies and all related artifacts for ontology development.

## 1.2 Relation to other Tasks and Deliverables

This deliverable is based on the use cases defined in D2.1 [2] and on the COGITO architecture defined in D2.4 [3]. Furthermore, D3.1 [4] was also considered for identifying potential existing ontologies to reuse in the COGITO ontologies.

The COGITO ontology network described here will allow the sharing and interoperability among the different components of the COGITO architecture, especially in the communications with the Digital Twin platform. That is why this document and the future versions of the ontology will be so important when developing each component of the COGITO architecture, that is, the future results of work packages 4, 5, 6, and 7 and their integration in work package 8.

## 1.3 Structure of the Deliverable

- **Section 2** provides an overview of the methodology used to develop the COGITO ontologies;
- **Section 3** presents the requirements that should be satisfied by the COGITO ontologies;
- **Section 4** describes the methodology and results for analysing the coverage of existing ontologies to the project requirements;
- **Section 5** provides a description of the COGITO ontologies;
- **Section 6** presents the deployed ontology development infrastructure; and
- **Section 7** provides some conclusions and insights into future work.

<sup>1</sup> <https://www.w3.org/TR/owl2-primer/>

<sup>2</sup> <https://cogito.iot.linkeddata.es/>

## 2 Ontology Development Methodology

This section presents the ontology development methodology used for the development of the COGITO ontology network. This methodology includes four activities: 1) ontology requirements specification, 2) ontology implementation, 3) ontology publication, and 3) ontology maintenance. This development methodology is named Linked Open Terms (LOT) [1] and is based on the NeOn methodology [5]. It has been used and refined in previous ontology development processes from the European projects VICINITY [6], BIMERR [7] and DELTA [8], and adapted to the particularities of COGITO (see sections 3 and 5).

Figure 1 shows an overview of the activities that are performed and the artefacts that result from them: the ontology requirements specification document (ORSD), the ontology implementation, the ontology available online, issues, bugs, etc. The following subsections detail each activity (and product). In the figures included in this section, these artefacts are represented in green squares and are assigned a number; the different activities in the methodology enumerate in a green circle the numbers of all the input artefacts that are used in the activity.

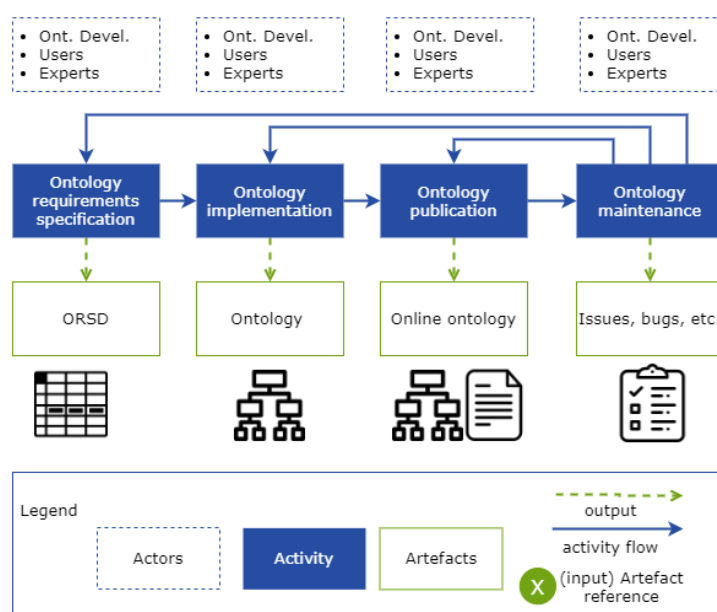


Figure 1 - Ontology development methodology followed in COGITO

### 2.1 Ontological Requirements Specification

The aim of the requirements specification activity is to identify and define the requirements the ontology to be created needs to fulfil. During this first activity, it is necessary to involve experts in the domain to generate the appropriate industry perspective and knowledge. Figure 2 shows the workflow for the ontology requirements specification activity.

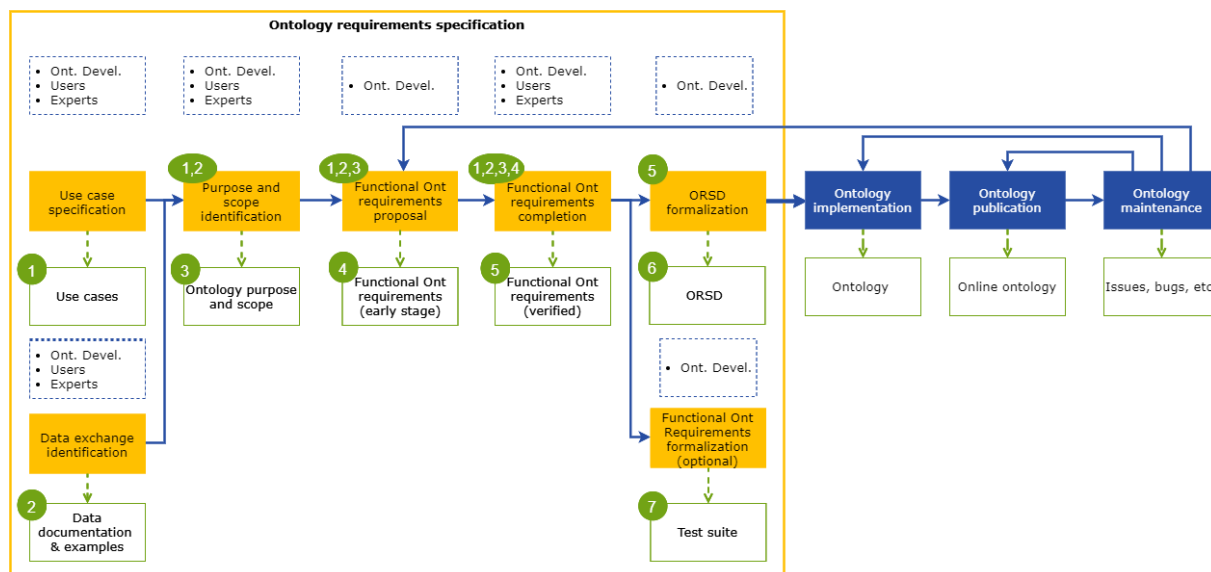


Figure 2 - Workflow for ontology requirements specification

### 2.1.1 Identification of Purpose and Scope

The goal of this step is to define the purpose and scope of the given ontology or module. To this end, the ontology development team works in collaboration with users and domain experts to define the purpose and scope of each ontology or module to be developed.

### 2.1.2 Data Exchange Identification and Use Case Specification

During this step, the ontology development team needs to gather the necessary documentation about the domain to be modelled. This documentation might correspond to the following.

- Standards
- Datasets
- API specifications
- Use cases

This documentation needs to be collected by both the ontology development team and domain experts.

### 2.1.3 Ontological Requirement Proposal

Taking as input the documentation and the data provided by domain experts and users, the ontology development team generates a first proposal of ontological requirements written in the form of Competency Questions or assertions.

The format used for this requirement proposal follows a tabular approach and includes the following fields:

- Requirement identifier, which must be unique for each requirement.
- Partner who proposed the requirement.
- Component of the COGITO architecture from which the requirement was extracted.
- Sprint in which the requirement is planned to be implemented.
- Competency question or assertion.
- Status of the requirement, which can be: (1) Proposed, (2) Accepted, (3) Rejected, (4) Pending, or (5) Deprecated.
- In case the requirement is deprecated, the identifier of the updated requirement is used.
- Comments on the requirement.
- Provenance of the requirement, e.g., use case or standard.
- Priority of the requirement, which can be: (1) High, (2) Medium, or (3) Low.

The ontological requirements are completed iteratively, as the following ontology development process is incremental.

#### 2.1.4 Completion and Validation of Ontology Requirements

During this activity, domain experts and users, in collaboration with the ontology development team, validate whether the ontology requirements defined in the previous step are correct and complete.

### 2.1.5 Prioritization of Ontological Requirements

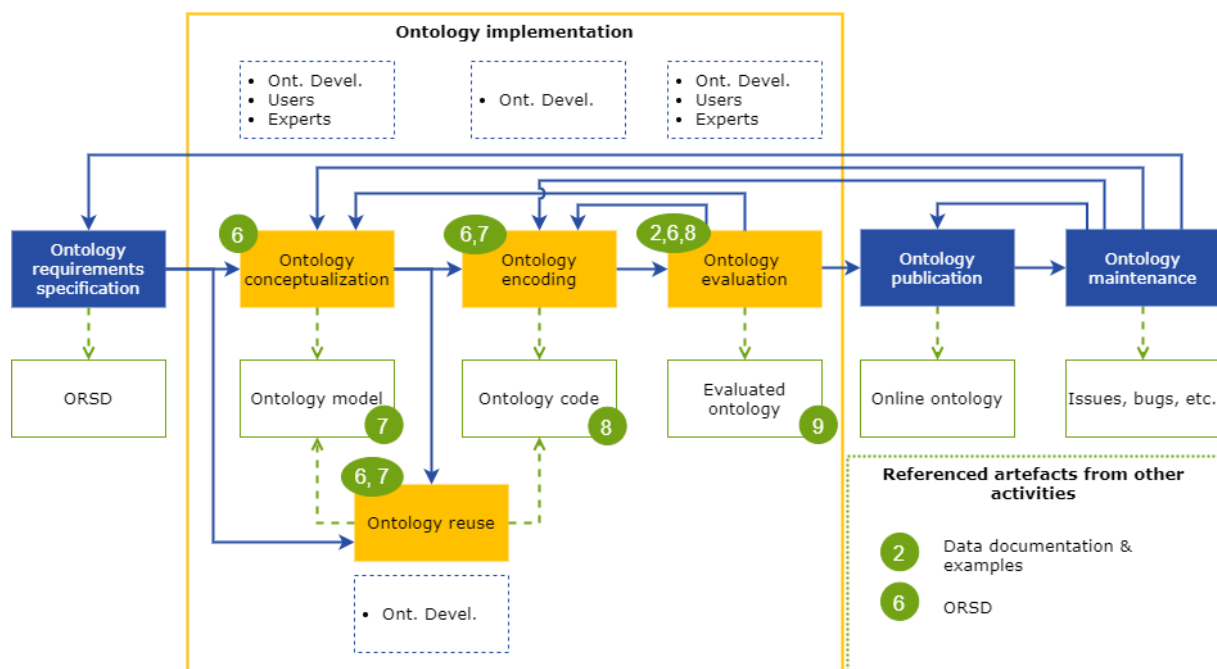
Prioritization of requirements allows development teams to schedule the development of the ontology in sprints. In the COGITO project, this prioritization will be performed if there is a need to prioritise functional requirements, mainly for those sprints with a high number of requirements in which the implementation of some of the requirements has to be postponed for a future sprint.

If this prioritisation is needed, to carry it out, the ontology development team works with the domain experts to identify which requirements need to be fulfilled first.

## 2.2 Ontology Implementation

During the implementation activity, the ontology is built using a formal language based on the requirements identified in the previous activity.

Taking as input the set of requirements collected in the previous activity, the ontology implementation activity is carried out through several sprints. To this end, the ontology developers schedule the ontology development according to the requirements that were identified, and the ontology development team builds the ontology iteratively by implementing only a certain number of requirements in each iteration. The output of each iteration is a new version of the ontology implementation. Figure 3 shows the steps that are followed in this ontology implementation activity.



**Figure 3 - Workflow for the ontology implementation activity**

### 2.2.1 Ontology Conceptualization

The aim of this activity is to build an ontology model from the ontological requirements identified in the requirements specification process that represents the domain of the ontology. Therefore, during the

conceptualization of the ontology, the domain knowledge obtained from the previous activity is organized and structured into a model by the ontology development team.

### 2.2.2 Ontology Encoding

The purpose of the encoding activity is to implement the ontology in an implementation language, such as OWL. The ontology code resulting from this activity includes, in addition to the ontology classes, properties, and axioms. Furthermore, following the FAIR principles [9], the ontology code also includes ontology metadata, such as the creator, title, publisher, license, and version of the ontology, in addition to the metadata for each of the ontology.

To manage the ontology versions developed in the COGITO project, the following version convention was adopted. Following this convention, each release will follow the pattern v.major.minor.fix, where each field follows the rules:

- **major:** The field is updated when the ontology covers the entire domain that it intends to model. That is, it is a complete product and covers the final goal of the development.
- **minor:** The field is updated when:
  - All the requirements of a subdomain are covered.
  - Documentation is added to the ontology.
- **fix:** The field is updated when:
  - Typos or bugs are corrected in the ontology.
  - Classes, relationships, axioms, individuals, or annotations are added, deleted, or modified, but the domain is not covered.

In each iteration, the minor and fix fields might be changed from zero to several times.

### 2.2.3 Ontology Evaluation

Once the ontology is encoded, it should be evaluated before its online publication. The development of the ontology must guarantee the following aspects:

- The ontology is consistent.
- The ontology does not have syntactic, modelling, or semantic errors.
- The ontology fulfils the requirements scheduled for the ontology, to ensure that the ontology is completed regarding the domain experts needs.

## 2.3 Ontology Publication

During the ontology publication activity, the ontology development team provides an online ontology that is accessible both as a human-readable document and as a machine-readable file from its URI. Figure 4 shows an overview of the steps performed during the ontology publication activity.

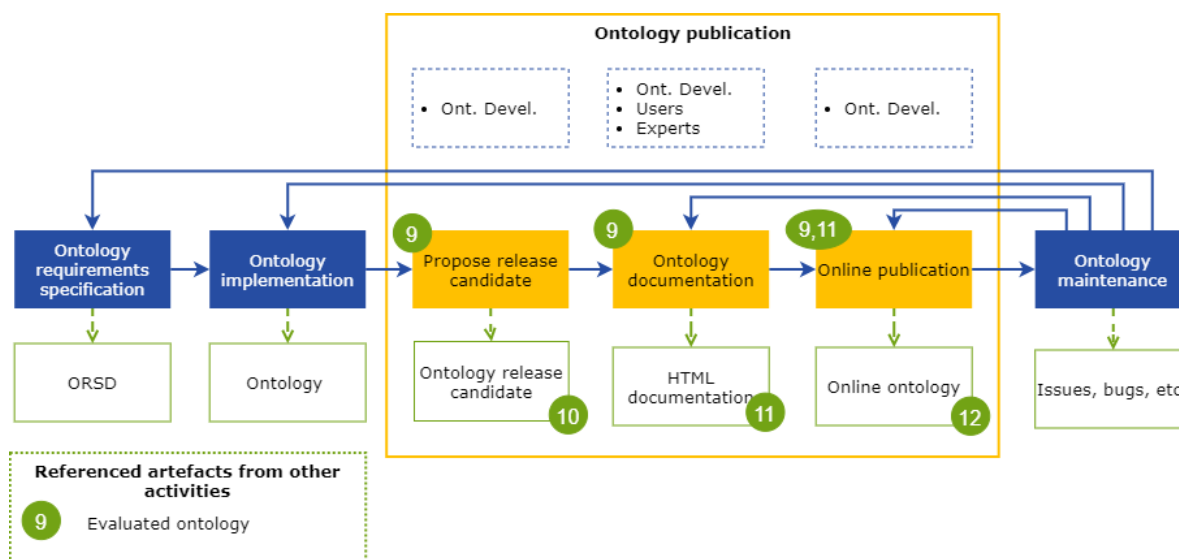


Figure 4 - Workflow for the ontology publication activity

### 2.3.1 Propose Release Candidate

Once the ontology developers have implemented and evaluated the ontology, the next task is to decide whether the current version is going to be published on the Web (or shared among other project partners, for example software developers making use of the ontology) or whether is needed to document such version of the ontology for any other reason (for example a set of requirements are fulfilled and triggers a new release). In this case, the version at hand becomes a release candidate. In case the ontology is not selected as release, the ontology development team generates a pre-release version of the ontology. Both release and pre-release versions of the ontologies are evaluated and ready to be used.

### 2.3.2 Ontology Documentation

Taking the ontology generated in the previous activities as input, the ontology development team generates the ontology documentation. This documentation includes the following:

- An HTML description of the ontology that describes the classes, object properties, and data properties of the ontology, the license URI, and title being used. Domain experts must collaborate with the ontology development team to describe classes and properties. This description also includes metadata such as creator, publisher, date of creation, last modification, or version number. It also includes links to different formats of serialization of the ontology, such as TTL, JSON-LD, or RDF/XML.
- Diagrams that store the graphical representation of the ontology, including taxonomy and class diagrams.

### 2.3.3 Online Publication

During this activity, the ontology, which should already be validated and documented, is published on the Web. This ontology is accessible through its URI as a machine-readable and human-readable file using content negotiation.

## 2.4 Ontology Maintenance

During this activity, the ontology is updated, and new requirements can be proposed to be added to the ontology. Furthermore, during this activity, the ontology development team, together with domain experts and users, can identify and correct errors in the ontology. Figure 5 shows the steps to follow in the maintenance activity of the ontology.

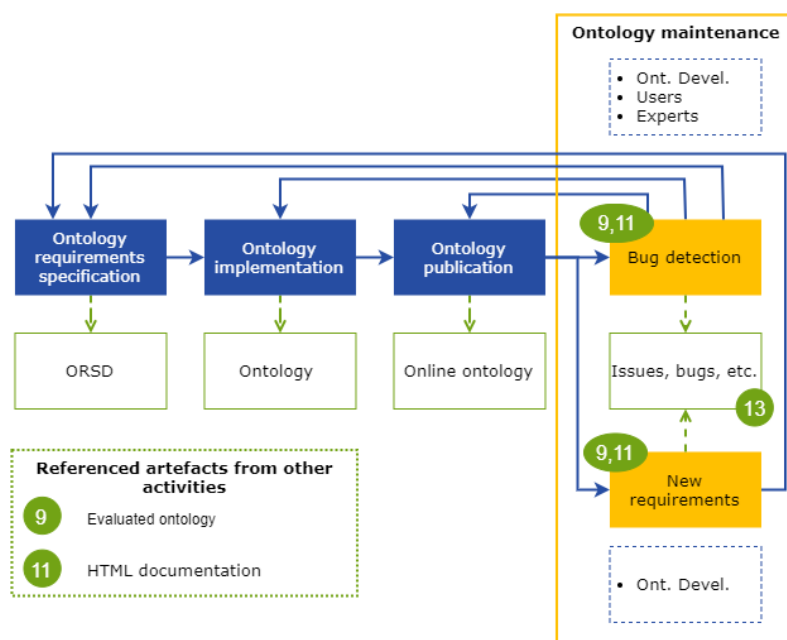


Figure 5 - Workflow for the ontology maintenance activity

### 2.4.1 Bug Detection

Once the ontology developers have published the ontology, any user, developer, or domain expert can detect and inform about bugs. This notification should be done using an issue tracker, allowing all the information related to the bug, as well as the actor that identifies it, to be stored.

### 2.4.2 New Requirements Proposal

During this activity, new requirements can be proposed for the ontology. Ontology developers, domain experts, or users can propose modifications to improve the published ontology version. This proposal should be done through an issue tracker so that all the information related to it is stored.



### 3 Ontology Requirements Specification

Since the goal of the ontology is to facilitate data sharing and interoperability among the COGITO components through the Digital Twin Platform (see Table 1), we decided to analyse each component of the architecture, defined in D2.4 [3], in order to gather information about the data it needs as input from other components and the data it produces as output to other components.

Several meetings with the partners involved in the components -so far, Process Modelling and Simulation (PSM), Work Order Definition and Monitoring tool (WODM), and Internet of Things (IoT) data preprocessing- have helped us to specify requirements aligned with the needs of those components. These requirements per component have been reorganised into three different domains: the construction itself (that is, the result of a construction process), the construction process and the resources used in that process. Table 2 presents the list of requirements for the domains we have identified.

**Table 1 – General information about the ontology requirements specification**

<b>Purpose</b>	To facilitate data sharing and interoperability among the COGITO components through the Digital Twin Platform
<b>Scope</b>	Limited to the data shared among COGITO components through the Digital Twin Platform.
<b>Implementation language</b>	Web Ontology Language (OWL)
<b>Intended end-users</b>	COGITO components and application developers COGITO end-users and stakeholders
<b>Non-functional ontology requirements</b>	Annotated in English Linked to standards when possible Open license Available online
<b>Ontology functional requirements</b>	Detailed information on the COGITO ontology portal

**Table 2 - List of requirements per domain**

Identifier (domain+id)	Component	Competency Question / Fact
CONS-1	PMS	A construction can contain one or several zones
CONS-2	PMS, WODM	A construction can have one or several spaces
CONS-3	PMS	A zone can have one or several spaces
CONS-4	PMS	A (construction) element has quality information
CONS-5	PMS	A (construction) element is a spatial thing
CONS-6	WODM	A construction is composed by elements
CONS-7	WODM	Each (construction) element has a location
PROC-1	WODM	A task has quality information
PROC-2	WODM	A task is related to one or several (construction) elements
PROC-3	WODM	A (project) process has a cost, which is measured in a certain currency
PROC-4	WODM	A task belongs to a certain process
PROC-5	WODM	A task includes progress information
PROC-6	WODM	A task can have a date of creation
PROC-7	WODM	A task can have an order in the workflow
PROC-8	WODM	A task is related to one or several (construction) elements

PROC-9	WODM	A task has requirements regarding resources
PROC-10	PMS	Resources can be allocated to a task
RESO-1	PMS, WODM	Workers, equipment and materials are resources
RESO-2	PMS	A resource is a spatial thing
RESO-3	WODM	A resource has a location
RESO-4	WODM	A worker has an identifier and a password
RESO-5	WODM	A worker performs a role for a certain task
RESO-6	WODM	A worker executes an action in a task
RESO-7	WODM	A piece of equipment has a status
RESO-8	IoT	A sensor measures the location
RESO-9	IoT	A location includes: altitude, longitude and latitude
RESO-10	IoT	A location also includes the time of the measure
RESO-11	IoT	A location includes the accuracy of the measure

## 4 Coverage Analysis

This section presents the process that has been followed to analyse the coverage of existing ontologies regarding the COGITO ontology requirements presented above in order to identify potential ontologies to be reused and the needs for extension. As mentioned in the previous sections, the set of ontology requirements was extracted from the use cases reported in D2.1 and the COGITO architecture that was extracted from D2.4.

### 4.1 Methodology

The methodology followed is shown in Figure 6. It is based on the ontology conformance approach proposed by Fernandez-Izquierdo and García-Castro [12] and consists of 5 steps:

1. To specify the set of ontology requirements considering the use cases and the COGITO architecture.
2. To identify the domains associated with the proposed requirements, for example, building or process.
3. To generate the set of test designs based on such requirements following the test syntax proposed in [12]. These tests should be abstract in the sense that they do not include any information about the ontology on which the tests are going to be executed, since they will be executed on multiple ontologies. The tests will be stored in an RDF file following the Verification Test Case ontology<sup>3</sup>.
4. To select the ontologies to be analysed. These ontologies were selected from the set collected in the survey presented in D3.1 [4] based on the domains identified in Step 2.
5. To execute the tests generated in Step 3 on the ontologies selected in Step 4.
6. To analyse the results obtained from the test execution in Step 5. This analysis should consider [12]:
  - The requirements that are *passed* by the analysed ontology, which represents the coverage between the analysed ontology and the COGITO requirement. Ontologies that have met the requirements could be considered for reuse.
  - The requirements that result in *undefined*, that is, the information that is included in the requirement, are out of the scope of the ontology.
  - The requirements that result in *absent*, i.e., the information related to restrictions and relations that are defined in the COGITO requirements but not in the ontology.
  - The requirements that result in *conflict*, that is, the incompatibilities between the COGITO requirements and the ontology.

---

<sup>3</sup> <https://w3id.org/def/vtc#>

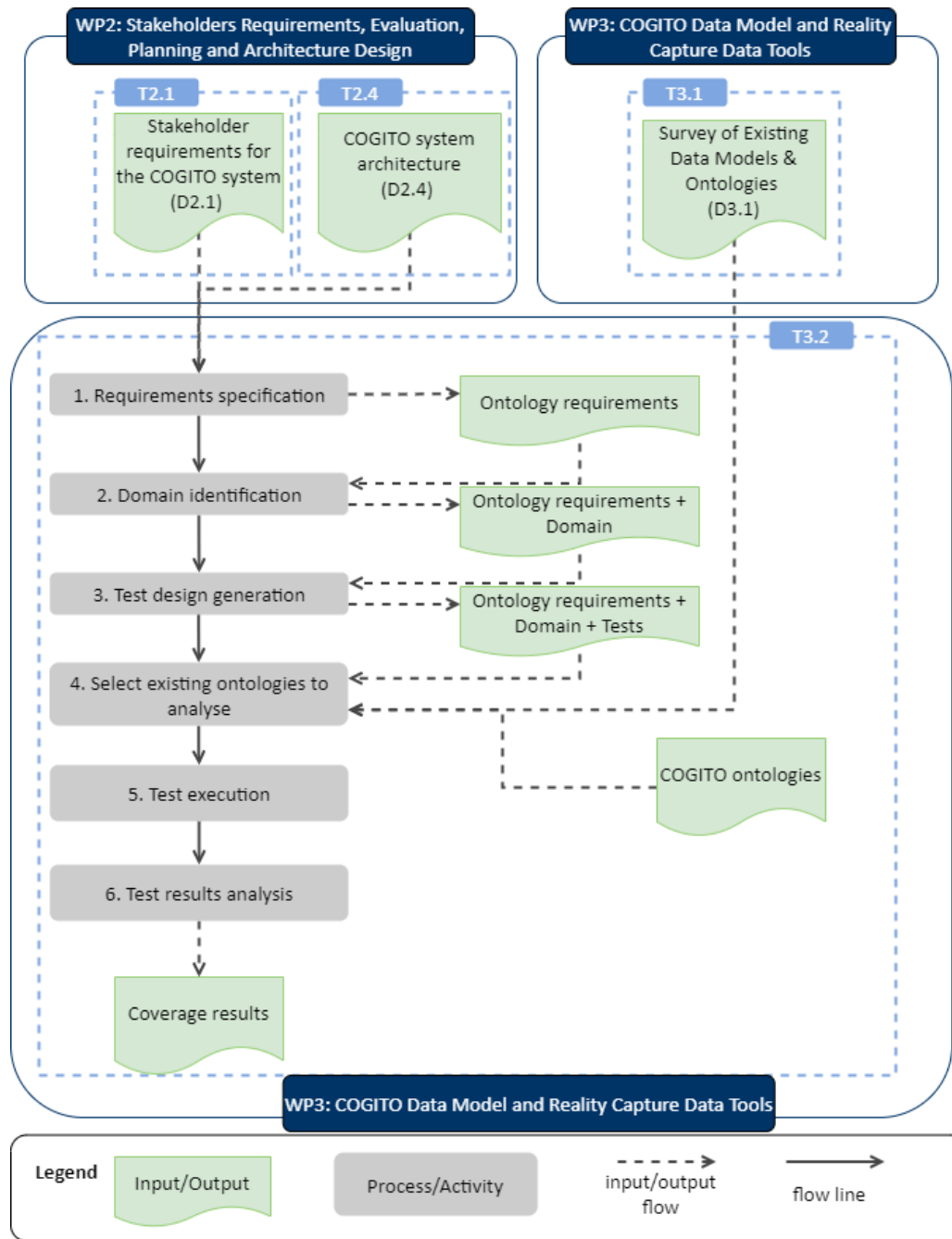


Figure 6 - Methodology for the coverage analysis

## 4.2 Results

This section presents the coverage results obtained after applying the methodology presented in the previous section. The requirements were divided into three modules, namely: (1) construction, (2) resources, and (3) construction process. These modules are related to the categories of requirements currently covered in the ontology.

### 4.2.1 Construction

Regarding the Construction module, we selected those ontologies identified in D3.1 related to the building domain, namely, ETSI SAREF4BLDG<sup>4</sup> [17], W3C BOT<sup>5</sup> [13], Brick<sup>6</sup>, and BPO<sup>7</sup>.

Table 3 shows the term coverage for each of the ontologies to be analysed and Table 4 shows the set of requirements proposed for the Construction domain and the test results for each of these ontologies. The tests used to check whether the requirements were satisfied are available online in the GitHub repository<sup>8</sup>, and the tables summarise the results of their execution.

**Table 3 – Term coverage results for the Construction domain**

Term	Test results			
	SAREF4BLDG	BOT	Brick	BPO
<b>containsZone</b>	undefined	passed	undefined	undefined
<b>hasElement</b>	undefined	passed	undefined	undefined
<b>hasQualityInformation</b>	undefined	undefined	undefined	undefined
<b>hasSpace</b>	passed	passed	undefined	undefined
<b>location</b>	passed	undefined	passed	undefined
<b>Construction</b>	undefined	undefined	undefined	undefined
<b>Element</b>	passed	passed	undefined	passed
<b>QualityInformation</b>	undefined	undefined	undefined	undefined
<b>Space</b>	passed	passed	passed	undefined
<b>SpatialThing</b>	undefined	undefined	undefined	undefined
<b>Zone</b>	undefined	passed	passed	undefined

**Table 4 – Requirement coverage results for the Construction domain**

ID	Requirement	Test results			
		SAREF4BLDG	BOT	Brick	BPO
<b>CONS-1</b>	A construction can contain one or several zones	undefined	undefined	undefined	undefined
<b>CONS-2</b>	A construction can have one or several spaces	undefined	undefined	undefined	undefined
<b>CONS-3</b>	A zone can have one or several spaces	undefined	passed	undefined	undefined
<b>CONS-4</b>	A (construction) element has quality information	undefined	undefined	undefined	undefined
<b>CONS-5</b>	A (construction) element is a spatial thing	undefined	undefined	undefined	undefined
<b>CONS-6</b>	A construction is composed by elements	undefined	undefined	undefined	undefined
<b>CONS-7</b>	Each (construction) element has a location	undefined	undefined	undefined	undefined

The tables clearly show that the requirements in the Construction domain are not directly covered by any of the ontologies. This is expected, since one of the main concepts in these requirements is the construction one and the analysed ontologies are mostly focused on buildings.

<sup>4</sup> <https://saref.etsi.org/saref4bldg/>

<sup>5</sup> <https://w3c-lbd-cg.github.io/lbd/bot/>

<sup>6</sup> <https://brickschema.org/>

<sup>7</sup> <https://www.projekt-scope.de/ontologies/bpo/>

<sup>8</sup> <https://github.com/oeg-upm/cogito-building-ontology/blob/main/tests/testsuite.ttl>

On the other hand, analysing the coverage according to the terms that appear in the requirements it can be seen that SAREF4BLDG, BOT and Brick already include some of the required terms. The current requirements are focused on spatial and topological information (covered by those three ontologies) and on construction elements (covered by SAREF4BLDG, BOT, and BPO). SAREF4BLDG and BOT are potential candidates to be reused in the COGITO ontology, since they cover these two aspects. In order to represent spatial and topological information, even if one of the ontologies is reused, the approach to follow should try to be compatible with as many approaches as possible. Besides, if in the future there are more requirements related to the schematic description of building products, the BPO ontology is a good candidate to be reused.

The tables also allow one to identify those requirements that are currently out of the scope of the analysed ontologies and that could lead to potential extensions. One of them, as mentioned before, is the description of the construction, which encompasses more information than a perspective only based on buildings. The other is the quality of the construction elements.

#### 4.2.2 Resource

Regarding the Resource module, we selected those ontologies identified in D3.1 related to the general domain, namely, DICO<sup>9</sup>, ifcOWL<sup>10</sup> and BIMERR<sup>11</sup>.

Table 5 shows the term coverage for each of the ontologies to be analysed and Table 6 shows the set of requirements proposed for the Resource domain and the test results for each of the ontologies. The tests used to check whether the requirements were satisfied by these ontologies are available online in the GitHub repository<sup>12</sup>, and the tables summarise the results of their execution.

<sup>9</sup> <https://digitalconstruction.github.io/>

<sup>10</sup> [https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2\\_TC1/OWL#](https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2_TC1/OWL#)

<sup>11</sup> <https://bimerr.iot.linkeddata.es/>

<sup>12</sup> <https://github.com/oeg-upm/cogito-resources-ontology/blob/master/tests/testsuite.ttl>

Table 5 – Term coverage results for the Resource domain

Term	Test results		
	DICO (Entities)	ifcOWL	BIMERR (Material, Process)
executesAction	undefined	undefined	undefined
hasAccuracy	undefined	undefined	undefined
hasAltitude	passed	passed	undefined
hasIdentifier	undefined	undefined	undefined
hasLatitude	passed	passed	undefined
hasLocation	passed	passed	undefined
hasLongitude	passed	passed	undefined
hasRole	passed	passed	undefined
hasStatus	undefined	passed	undefined
hasTimestamp	undefined	undefined	undefined
inTask	undefined	undefined	undefined
measures	undefined	undefined	undefined
participatesInTask	undefined	undefined	undefined
Action	undefined	undefined	undefined
Equipment	passed	undefined	undefined
Location	passed	passed	undefined
Material	undefined	passed	passed
Resource	undefined	passed	undefined
Role	undefined	passed	undefined
Sensor	undefined	passed	undefined
SpatialThing	undefined	passed	undefined
Status	undefined	passed	undefined
Task	undefined	passed	passed
Worker	undefined	undefined	passed
WorkerParticipation	undefined	undefined	undefined

Table 6 – Requirement coverage results for the Resource domain

ID	Requirements	Results		
		DICO (Entities)	ifcOWL	BIMERR (Material, Process)
RESO-1	Workers, equipment and materials are resources	undefined	undefined	undefined
RESO-2	A resource is a spatial thing	undefined	undefined	undefined
RESO-3	A resource has a location	undefined	undefined	undefined
RESO-4	A worker has an identifier and a password	undefined	undefined	undefined
RESO-5	A worker performs a role for a certain task	undefined	undefined	undefined
RESO-6	A worker executes an action in a task	undefined	undefined	undefined
RESO-7	A piece of equipment has a status	undefined	undefined	undefined
RESO-8	A sensor measures the location	undefined	undefined	undefined
RESO-9	A location includes: altitude, longitude and latitude	passed	undefined	undefined
RESO-10	A location also includes the time of the measure	undefined	undefined	undefined
RESO-11	A location includes the accuracy of the measure	undefined	undefined	undefined

The tables show that, as in the previous case, the requirements in the Resource domain are not directly covered by any of the ontologies. However, it can be seen that the DICO Entities and ifcOWL ontologies cover several terms of the requirements, while the BIMERR ontologies cover only a few of them. ifcOWL covers more terms than the DICO Entities ontology, due to its more general scope, and has potential to be reused. In any case, the DICO ontologies also have the potential to be reused as a result of their focus in the construction domain. The BIMERR ontologies may be relevant if in the future the requirements go deeper into the description of materials.

Out of the ontologies scope of the analysed is the description of workers and their activities in the construction process, which could lead to a potential extension. Another thing that is not present in the ontologies is detailed information about sensors and their accuracy, but this is something already covered by IoT ontologies such as SSN or SAREF.

#### 4.2.3 Construction Process

Regarding the resources module, we selected those ontologies identified in D3.1 related to the general domain, namely, DICO Processes ontology, BBO<sup>13</sup> [14] and W3C Time<sup>14</sup>.

Table 7 shows the term coverage for each of the ontologies to be analysed and Table 8 shows the set of requirements proposed for the Construction Process domain and the test results for each of the ontologies. The tests used to check whether the requirements were satisfied by these ontologies are available online in the GitHub repository<sup>15</sup>, and the tables summarise the results of their execution.

<sup>13</sup> <https://www.irit.fr/recherches/MELODI/ontologies/BBO#>

<sup>14</sup> <https://www.w3.org/TR/owl-time/>

<sup>15</sup> <https://github.com/oeg-upm/cogito-construction-process-ontology/blob/master/tests/testsuite.ttl>



Table 7 – Term coverage results for the Construction Process domain

Term	Test results		
	DICO (Processes)	BBO	W3C Time
belongsTo	undefined	undefined	undefined
hasCost	undefined	undefined	undefined
hasCreationDate	undefined	undefined	undefined
hasOrder	undefined	undefined	undefined
hasProgress	undefined	undefined	undefined
hasQualityInformation	undefined	undefined	undefined
isAllocatedTo	undefined	undefined	undefined
isMeasuredIn	undefined	undefined	undefined
isRelatedTo	undefined	undefined	undefined
requiresResource	undefined	undefined	undefined
Cost	undefined	undefined	undefined
Currency	undefined	undefined	undefined
Element	undefined	undefined	undefined
Process	undefined	passed	undefined
QualityInformation	undefined	undefined	undefined
Resource	undefined	undefined	undefined
Task	passed	passed	undefined

Table 8 – Requirement coverage results for the Construction Process domain

ID	Requirement	Results		
		DICO (Processes)	BBO	W3C Time
PROC-1	A task has quality information	undefined	undefined	undefined
PROC-2	A task is related to one or several (construction) elements	undefined	undefined	undefined
PROC-3	A (project) process has a cost, which is measured in a certain currency	undefined	undefined	undefined
PROC-4	A task belongs to a certain process	undefined	undefined	undefined
PROC-5	A task includes progress information	undefined	undefined	undefined
PROC-6	A task can have a date of creation	undefined	undefined	undefined
PROC-7	A task can have an order in the workflow	undefined	undefined	undefined
PROC-8	A task has requirements regarding resources	undefined	undefined	undefined
PROC-9	Resources can be allocated to a task	undefined	undefined	undefined

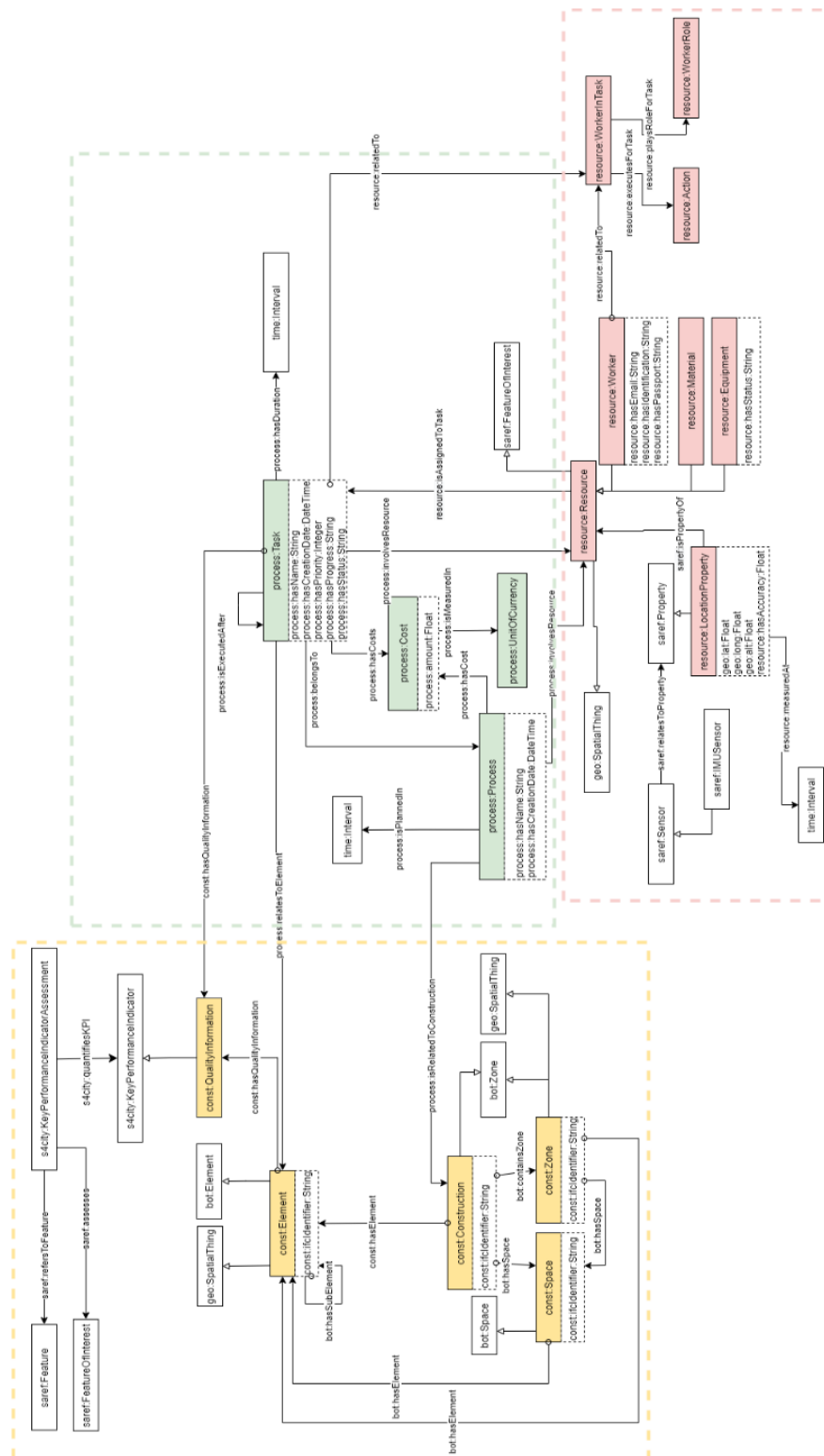
As in the previous cases, the requirements in the Construction Process domain are not directly covered by any of the ontologies. Furthermore, in this case, the coverage of terms is minimal (only the Task term is covered by DICO Processes and BBO).

The W3C Time ontology does not cover any requirement or term, which makes sense since the current requirements do not deal with the detailed representation of temporal information. Besides, both DICO Processes and BBO are intended to represent processes, but they do it from different perspectives: DICO Processes focuses on the description of activity flows and changes of states through conditions and effects, and BBO is a formalization of BPMN. Therefore, none of the analysed ontologies is a candidate to be reused right now; however, once we have more requirements, we will analyse them again.

The analysis shows a clear gap in the representation of processes according to their relationship with the construction process, i.e., the resources (workers, equipment, and materials) involved in the process, the quality of tasks, and the cost of tasks.

## 5 Overview of the COGITO Ontology Network

The ontology network developed for the COGITO project consists for now of three ontology modules, where each module corresponds to one specific domain, namely, construction, process, and resource. Figure 7, provides a graphical overview of the COGITO ontology network showing the main concepts defined in each module.



**Figure 7 – COGITO ontology network**

Coloured boxes are used to represent modules currently implemented, while white boxes represent reused terms from existing ontologies. As can be observed, some of the ontologies reviewed in previous steps of the project were reused according to the results of the process described in Section 4, like BOT. Additional cross-domain ontologies, such as WGS84, were also reused once requirements and conceptualizations were analysed in detail. The prefixes, and corresponding ontologies, created and reused in the COGITO ontology are listed in Table 9 and Table 10, respectively.

Table 9 – List of the ontologies created in the COGITO project

Prefix	Namespace
const	<a href="https://cogito.iot.linkeddata.es/def/construction#">https://cogito.iot.linkeddata.es/def/construction#</a>
process	<a href="https://cogito.iot.linkeddata.es/def/process#">https://cogito.iot.linkeddata.es/def/process#</a>
resource	<a href="https://cogito.iot.linkeddata.es/def/resource#">https://cogito.iot.linkeddata.es/def/resource#</a>

Table 10 – List of the ontologies reused in the COGITO project

Prefix	Namespace
bot	<a href="https://w3id.org/bot#">https://w3id.org/bot#</a>
geo	<a href="http://www.w3.org/2003/01/geo/wgs84_pos">http://www.w3.org/2003/01/geo/wgs84_pos</a>
saref	<a href="https://saref.etsi.org/core#">https://saref.etsi.org/core#</a>
s4city	<a href="https://saref.etsi.org/saref4city#">https://saref.etsi.org/saref4city#</a>
time	<a href="http://www.w3.org/2006/time#">http://www.w3.org/2006/time#</a>

The main hierarchies between concepts are also included. These hierarchies are represented by arrows with white endings (triangles) and can be read as follows: the class in the origin of the arrow is a subclass of the class at the end of the arrow. Ad hoc relations between different modules and within modules are also present. Arrows are used to represent these properties between classes and to represent some RDF, RDFS, and OWL constructs. More precisely:

- Plain arrows with white triangles represent the subclass relationship between two classes. The origin of the arrows is the class to be declared as a subclass of the class at the destination of the arrow.
- Plain arrows between two classes indicate that the object property has declared as domain the class in the origin and as range the class in the destination of the arrow. The identifier of the object property is indicated within the arrow.
- Dashed labelled arrows between two classes indicate that the object property can be instantiated between the classes in the origin and the destination of the arrow. The identifier of the object property is indicated within the arrow.
- Dashed arrows with the identifiers between stereotype signs (i.e., "<< >>") refer to OWL constructs that are applied to some ontology elements, that is, they can be applied to classes or properties depending on the OWL construct being used.
- Dashed arrows with no identifier are used to represent the *rdf:type* relation, indicating that the element in the origin is an instance of the class in the destination of the arrow.

Datatype properties are denoted by rectangles attached to the classes, in a UML-oriented way. Dashed boxes represent datatype properties that can be applied to the class it is attached to, while plain boxes represent that the domain of the datatype property is declared to be the class attached.

For more details about the graphical notation used in this diagram, you can see the Chowlk Visual Notation<sup>16</sup>.

The complete and up-to-date documentation of each ontology module is provided online, as explained in Section 3. In the rest of this section, only the main concepts and modelling decisions are detailed.

<sup>16</sup> [https://chowlk.linkeddata.es/chowlk\\_spec](https://chowlk.linkeddata.es/chowlk_spec)

## 5.1 Construction Module

The current conceptual model defined for the Construction module is depicted in Figure 8.

The main classes and properties in this module are based on the Building Topology Ontology (BOT), which is used to describe the topological concepts of a construction.

- **const:Zone**: is defined as a subclass of **bot:Zone** and, as such, a part of the physical or a virtual world that is inherently both located in this world and has a 3D spatial extent.
- **const:Space** is defined as a subclass of **bot:Space** and, as such, a part of the physical world or a virtual world whose 3D spatial extent is bounded actually or theoretically, and provides for certain functions within the zone it is contained in. It is also a subclass of **bot:Zone**.
- **const:Construction**: is defined as something constructed; a structure. It is a class which is not considered in BOT, but we can include it into the **bot:Zone** hierarchy, as well as **Space**. A **const:Construction** can contain **const:Zone** and **const:Space**, reflected in the diagram by **bot:containsZone** and **bot:hasSpace**, respectively.
- **const:Element** is defined as a subclass of **bot:Element** and, as such, constituent of a construction entity with a characteristic technical function, form, or position. A **const:Element** can contain sub-elements. A **const:Zone** can contain some **const:Elements** (and so do **const:Construction** and **const:Space**). A **const:Element** can have information about its quality (**const:QualityInformation**), which has been defined as a subclass of **s4city:KeyPerformanceIndicator**.

These sub-classes have been created because we foresee some specific data properties that will be needed for them. They are also subclasses of **geo:SpatialThing** in order to reuse its location properties.

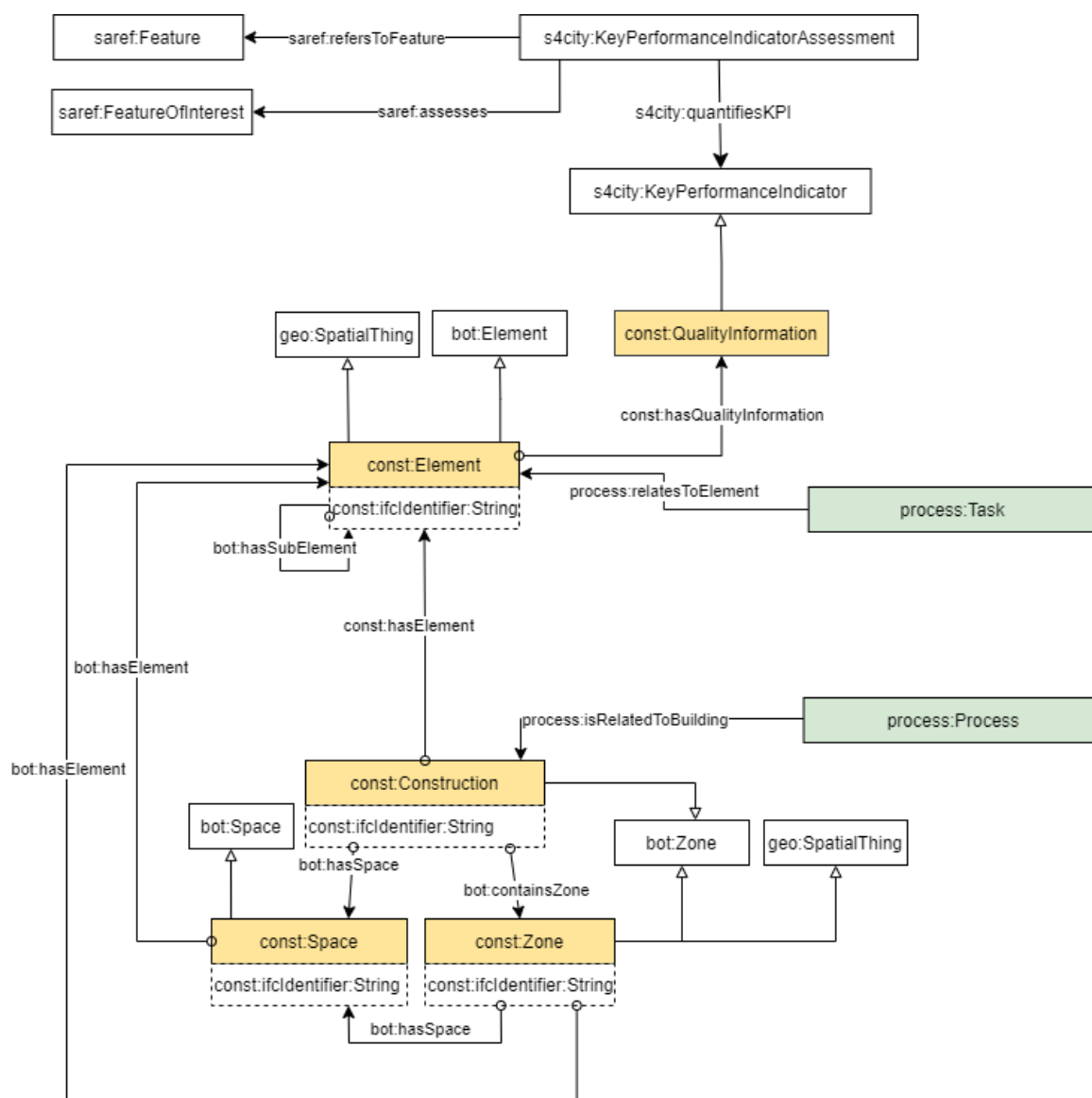


Figure 8 – General overview of the COGITO construction ontology

## 5.2 Process Module

The current conceptual model defined for the Process module is depicted in Figure 9. The main classes and properties in this module have been defined anew:

- `process:Process` is defined as a series of actions aimed at accomplishing some result (in this case, a `const:Construction`).
- `process:Task` is defined as a piece of work, which is carried out in a `process:Process` (`process:belongsTo`); and is related to a `const:Element`. A `process:Task` can have information about its quality (`cost:QualityInformation`) and a duration (`process:hasDuration`). We can include the `process:status` and the `process:progress` of a `process:Task`. A `process:Task` takes place at a time interval (`time:Interval`) defined by a beginning and an end instant.
- `process:Cost` is defined as the price paid to acquire, produce, accomplish, or maintain anything (in this case, `process:Process` and `process:Task`); and this price is measured (`process:measuredIn`) in a currency (`process:UnitOfCurrency`).

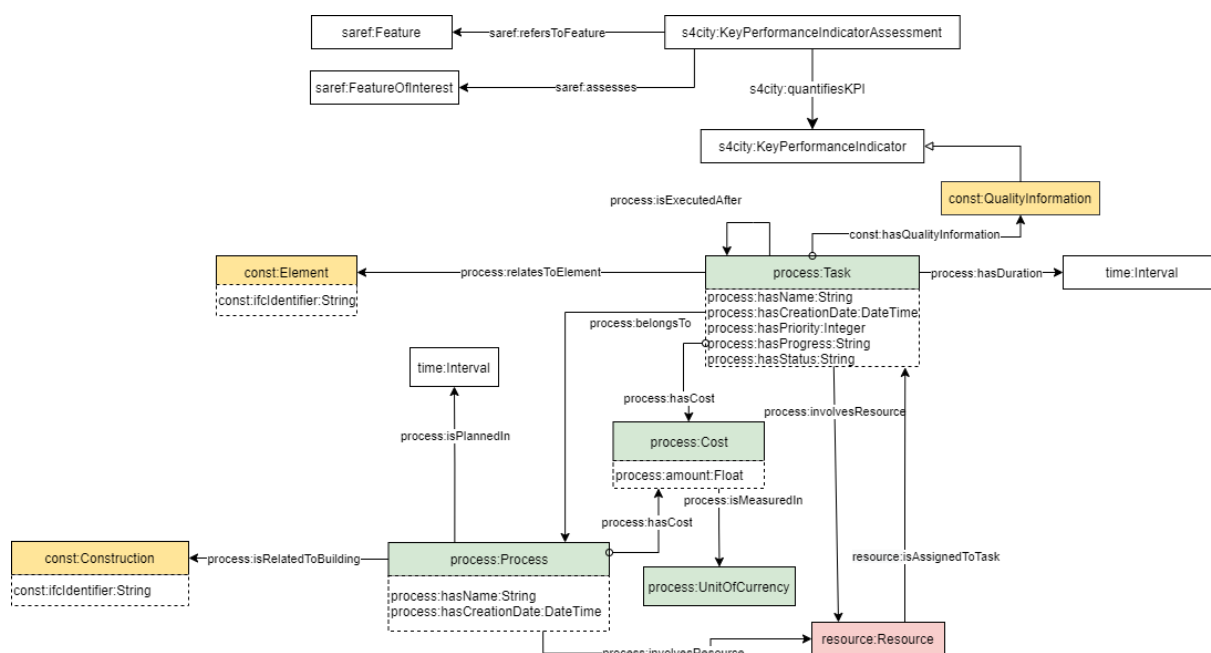


Figure 9 - General overview of the COGITO process ontology

### 5.3 Resource Module

The current conceptual model defined for the Resource module is depicted in Figure 10. The main classes and properties of this module are the following:

- `resource:Resource` is defined as a source of supply, support, or aid, especially one that can be readily drawn upon when needed. `resource:Worker`, `resource:Material`, and `resource:Equipment` are subclasses of `resource:Resource`; and all of them have a location (`resource:LocationProperty`). It is also a subclass of `geo:SpatialThing` in order to reuse its location properties.
- `resource:Worker` is defined as a laborer or employee who plays a role (`resource:WorkerRole`) and does an action (`resource:Action`) for a `process:Task`.
- `resource:LocationProperty` is defined as a place of activity or situation, its coordinates are geo properties of longitude, altitude, and latitude. Since a `saref:Sensor` will be used to measure this property, the accuracy of this measured is also relevant (`resource:hasAccuracy`), as well as the time it was measured (`time:Interval`).

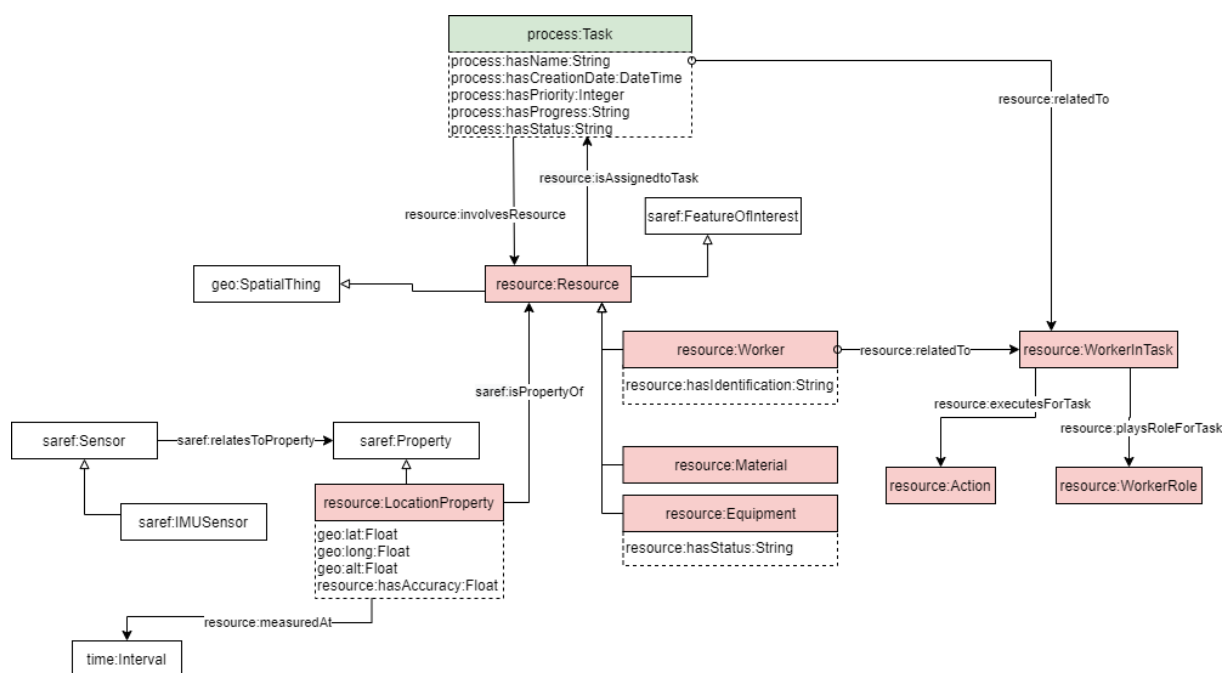


Figure 10 – General overview of the COGITO resource ontology



## 6 Ontology Development Infrastructure

This section describes the ontology development infrastructure used to support the activities in the ontology development process described above.

### 6.1 Infrastructure to Support the Requirements Specification Activity

The ontology development team used spreadsheets to store the requirements per component. An excerpt of these requirements is shown in Figure 11.

Identifier (component+id)	Competency Question / Natural language sentence (fact)	Answer	Status (Proposed, Accepted, Rejected, Pending, Deprecated)	Superseded by	Comments	Extracted from (provenance)	Priority (High, Medium, Low)
pms-1	A construction can contain one or several zones					minutes of meeting on PMS	
pms-2	A construction can have one or several spaces					minutes of meeting on PMS	
pms-3	A zone can have one or several spaces					minutes of meeting on PMS	
pms-4	Resources can be allocated to a task					minutes of meeting on PMS	
pms-5	Workers, equipment and materials are resources					minutes of meeting on PMS	
pms-6	A resource is a spatial thing					minutes of meeting on PMS	
pms-7	A (construction) element has quality information					minutes of meeting on PMS	
pms-8	A (construction) element is a spatial thing					minutes of meeting on PMS	
pms-9	A task has quality information					minutes of meeting on PMS	
pms-10	A task is related to one or several (construction) elements					minutes of meeting on PMS	

Figure 11 - COGITO ontology requirements for PMS

These requirements per component were reorganised per domain (see Section 3) and converted into an HTML file and uploaded to the COGITO ontology portal<sup>17</sup> with the most relevant information for users to facilitate visualization. Figure 12 shows an excerpt of the HTML documentation of the COGITO requirements.


Ontology requirements							
							
Here you can find the list of the requirements identified for the COGITO Construction Process ontology and their main features.							
Identifier	Component	Competency Question / Natural language sentence (fact)	Answer	Status	Superseded by	Priority	
PROC-1	WODM	A task has quality information					
PROC-2	WODM	A task is related to one or several (construction) elements					
PROC-3	WODM	A (project) process has a cost, which is measured in a certain currency					
PROC-4	WODM	A task belongs to a certain process					
PROC-5	WODM	A task includes progress information					
PROC-6	WODM	A task can have a date of creation					
PROC-7	WODM	A task can have an order in the workflow					
PROC-8	WODM	A task has requirements regarding resources					
PROC-9	PMS	Resources can be allocated to a task					

Figure 12 - HTML requirements for the construction process

<sup>17</sup> <https://cogito.iot.linkeddata.es/>

## 6.2 Infrastructure to Support the Ontology Implementation Activity

To support the implementation activity, the ontology development team uses several tools to edit, store, and evaluate the ontology.

- For ontology edition, the ontology development team uses Protégé,<sup>18</sup> which allows the creation, visualisation, and manipulation of ontologies.
- For ontology storage, the ontology development team uses GitHub<sup>19</sup>. A GitHub repository is created for each ontology in the COGITO ontology network. Each repository includes:
  - A folder with the implementation of the ontology.
  - A folder with the ontology modelling diagrams.
  - A folder with the documentation of the ontology.
  - A folder with the requirements and tests of the ontology.

The development team uses the OnToology<sup>20</sup> tool to generate documentation and evaluate the ontology. OnToology, which integrates the tools Widoco<sup>21</sup> [10] and OOPS!<sup>22</sup> [11], automatically generates a folder in the GitHub repository that includes the resources: diagrams, documentation, and evaluation report.

## 6.3 Infrastructure to Support the Ontology Publication Activity

To support the publication activity, the ontology development team creates an ontology portal online to make the ontology and all the associated information (repository, requirements, tests, releases, etc.) to users. This ontology portal has different sections:

- Ontologies
- How we work

### 6.3.1.1 Ontologies

The Ontologies section, which is the main section of the portal, shows the main information about the ontologies created in the COGITO ontology network. Figure 13 shows an overview of the information exposed in this section of the portal. The section follows a tabular approach which includes the following:

- Link to the ontology documentation published on the Web
- Ontology Description
- Link to each GitHub repository
- Links to each GitHub issue tracker
- HTML description of the requirements identified by the domain experts
- Link to each ontology release

<sup>18</sup> <https://protege.stanford.e>

<sup>19</sup> <https://github.com/orgs/oeg-upm/teams/cogito>


<sup>20</sup> <https://ontology.linkeddata.es/>

<sup>21</sup> <https://github.com/dgarijo/Widoco>

<sup>22</sup> <http://oops.linkeddata.es/>

Ontologies

Ontology testing



Here you can find the list of ontologies developed for COGITO project

If you want to contribute developing ontologies please follow the [guidelines](#) we provide

Ontology	Description	Requirements	Repository	Issue tracker	Releases
COGITO Construction Process ontology	This ontology aims to model the construction process in the COGITO ontology	<a href="#">Ontology Requirements</a>	<a href="#">Ontology Repository</a>	<a href="#">Ontology Issue Tracker</a>	<a href="#">Ontology Releases</a>
COGITO Construction ontology	This ontology aims to model the construction data exchanges in the COGITO project	<a href="#">Ontology Requirements</a>	<a href="#">Ontology Repository</a>	<a href="#">Ontology Issue Tracker</a>	<a href="#">Ontology Releases</a>
COGITO Resources ontology	This ontology aims to model the resources in the COGITO project	<a href="#">Ontology Requirements</a>	<a href="#">Ontology Repository</a>	<a href="#">Ontology Issue Tracker</a>	<a href="#">Ontology Releases</a>

Figure 13 - Ontology section in the COGITO ontology portal

#### 6.3.1.2 How We Work

Regarding the section 'how we work,' it provides a brief overview of the proposed process for developing ontologies and some guidelines, which should be useful to anyone who wants to contribute to the ontologies. This section includes information about the following.

- How should the repository be structured.
- Tools recommended to be used during the ontology development process.
- Ontology versioning.
- Management of issues.

### 6.4 Infrastructure to Support the Ontology Maintenance Activity

To provide support for maintenance activity, ontology developers use the GitHub issue tracker, which manages and maintains the list of issues identified by domain experts and ontology developers. The GitHub issue tracker provides the status of the issue, assignee, and description and allows one to add comments to the issue to discuss about it. Each issue tracker is associated with a GitHub repository and, consequently, with an ontology in the COGITO ontology network.

To manage changes in the ontology, all new proposals and improvements must be agreed upon by all members of the ontology development team. If domain experts, users, or ontology developers want to add, delete, or modify concepts in the ontology, they must create a new issue in the GitHub issue tracker associated with the ontology to be modified, which will be used to discuss the approval or rejection of the proposal. Figure 14 shows the GitHub issue tracker for one of the COGITO ontologies.

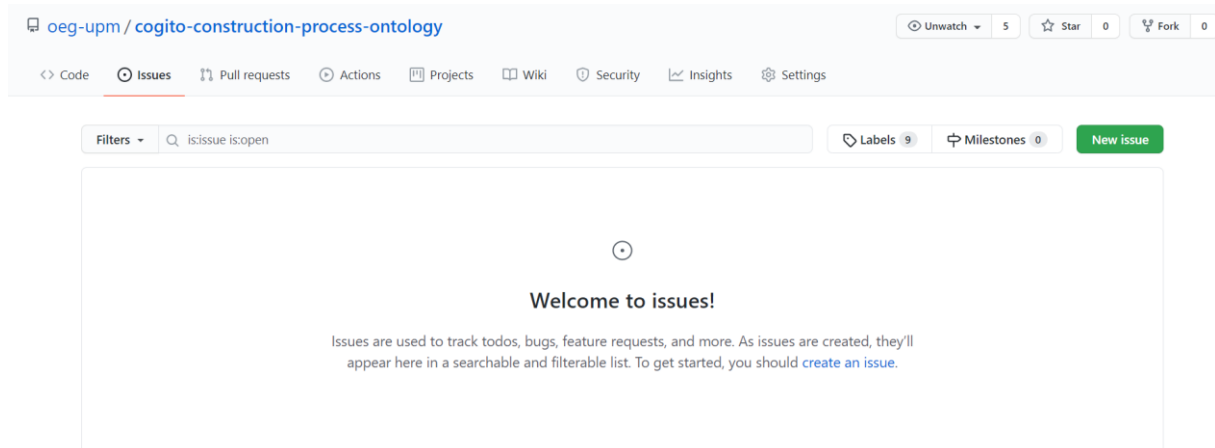


Figure 14 - Example of the GitHub issue tracker

## 7 Conclusions

This document details the methodology and technological infrastructure used to develop the COGITO ontology network, as well as the first version of the ontologies that make up such an ontology network. The development of these ontologies has been aligned with the gathering of requirements and with the specification of the COGITO architecture; since this is an ongoing task, the ontology network will evolve over time.

The current version of the COGITO ontology network (see Figure 7) consists of three modules, corresponding to the construction itself, the construction process, and the construction resources (see Table 9). These modules have already been conceptualised, implemented, and published in a first version of the COGITO ontologies and will be modified and complemented with new modules as the project progresses.

In the previous D3.1 deliverable, several existing ontologies that could be reused in COGITO were identified. However, as checked in the coverage analysis, the project requirements are not directly covered by them. This is something expected since different ontologies (even in the same domain) cover different aspects of their domain and are not focused on covering every potential scenario.

The coverage analysis allowed us to identify that some of the terms that appear in the requirements also appear in the ontologies to a greater or lower extent. This has helped identify some ontologies that are candidates to be reused in COGITO (ETSI SAREF4BLDG, W3C BOT, ifcOWL, and the DICO ontologies) and other ontologies that could be useful in the future if new requirements cover their specific domains.

In any case, during the implementation of the three current modules of the COGITO ontology network, it was clear that some ontologies fit our requirements better than others and are the ones reused in practice (W3C BOT, W3C Time, WGS84, ETSI SAREF, and ETSI SAREF4CITY). This highlights the fact that reusing ontologies requires analysing multiple factors beyond coverage, and some of them are not easily quantifiable or even cannot be measured objectively.

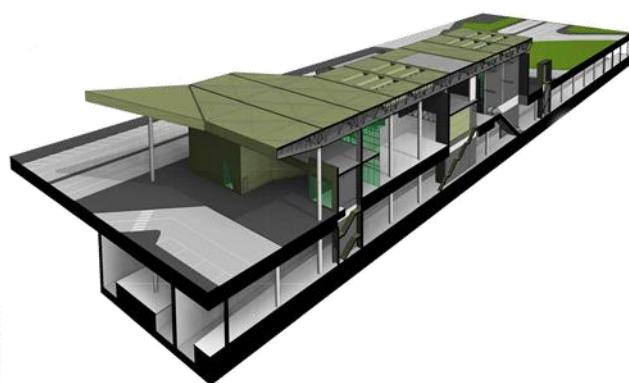
In any case, the coverage analysis allowed one to identify potential extensions currently not covered by the analysed ontologies such as: description of the construction as a whole, quality of construction elements, of relevant resources (workers, equipment, and materials) and their relationship with the construction process, and of metrics on the construction process such as the quality or the cost of tasks.

The COGITO ontology network will allow the sharing and interoperability among the different components of the COGITO architecture, especially in the communications with the Digital Twin platform. That is why this document and the future versions of the ontology will be so important when developing each component of the COGITO architecture, that is, the future results of work packages 4, 5, 6, and 7, and their integration in work package 8.

The next steps will be devoted to continuing to collect more ontological requirements from the pilots and from the architecture to iteratively evolve the COGITO ontology network. The COGITO ontology portal will serve as a living repository for the different ontologies and will contain the latest version of the developed ontologies and of all the related artifacts.

## References

- [1] M. Poveda-Villalón, A. Fernández-Izquierdo and R. García-Castro, “Linked Open Terms (LOT) Methodology,” January 2019. [Online]. Available: <http://doi.org/10.5281/zenodo.2539305>.
- [2] “D2.1 - Stakeholder Requirements for the COGITO System,” 2021.
- [3] “D2.4- COGITO System Architecture,” 2021.
- [4] “D3.1- Survey of Existing Models, Ontologies and Associated Standardization Efforts,” 2021.
- [5] M. C. Suárez-Figueroa, A. Gómez-Pérez and M. Fernández-López, “The NeOn Methodology for Ontology Engineering,” in *Ontology engineering in a networked world*, Springer, Berlin, Heidelberg, 2011.
- [6] R. García-Castro, A. Fernández-Izquierdo, C. Heinz, P. Kostelnik, Poveda-Villalón-María and F. Serena, “D2.2. Detailed Specification of the Semantic Model,” 2017.
- [7] S. Chávez, F. Bosché, N. Bountouni, S. Fenz, R. García-Castro, G. Giannakis, S. González-Gerpe, S. Kollmer, S. Kousouris, D. Ntalaperas, T. Thanos, F. Lampathaki, M. Poveda-Villalón, E. Valero, D. Vergeti and F. Tavakolizadeh, “D4.2 BIMERR Ontology & Data Model,” 2020.
- [8] A. Fernández-Izquierdo, A. Cimmino, R. García-Castro, M. Poveda-Villalón, S. Terzi and C. Patsonakis, “T1.3 DELTA Ontology and Data Modelling Framework,” 2019.
- [9] D. Garijo and M. Poveda-Villalón, “Best Practices for Implementing FAIR Vocabularies and Ontologies on the Web,” in *Applications and Practices in Ontology Design, Extraction, and Reasoning*, IOS Press, 2020.
- [10] D. Garijo, “WIDOCO: a wizard for documenting ontologies,” in *International Semantic Web Conference*, 2017.
- [11] M. a. S.-F. M. C. a. G.-P. A. Poveda-Villalón, “Validating ontologies with oops!,” 2012.
- [12] A. Fernández-Izquierdo and R. García-Castro, “Conformance testing of ontologies through ontology requirements,” *Engineering Applications of Artificial Intelligence*, vol. 97, 2021.
- [13] M. H. Rasmussen, P. Pauwels, C. Hviid and J. Karlshøj, “Proposing a Central AEC Ontology That Allows for Domain Specific Extensions,” in *Joint Conference on Computing in Construction*, 2017.
- [14] A. Annane, N. Aussenac-Gilles and M. Kamel, “BBO: BPMN 2.0 Based Ontology for Business Process Representation,” in *20th European Conference on Knowledge Management (ECKM'19)*, Lisbon, 2019.
- [15] ISO 16739, “Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries (ISO 16739:2013),” CEN, 2016.
- [16] ISO 13790, “Energy performance of buildings — Calculation of energy use for space heating and cooling,” 2008.
- [17] M. C. Suárez-Figueroa, A. Gómez-Pérez and B. Villazón-Terrazas, “How to write and use the ontology requirements specification document,” in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, 2009.



# COGITO

CONSTRUCTION PHASE  
DIGITAL TWIN MODEL

[cogito-project.eu](http://cogito-project.eu)



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958310