



D3.2 Hospital Knowledge Base and ODIN semantic ontology v1

Deliverable No.	D3.2	Due Date	28/02/2022
Description	This deliverable shows the first version of the ODIN semantic ontology, together with a literature search about the existing ontologies for healthcare and a practical application to one pilot		
Type	Report	Dissemination Level	PU
Work Package No.	WP3	Work Package Title	Platform integration, Privacy, Security and Trust + knowledge + cognition
Version	1.0	Status	Final



Authors

Name and surname	Partner name	e-mail
Ernesto Iadanza	UoW	ernesto.iadanza@warwick.ac.uk
Oralda Xhani	UoW	oralda.xhani@stud.unifi.it
Camilla Petraccone	UoW	camilla.petraccone @stud.unifi.it
Emanuele Casciaro	UoW	emanuele.casciaro@stud.unifi.it
Matt Philips	UoW	matt.philips.1@warwick.ac.uk
Pablo Lombillo	MYS	plombillo@mysphera.com
Juan Gonzalez Cabello	INETUM	juan.gonzalez@inetum.com
Alejandro M. Medrano Gil	UPM	amedrano@lst.tfo.upm.es
Ilias Kalamaras	CERTH	kalamar@iti.gr
Marcello Chiurazzi	SSSA	marcello.chiurazzi@santannapisa.it

History

Date	Version	Change
04/01/2022	0.1	Initial TOC
10/02/2022	0.2	Updated TOC and draft contents added
15/02/2022	0.3	Expanded some sections
18/02/2022	0.4	Added contributions on AI and IoT
25/02/2022	0.5	All sections completed, except from Chapter 7 and paragraph 8.1
28/02/2022	0.6	All sections completed – sent for internal review
07/03/2022	0.7	Improved figures in Section 5 for readability (suggestion from ROB)
21/03/2022	0.9	Completed after suggestions from UPM (second internal reviewer)
28/03/2022	1.0	Final Review and ready for submission

Key data

Keywords	Semantic web, ontologies, EMDN, JSON-LD, Turtle, RDF, OWL, SPARQL
Lead Editor	UoW
Internal Reviewer(s)	ROB, UPM

Abstract

This Deliverable discusses the first version (V1) of the ODIN semantic ontology, a platform that, through the Internet of Things (IoT), Robotics and Artificial Intelligence (AI), optimizes the context of the hospitals participating in the ODIN project. The analysis of the scientific publications made it possible to identify the articles that describe the most relevant ontologies for the purposes of the project. The ontologies that meet the requirements of all use cases were then chosen. ODIN's V1 was implemented using the Protégé software, whose classes, object properties and data properties build a semantic platform that fully represents the hospital environment. Through a formal language, it can be used as an automatic model that combines IoT, robotic agents and AI to improve worker activity and logistics. The contribution of this work constitutes a starting point for the realization of the objectives that the project sets itself, creating a database that facilitates the exchange of information in the field of hospital structures.

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Table of contents

TABLE OF CONTENTS	4
LIST OF TABLES	7
LIST OF FIGURES.....	8
1 INTRODUCTION	10
1.1 DELIVERABLE CONTEXT	10
2 SEMANTIC ONTOLOGIES	12
2.1 RDF – RESOURCE DESCRIPTION FRAMEWORK.....	13
2.2 OWL – ONTOLOGY WEB LANGUAGE	15
2.2.1 <i>OWL Lite</i>	16
2.2.2 <i>OWL DL</i>	18
2.2.3 <i>OWL Full</i>	19
2.3 JSON-LD, JSON FOR LINKING DATA	19
2.4 FIHR	20
2.5 R2RML	22
3 ONTOLOGIES AND HEALTHCARE.....	23
3.1 LITERATURE.....	23
3.1.1 <i>Materials and Methods</i>	23
3.1.2 <i>Ontology Search Engines</i>	24
3.2 RESULTS.....	25
4 SELECTED ONTOLOGIES	30
4.1 EXISTING ONTOLOGIES SELECTED FOR THE DEVELOPMENT OF ODIN ONTOLOGY	30
4.1.1 <i>BOT – Building Topology Ontology</i>	30
4.1.2 <i>Organization Ontology</i>	32
4.1.3 <i>NCIT – National Cancer Institute Thesaurus Ontology</i>	33
4.1.4 <i>SNOMED – Systemized Nomenclature of Medicine Ontology</i>	35
4.1.5 <i>ICD9CM – International Classification of Diseases 9</i>	35
4.1.6 <i>CORA – Core Ontology for Automation and Robotics</i>	36
4.1.7 <i>AI – Artificial Intelligence Ontology</i>	37
4.1.8 <i>WoT – Web of Thing Ontology</i>	38
5 ODIN ONTOLOGY.....	39
5.1 ODIN ONTOLOGY DESIGN AND REALIZATION	39
5.1.1 <i>Phase 0: Class Definition</i>	41
5.2 ODINEMDN – EUROPEAN MEDICAL DEVICE NOMENCLATURE SEMANTIC ONTOLOGY.....	53
5.2.1 <i>OdinEMDN Ontology</i>	53

5.2.2	<i>Phase 0: EMDN File Creation</i>	53
5.2.3	<i>Phase 1: Ontology Development</i>	54
5.2.4	<i>Class Definition</i>	56
5.2.5	<i>Object Properties Definition</i>	57
6	COLLECTION OF DATASETS FROM THE PILOTS	58
6.1	DATA IDENTIFICATION	58
6.2	DATA VALUATION	59
6.3	DATA MAPPING	60
6.4	DATA CONNECTION	61
6.4.1	<i>Direct approach: ODIN native</i>	61
6.4.2	<i>Tool Assisted approach</i>	61
6.4.3	<i>Automatization approach</i>	61
6.4.4	<i>Manual approach</i>	61
7	HARMONIZATION OF RESOURCES, ODIN COMMON DATA MODEL	63
7.1	RESOURCE REGISTRATION AND REPRESENTATION	63
7.2	TRANSFORMING DATA TO THE COMMON DATA MODEL	64
7.3	DATA MODEL-RELATED ODIN SERVICES	68
8	MAKING USE OF THE ONTOLOGY	70
8.1	IoT	70
8.1.1	<i>Perception and processing of data</i>	70
8.1.2	<i>Automation</i>	70
8.1.3	<i>Reasoning, Learning and Taking decisions</i>	71
8.2	ROBOTS	72
8.3	AI	74
8.3.1	<i>Uses of ontology for AI</i>	74
8.3.2	<i>Ontology enrichment with AI</i>	74
9	CASE STUDY: HOSPITAL CLINICO SAN CARLOS	75
9.1	DESCRIPTION OF THE PILOT	75
9.2	DATASETS	75
9.2.1	<i>PLANS</i>	76
9.2.2	<i>EQUIPMENT AND CONSUMABLES INVENTORY</i>	78
9.2.3	<i>ORGANIZATION CHART</i>	78
9.3	MAPPING THE PREMISES IN THE ODIN ONTOLOGY	80
9.3.1	<i>Phase 1: Properties Definition</i>	80
9.3.2	<i>Phase 1.1: Object Properties</i>	80
9.3.3	<i>Phase 1.2: Data Properties</i>	83
9.3.4	<i>Phase 2: Individuals Definition</i>	84
9.3.5	<i>Phase 3: Ontology Development</i>	85
9.3.6	<i>CARDIOLOGY DEPARTMENT</i>	86
9.3.7	<i>EMERGENCY ROOM</i>	89

9.4	MAPPING THE ROBOTS AND THEIR TASKS IN THE ODIN ONTOLOGY	92
10	CONCLUSIONS	96
11	REFERENCES	97

List of tables

TABLE 1. DELIVERABLE CONTEXT	10
TABLE 2. RDFS VOCABULARY	14
TABLE 3: OWL LITE SYNOPSIS	16
TABLE 4: OWL DL AND OWL FULL SYNOPSIS	18
TABLE 5. SUMMARY OF THE CHARACTERISTICS THE SELECTED ARTICLES.....	26
TABLE 6 SOME EXAMPLE TYPES OF NON-RELATIONAL DATABASES AND IMPLEMENTATION (SOURCE WIKIPEDIA).....	59
TABLE 7: EXISTING TOOLS FOR AUTOMATIC SEMANTIC TRANSLATION.	66
TABLE 8: TABLE OF ODIN'S ONTOLOGY OBJECT PROPERTY.....	80
TABLE 9: ODIN DATA PROPERTIES	83

List of figures

FIGURE 1: SEMANTIC WEB STACK	12
FIGURE 2. THE RDF TRIPLE	13
FIGURE 3. INTRODUCTION TO RDFS.....	14
FIGURE 4: STRUCTURE OF OWL 2	15
FIGURE 5: OWL VARIANTS	16
FIGURE 6: A JSON-LD DOCUMENT	20
FIGURE 7: FHIR RESOURCE PERSON	20
FIGURE 8: FHIR PERSON RESOURCE, XML TEMPLATE.....	21
FIGURE 9: FHIR JSON ENCODING FOR PERSON RESOURCE	21
FIGURE 10: FHIR TURTLE ENCODING	21
FIGURE 11: R2RML MAPPING LANGUAGE	22
FIGURE 12: R2RML DIRECT MAPPING.....	22
FIGURE 13. FLOW DIAGRAM REPRESENTING THE PROCESS OF SELECTION OF THE INCLUDED STUDIES..	25
FIGURE 14. FOUND ONTOLOGIES	30
FIGURE 15. BOT ONTOLOGY CLASSES.....	30
FIGURE 16. BOT ONTOLOGY OBJECT PROPERTIES.....	31
FIGURE 17. BOT ONTOLOGY - EXAMPLES OF OBJECT PROPERTIES LINKING CLASSES	32
FIGURE 18. BOT ONTOLOGY - EXAMPLE OF OBJECT PROPERTIES FOR THE CLASS BOT:INTERFACE	32
FIGURE 19. ORGANIZATION ONTOLOGY.	33
FIGURE 20. NCIT ONTOLOGY	34
FIGURE 21: SNOMED-CT MAIN TYPES OF COMPONENTS	35
FIGURE 22: ICD9CM DISEASES AND INJURIES CLASS AND SUBCLASSES	36
FIGURE 23: ICD9CM PROCEDURES CLASS AND SUBCLASSES	36
FIGURE 24. CORA ONTOLOGY	37
FIGURE 25: AI ONTOLOGY CLASSES AND SUBCLASSES	37
FIGURE 26: OVERVIEW OF THE WoT ONTOLOGY.....	38
FIGURE 27: ODIN ONTOLOGY OVERVIEW	40
FIGURE 28: ODIN ONTOLOGY CLASSES	42
FIGURE 29: DEVICE CLASS OF THE ODIN ONTOLOGY	43
FIGURE 30: AI ODIN'S ONTOLOGY	44
FIGURE 31: AI ODIN'S ONTOLOGY	45
FIGURE 32: AI ODIN'S ONTOLOGY	46
FIGURE 33: ODIN SITE OF CARE DELIVERY CLASS	47
FIGURE 34: ODIN'S ONTOLOGY ELEMENT CLASS	47
FIGURE 35: UNIT OF MEASURE CLASS OF ODIN ONTOLOGY.....	48
FIGURE 36: ICD9 CLASS OF ODIN ONTOLOGY.....	49
FIGURE 37: OCCUPATION CLASS OF ODIN	51
FIGURE 38: ORGANIZATION CLASS.....	52
FIGURE 39: EMDN MEDICAL DEVICES ALPHANUMERIC STRUCTURE	53
FIGURE 40: CELLFILE PLUGIN IN PROTÉGÉ.....	54

FIGURE 41: TRANSFORMATION RULES	55
FIGURE 42: TRANSFORMATION RULES	55
FIGURE 43: ODIN EMDN ONTOLOGY CLASSES.....	56
FIGURE 44: ODIN EMDN ONTOLOGY CLASSES, ONTOGRAF VIEW	56
FIGURE 44 DATASET COLLECTION ETL METHODOL	58
FIGURE 45: RESOURCE REGISTRATION PROCESS.....	64
FIGURE 46: RESOURCE COMMUNICATION WITHIN THE ODIN ARCHITECTURE.	65
FIGURE 47: EACH CONNECTOR PERFORMS SEMANTIC TRANSLATION AND TRANSPORT TO THE ESB MESSAGE FORMAT.....	65
FIGURE 48: AUTOMATIC SEMANTIC TRANSLATION AND DATA HARMONIZATION PROCESS.	66
FIGURE 51: HAEMODYNAMIC MAP	76
FIGURE 52: EMERGENCY MAP 1	77
FIGURE 53: EMERGENCY MAP 2	77
FIGURE 54: CONSUMABLES AND EQUIPMENT DATA	78
FIGURE 55: CONSUMABLES AND EQUIPMENT DATA	78
FIGURE 56: CARDIOLOGY DEPARTMENT DESCRIPTION	79
FIGURE 57: CARDIOLOGY DEPARTMENT UNITS.....	79
FIGURE 58: HEAD OF HEMODYNAMICS UNIT.....	80
FIGURE 59: SITE OF CARE DELIVERY CLASS INDIVIDUALS	84
FIGURE 60: SITE OF CARE DELIVERY CLASS INSTANCES	85
FIGURE 61: MAP OF HCSC WITH THE PROPERTY HAS BUILDING.....	85
FIGURE 62: STOREYS OF THE BUILDINGS DECLARED WITH THE PROPERTY HAS STOREY.....	86
FIGURE 63: SPACES OF THE HAEMODYNAMICS DEFINED WITH THE PROPERTY HAS SPACE	86
FIGURE 64: ADJACENT SPACES IN HAEMODYNAMICS	87
FIGURE 65: FACING SPACES IN HAEMODYNAMICS	87
FIGURE 66: ELEMENT CONTAINED IN SOME SPACES OF HAEMODYNAMICS.....	87
FIGURE 67: MEDICAL DEVICES LOCATED IN THE CARDIOLOGY DEPARTMENT	88
FIGURE 68: PART OF THE MEDICAL DEVICES IN THE HAEMODYNAMICS UNIT.....	88
FIGURE 69: SPACES OF THE FIRST STOREY OF EMERGENCY	89
FIGURE 70: SPACES OF THE SECOND STOREY OF EMERGENCY	89
FIGURE 71: ADJACENT SPACES IN EMERGENCY.....	90
FIGURE 72: ADJACENT SPACES IN EMERGENCY 2.....	90
FIGURE 73: FACING SPACES IN EMERGENCY.....	91
FIGURE 74: FACING SPACES IN EMERGENCY 2.....	91
FIGURE 75: PART OF THE MEDICAL DEVICES LOCATED IN EMERGENCY	92
FIGURE 76: INTRODUCTION TO DEVICES THAT THE ROBOT CARRIES	93
FIGURE 77: A ROBOT CARRIES DEVICES FROM THE CD STORAGE	93
FIGURE 78: SPACES WHERE THE ROBOT DELIVERS	94
FIGURE 79: TECHNICAL SPECIFICATIONS OF THE ROBOT	94
FIGURE 80: TECHNICAL SPECIFICATIONS OF THE ROBOT CLASS.....	95

1 INTRODUCTION

Ontologies are vocabularies that make explicit the meaning of words and generate, by means of the relations between them, a tree that, as it extends, creates ever finer distinctions. You can structurally represent both general concepts and single semantic fields, or a domain. In both cases there are many points of view from which to analyze a topic. This condition makes us understand that a single tree is not enough to express the concepts in a specific domain. Instead, we need a family of trees that, on the whole, fully describe it. In semantic platforms, classes are concepts belonging to a certain domain, which in turn can be deepened and thus a hierarchy of classes and subclasses is created. Furthermore, each class or subclass is populated by an individual. Finally, the classes are related to each other through properties which express the relationships between concepts. Communication is sharing of knowledge and is effective when a common language is used. The language used for the development of ontologies is the Web Ontology Language (OWL), a standard from the World Wide Web Consortium (W3C). The most effective way to build an ontology is to reuse existing semantic platforms, ensuring data interoperability, and then managing it with software that exploits it, such as Protégé.

The ODIN project intends to establish a semantic platform capable of optimizing the organization of products and services in healthcare contexts. Ontologies offer a wide range of applications. Two ontologies have been created, the ODIN Ontology and the OdinEMDN Ontology. The latter was necessary because, to the best of our knowledge, there was no ontology containing the classification of all medical devices on the market, while ODIN concretizes the semantic description of the objectives that the project arises. This first version (V1) was tested for the pilot SERMAS - San Carlos Clinical Hospital, describing the Hemodynamic and Emergency units, thanks to the documentation provided by the pilot.

1.1 Deliverable context

This deliverable aims to implement the ODIN semantic platform, defining the decisions made in the design phase. The results of this deliverable should be the starting point for future developments of the ODIN Ontology, as the tools and knowledge used to implement ODIN V1 are provided in this document. First, we provide a brief introduction to the programming languages that are typically used to develop an ontology. Then there is a review of the found literature and existing ontologies. The ontologies that have proved important for the creation of the ODIN Ontology are described. Finally, there is a thorough description of the ODIN and OdinEMDN Ontologies, and their mapping on a specific pilot.

Table 1. Deliverable context

PROJECT ITEM IN THE DOA	RELATIONSHIP
Project Objectives	<p>The deliverable is relevant to ODIN’s Objective 1, as it describes and defines the semantic ontology which is the grounding to build the ODIN platform’s data model and to model the interactions between entities.</p> <p>The WP3 objectives are:</p> <ul style="list-style-type: none"> • Semantic models, creating an efficient data model, based on standards and from the most appropriately selected datasets by efficient and effective pre-processing of all available data.

	<ul style="list-style-type: none"> • Knowledge Base, providing the component hold and populate it with the standard knowledge needed and collected in the hospital environment. • Data interpretation, providing an inference engine which could be capable of interpreting semantic rules through the RIF or SWRL standards. • Integration of the platform, ensuring all the components in the platform integrate correctly in a single execution context and that this execution context is easy to deploy in pilots. • Implement Privacy, security, and trust mechanisms in the platform • Ensure Platform documentation provides the necessary information and at the appropriate level • Provide any user or tool support 		
Exploitable results	<p>There is no specific contribution to any exploitable results. Instead, the infrastructure presented hereby will be used as the basis for the development of potentially exploitable components.</p>		
Workplan	<p>D3.2 is attributed to the task T3.2 of WP3, Platform Semantic Ontology and Harmonized Datasets</p>		
Milestones	<p>The D3.2 is a key deliverable for the PREPARATION (MS1) and IMPLEMENTATION (MS3) milestones of the project.</p>		
Deliverables	D3.4 – D3.6	Privacy Security and Trust report	Regarding security mechanisms
	D3.7 – D3.9	Technical Support Plan and Operations	Regarding component documentation and feedback collection.
	D3.10 – D3.12	ODIN platform	Regarding the application of DevOps in the development of the ODIN platform.
Risks	<p>The ontology provided in this deliverable can help in minimizing the following risks identified in the Grant Agreement:</p> <ul style="list-style-type: none"> • Technologies not available in time • Technical problems during component/module development • Complexity of unification procedure 		

2 SEMANTIC ONTOLOGIES

A semantic ontology is a knowledge representation tool based on formal collections of terms. It is used to describe and represent an area of interest, i.e. a domain, in a unified and non-ambiguous way. Ontologies lay the basis of the Semantic Web, an extension of the World Wide Web, set by the World Wide Web Consortium (W3C)³ with the main goal of enabling computers to support interactions over the network. The Semantic Web, Figure 1, is a web of data, which means that, through several technologies, it provides an environment where it is possible to query data and draw inferences using ontologies. Ontologies and vocabularies are therefore used in a multitude of applications to help with the integration of data derived from different providers or to organize knowledge in a formal way, linking the terms through logical relations. Ontologies make it also possible to draw inferences, i.e. automatic procedures that generate new relationships based on the information stored in the vocabulary itself, in order to perform reasoning procedures. The structure of an ontology is hierarchical, and it is based on methods that classify the contained items in “classes” and “subclasses”. In doing so individual resources can therefore be associated with them, creating logical relationships between classes and instances. Given the wide range of operations provided by ontologies, the W3C offers a set of several techniques to define different forms of vocabularies in a standard way, that includes RDF, RDFS and Web Ontology⁴. The Web Ontology Language (OWL) is a logic-based language designed to represent complex knowledge and relationships between items and is the format used to define ontologies⁵, while the Resource Description Framework (RDF) is the web’s standard model for the exchange of data. The semantic ontology technology has become more and more crucial for the performance of several applications because of its characteristics of being able to provide a common representation of a domain among different users and a defined semantics that links concepts and instances, to support interoperability between heterogeneous data archives and to favour the reuse and sharing of knowledge [1].

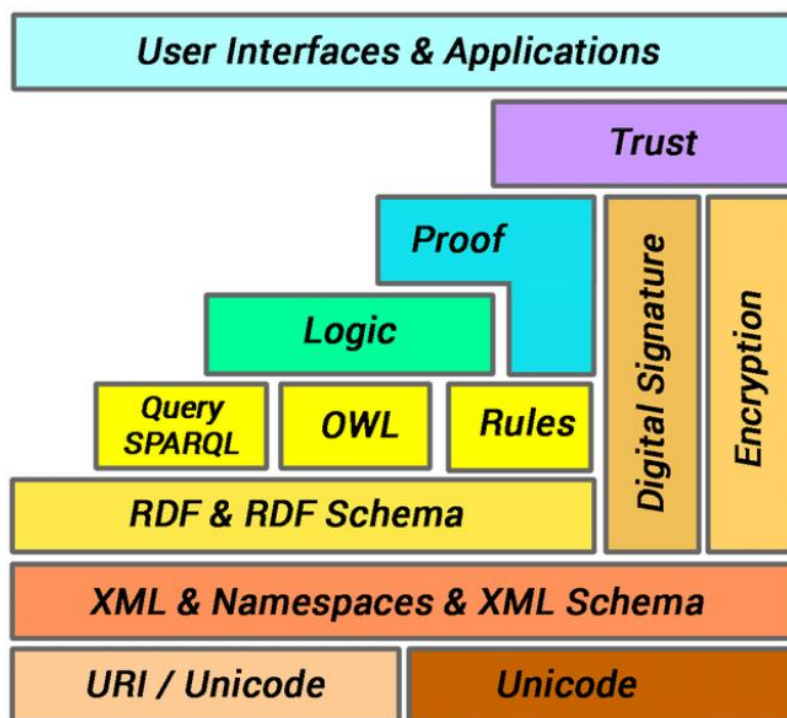


Figure 1: Semantic Web stack

2.1 RDF – RESOURCE DESCRIPTION FRAMEWORK

Interoperability is defined as the efficient flow of data and services between hardware and software systems, allowing communication between them. On a semantic level, when you browse the web, you follow links that lead to a resource with a Uniform Resource Identifier-URI that uniquely identifies it. This resource is also referred to as a document or object to emphasize that it can be read by both humans and machines. Metadata refers to the information about the asset. The crucial point is that metadata is machine-understandable, and its use necessitates rules for semantics, which provides explicit meaning, syntax, which allows information to be exchanged across programs, and structure, which is a formal syntactic constraint.

The Resource Description Framework-RDF is one of the most important recommendations of the W3C that enables the exchange and usage of metadata, as well as compatibility between apps that communicate machine-readable data. As a result, it does not specify semantics, but it does provide a standard framework for expressing them.

The RDF model is based on 3 concepts:

- Resource: is all that is described, is identified with the IRI;
- Properties: it is an attribute that you want to assign to a resource.
- Statement: is the association of the property to the resource, its structure is a triple subject / predicate / object, as showed in Figure 2.

Subject, predicate, and object constitute a semantic triple. It is possible to use a predicate to connect an object (subject) to another object (object) or a literal [2] .

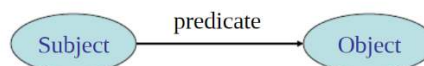


Figure 2. The RDF triple.

Inference is a core part, as it is a technique for deducing or discovering new data from existing triples. It is not necessary to store all the relationships; thus, new facts and relationships can be formed from existing triples. RDF is composed by 2 parts:

- RDF Model and Syntax
- RDF Schema-RDFS

RDF Schema is a simple vocabulary description language, defines the vocabulary used in RDF data models and extends RDF as shows table 2. Figure 3 is an example of how the RDFS vocabulary can be used to describe something.

Table 2. RDFS vocabulary.

rdfs:Resource	Class of resources.
rdfs:Class	Class of all classes.
rdfs:Literal	Class of strings.
rdfs:Property	Class of all properties.
rdfs:Datatype	Class of datatypes.
rdfs:subClassOf	Relates class to one of its super classes, declare hierarchies of classes, is transitive.
rdfs:subPropertyOf	Relates a property to one of its super properties, is transitive by definition.
rdfs:domain	Declares the class of the subject in a triple.
rdfs:range	Declares the class or datatype of the object in a triple.
rdfs:comment	Typically provides a longer text description of the resource.
rdfs:label	Associates the resource with a human-friendly name.
rdfs:isDefinedBy	Relates a resource to the place where its definition, typically an RDF schema.
rdfs:seeAlso	Relates a resource to another resource that explains it.

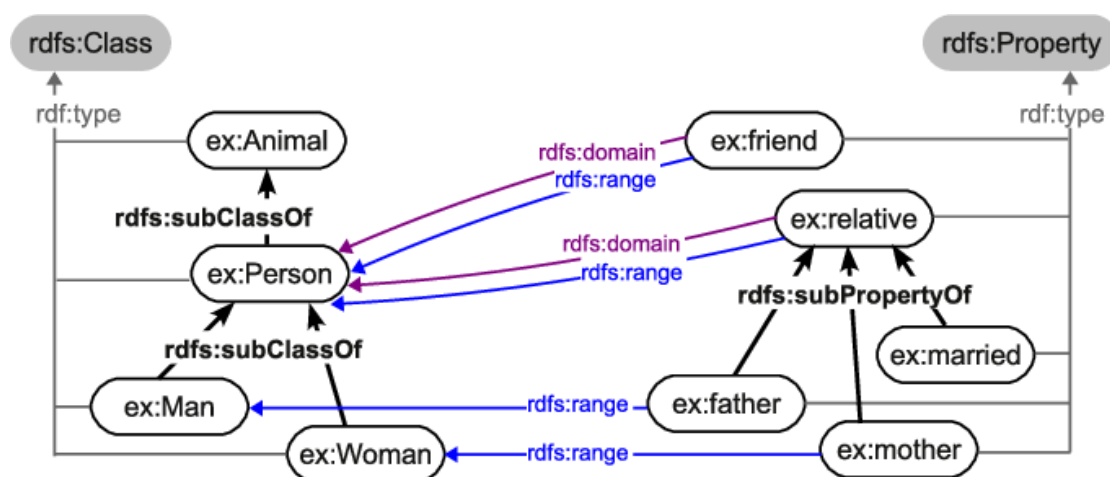


Figure 3. Introduction to RDFS.

2.2 OWL – ONTOLOGY WEB LANGUAGE

The OWL Web Ontology Language is based on description logic, it is a W3C recommendation since 2004, and it extends the semantic expressiveness of RDF. The new version is called OWL 2 and it is an extension of OWL, a W3C recommendation since 2009. An OWL ontology consists of classes/properties/individuals [3]:

- **Class:** A class is made up of instances that have properties in common. It is possible to create a hierarchy of classes with the `subClassOf` construct, thus specifying that one class is a subclass of another which will be the superclass. There are two predefined classes, the *Thing* class, which represents the superclass of all classes and, also, the class of all individuals, the *Nothing* class which is an empty class. The OWL classes are comparable with classes in RDF.
- **Properties:** Properties are relations that link two objects. The objects can be instances of two classes, in the case of an *Object property*, or instances of a class and a data value, in the case of a *Datatype property*. When defining any type of property, it is also necessary to specify the Domain and Range of the property, that establish which instances the property can be applied to and which instances can have it as a value. The OWL properties are comparable with properties in RDF.
- **Individuals:** In OWL, instances of a class are referred to as Individuals. As described above, different individuals can be linked to each other through properties. The OWL individuals are comparable with objects in RDF.

OWL 2 can be represented in different syntax variants which are shown below in Figure 4.

The following subsections will describe the 3 sublanguages of OWL, Figure 5, which are OWL Lite, OWL DL, OWL Full.

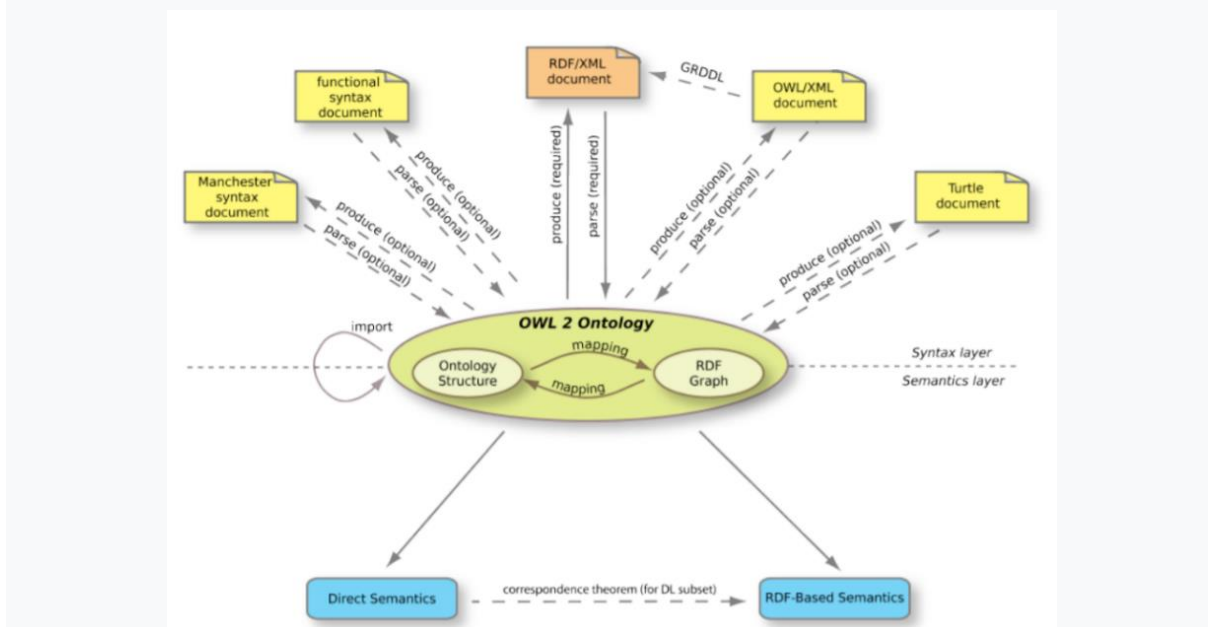


Figure 4: Structure of OWL 2

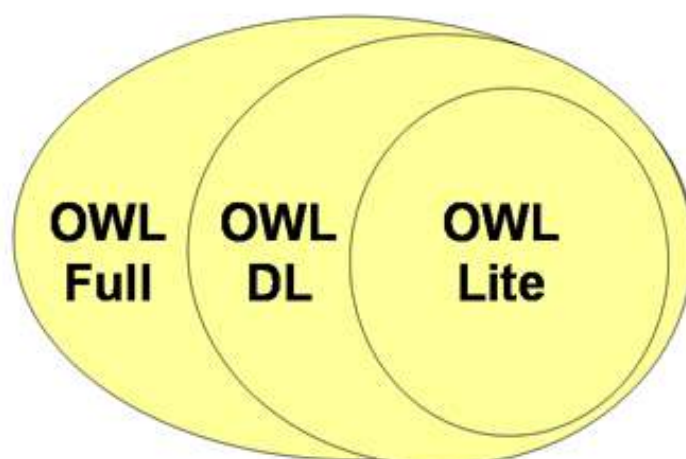


Figure 5: OWL variants

2.2.1 OWL Lite

OWL Lite can be seen as an extension of a portion of RDF, this version of OWL has more restrictions than the DL and Full version, it's easy to implement but has a low expressive power compared to the other versions [4].

The list of OWL Lite language constructs is shown in Table 3.

Table 3: OWL Lite synopsis

RDF Schema	Class	
	rdf:Property rdfs:subClassOf rdfs:subPropertyOf rdfs:domain rdfs:range owl:Individual	Rdf:Property and rdfs:subPropertyOf creates properties hierarchies. Rdfs:domain and rdfs:range explicits those individuals who can be applied a property and who can have a property as value.
Equivalent/Different	owl:equivalentClass owl:equivalentProperty owl:sameAs owl:differentFrom owl:allDifferent	Two classes are equivalent if they have the same individuals, two properties are equivalent if they link one individual to the same group of individuals. The properties sameAs, differentFrom, allDifferent, refers to individuals.
	Properties	owl:inverseOf owl:transitiveProperty owl:symmetricProperty

	<p>owl:functionalProperty owl:inverseFunctionalProperty</p>	<p>A transitive property is a property that can be deduced from the reasoner, for example if the couples (Michael, David) and (David, Clara) are ancestors, then for the reasoner (Michael, Clara) are ancestors too.</p> <p>If a property is a functional property, then each individual should have a unique value.</p>
Restrictions on the type of property	<p>owl:allValuesFrom owl:someValuesFrom</p>	<p>This is a range restriction for the instances of a class. If a class has as restriction property allValuesFrom another class, and if an instance of the first class is connected to another instance of a second class, then the first individual represents a range restriction for the second one.</p> <p>Rather with someValuesFrom it is needed that there's at least one instance of the property that belongs to the class, thus it doesn't limit all the property values as in the case of allValuesFrom.</p>
Cardinality restrictions	<p>owl:minCardinality owl:maxCardinality owl:cardinality</p>	<p>If a class has 1 min or max cardinality, then all the instances of this class must be linked to one min 1 or max 1 individual of the class.</p> <p>OWL Lite defines "at least one", "not more than one", "exactly one" values, then OWL DL expands the properties also to positive integer values other than 0 or 1.</p>
Intersection of Classes	<p>owl:intersectionOf</p>	<p>Intersection between classes and restrictions.</p>
Annotation properties	<p>rdfs:label rdfs:comment rdfs:seeAlso rdfs:isDefinedBy</p>	<p>This annotation properties allows to better define classes, properties and individuals.</p>
Datatypes	<p>datatypeProperty</p>	<p>A relation between two instances of a class and an RDF or XML data value.</p>

2.2.2 OWL DL

OWL DL is the version of OWL that provides maximum expressiveness. It extends the vocabulary of OWL Lite includes all the linguistic constructs of OWL Full but has more restrictions than this one.

The list of OWL DL language constructs is shown in Table 4.

Table 4: OWL DL and OWL Full synopsis

		A class can be described by listing all of its instances, thanks to one of. With dataRange the list is defined by literals.
Class Axioms	owl:oneOf	If two classes are disjoint, an instance of the first class cannot be an instance of the second class as well.
	owl:dataRange	
	owl:disjointWith	
	owl:equivalentClass	Two equivalent classes have the same extensions and therefore the same individuals.
	rdfs:subClassOf	With subClassOf can be defined the restrictions, characteristics, that the class must inherit.
Boolean Combinations	owl:unionOf owl:complementOf	Can represent Boolean combinations

of Class Expressions	owl:intersectionOf	of class and restrictions.
Arbitrary Cardinality	owl:minCardinality owl:maxCardinality owl:cardinality	Cardinality extended to positive integer values also different from 0 and 1.
Filler Information	owl:hasValue	Property restriction.

2.2.3 OWL Full

OWL Full is the version of OWL that guarantees maximum expressiveness without any computational guarantee, i.e., the inference system may not deduce all the conclusions. This version uses the same language constructs as OWL DL, Table 4, without any restrictions.

2.3 JSON-LD, JSON FOR LINKING DATA

JSON-LD or JavaScript Object Notation for Linked Data is a Linked Data serialization recommended by the W3C. Is an extension for the JSON format that integrates Linked Data to a website and also an RDF serialization format providing context to data. Adds the 'name: value' pairs of the JSON annotation using the keywords introduced by the '@' symbol. The keywords '@context' and '@type' are particularly important. The standard for structuring data is given by schema.org. Figure 6 provides an example of a JSON-LD document [5].

```

{
  "@context": {
    "name": "http://xmlns.com/foaf/0.1/name",
    "knows": "http://xmlns.com/foaf/0.1/knows"
  },
  "@id": "http://me.markus-lanthaler.com/",
  "name": "Markus Lanthaler",
  "knows": [
    {
      "@id": "http://manu.sporny.org/about#manu",
      "name": "Manu Sporny"
    }, {
      "name": "Dave Longley"
    }
  ]
}

```

Figure 6: A JSON-LD document

2.4 FHIR

HL7 FHIR, Fast Healthcare Interoperability Resources, is a standard for exchanging electronic healthcare information allowing request and transfer data between various healthcare systems. The building blocks of FHIR are called resources, in Figure 7 there is an example of the resource Person.

The main goal of FHIR is to solve a wide range of clinical and administrative healthcare problems to improve interoperability, it can be expressed as XML (Figure 8), JSON (Figure 9) or RDF/TURTLE (Figure 10) encodings [6].

Structure

Name	Flags	Card.	Type	Description & Constraints
Person	TU		DomainResource	A generic person record Elements defined in Ancestors: id, meta, implicit(Rules, language, text, contained, extension, modifierExtension)
Identifier		0..*	Identifier	A human identifier for this person
name	Σ	0..*	HumanName	A name associated with the person
telecom	Σ	0..*	ContactPoint	A contact detail for the person
gender	Σ	0..1	code	male female other unknown AdministrativeGender (Required)
birthDate	Σ	0..1	date	The date on which the person was born
address		0..*	Address	One or more addresses for the person
photo		0..1	Attachment	Image of the person
managingOrganization	Σ	0..1	Reference(Organization)	The organization that is the custodian of the person record
active	? Σ	0..1	boolean	This person's record is in active use
link		0..*	BackboneElement	Link to a resource that concerns the same actual person
target		1..1	Reference(Patient Practitioner RelatedPerson Person)	The resource to which this actual person is associated
assurance		0..1	code	level1 level2 level3 level4 IdentityAssuranceLevel (Required)

Figure 7: FHIR resource Person

```

<Person xmlns="http://hl7.org/fhir">
  <!-- from Resource: id, meta, implicitRules, and language -->
  <!-- from DomainResource: text, contained, extension, and modifierExtension -->
  <identifier><!-- 0..* Identifier A human identifier for this person --></identifier>
  <name><!-- 0..* HumanName A name associated with the person --></name>
  <telecom><!-- 0..* ContactPoint A contact detail for the person --></telecom>
  <gender value="[code]"/><!-- 0..1 male | female | other | unknown -->
  <birthDate value="[date]"/><!-- 0..1 The date on which the person was born -->
  <address><!-- 0..* Address One or more addresses for the person --></address>
  <photo><!-- 0..1 Attachment Image of the person --></photo>
  <managingOrganization><!-- 0..1 Reference(Organization) The organization that is the custodian of the pe
  rson record --></managingOrganization>
  <active value="[boolean]"/><!-- 0..1 This person's record is in active use -->
  <link> <!-- 0..* Link to a resource that concerns the same actual person -->
  <target><!-- 1..1 Reference(Patient|Practitioner|RelatedPerson|Person) The resource to which this actua
  l person is associated --></target>
  <assurance value="[code]"/><!-- 0..1 level1 | level2 | level3 | level4 -->
</link>
</Person>

```

Figure 8: FHIR Person resource, XML template

```

{
  "resourceType": "Person",
  // from Resource: id, meta, implicitRules, and language
  // from DomainResource: text, contained, extension, and modifierExtension
  "identifier": [{ Identifier }], // A human identifier for this person
  "name": [{ HumanName }], // A name associated with the person
  "telecom": [{ ContactPoint }], // A contact detail for the person
  "gender": "<code>", // male | female | other | unknown
  "birthDate": "<date>", // The date on which the person was born
  "address": [{ Address }], // One or more addresses for the person
  "photo": { Attachment }, // Image of the person
  "managingOrganization": { Reference(Organization) }, // The organization that is the custodian of the
  person record
  "active": <boolean>, // This person's record is in active use
  "link": [{ // Link to a resource that concerns the same actual person
    "target": { Reference(Patient|Practitioner|RelatedPerson|Person) }, // R! The resource to which thi
    s actual person is associated
    "assurance": "<code>" // level1 | level2 | level3 | level4
  }]
}

```

Figure 9: FHIR JSON encoding for Person resource

```

@prefix fhir: <http://hl7.org/fhir/> .

[ a fhir:Person;
  fhir:nodeRole fhir:treeRoot; # if this is the parser root

  # from Resource: .id, .meta, .implicitRules, and .language
  # from DomainResource: .text, .contained, .extension, and .modifierExtension
  fhir:Person.identifier [ Identifier ], ... ; # 0..* A human identifier for this person
  fhir:Person.name [ HumanName ], ... ; # 0..* A name associated with the person
  fhir:Person.telecom [ ContactPoint ], ... ; # 0..* A contact detail for the person
  fhir:Person.gender [ code ]; # 0..1 male | female | other | unknown
  fhir:Person.birthDate [ date ]; # 0..1 The date on which the person was born
  fhir:Person.address [ Address ], ... ; # 0..* One or more addresses for the person
  fhir:Person.photo [ Attachment ]; # 0..1 Image of the person
  fhir:Person.managingOrganization [ Reference(Organization) ]; # 0..1 The organization that is the custo
  dian of the person record
  fhir:Person.active [ boolean ]; # 0..1 This person's record is in active use
  fhir:Person.link [ # 0..* Link to a resource that concerns the same actual person
    fhir:Person.link.target [ Reference(Patient|Practitioner|RelatedPerson|Person) ]; # 1..1 The resource
    to which this actual person is associated
    fhir:Person.link.assurance [ code ]; # 0..1 level1 | level2 | level3 | level4
  ], ...;
]

```

Figure 10: FHIR Turtle encoding

2.5 R2RML

R2RML is a construction of knowledge graph from heterogeneous Data Sources, usually in non-RDF data formats such as relational databases and rely on the graphic mapping rules of R2RML to generate the RDF. R2RML is a W3C recommendation, a mapping language that normally has as *input* a Database (schema or data) but can also have normally target ontologies and mappings between the database and target ontologies in R2ML, and as *output* an RDF graph that will be available for loading it into a SPARQL endpoint [7]. There are many different mapping alternatives.

Referring to R2RML, Figure 11, basically we talk about triples maps and those ones are going to refer to logical tables or views or a valid SQL query, to get data from the input database. Then every logical table is mapped in RDF with rules to generate Subject maps, Predicate maps, and Object maps, and then the combination of this elements produces the triple. Figure 12 shows a direct mapping as R2RML.

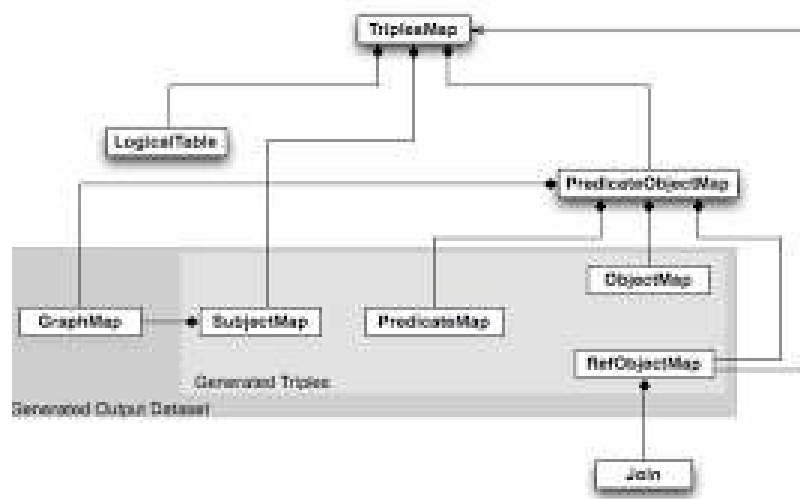


Figure 11: R2RML mapping language

```

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix ex: <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@base <http://example.com/base/> .

<TriplesMap1>
  a rr:TriplesMap;

  rr:logicalTable [ rr:tableName "\"Student\""; ];

  rr:subjectMap [ rr:template "http://example.com/student/{\"ID\"}"; ];

  rr:predicateObjectMap
  [
    rr:predicate    ex:firstName ;
    rr:objectMap    [ rr:column "\"FirstName\""; ]
  ];

  rr:predicateObjectMap
  [
    rr:predicate    ex:lastName ;
    rr:objectMap    [ rr:column "\"LastName\""; ]
  ]

```

Figure 12: R2RML direct mapping

3 ONTOLOGIES AND HEALTHCARE

One of the main goals in the Odin project is to be able to describe infrastructures and facilities, locate medical devices and obtain information about their status, to enable a rational and fast management of the resources present within a clinical process. A semantic ontology realizes the expected results from an organizational and management point of view. The objective of this section is to conduct a detailed analysis of the literature with the aim of knowing the ontologies useful for the Odin platform. A fundamental aspect when a new ontology is developed, is to ensure cooperation and exchange of information at the semantic level. For this purpose, it is therefore important to reuse existing ontologies.

3.1 LITERATURE

The research has been carried out through specific search engines both for the literature review, and for finding ontologies. In the next subsections, the literature search methods and sources are described, emphasizing the rationale behind the choices made.

3.1.1 Materials and Methods

The tools adopted for literature research will be explained in the following subsections which also define the working method adopted for the research and each choice criterion that has been used to select the most relevant articles for the mission of the Odin project.

3.1.1.1 Information Source

The literature search was carried out through the Scopus⁴ database. Scopus is developed by the publishing company Elsevier, and it collects documents belonging to a wide range of subject areas, including several fields regarding the medical and computer science fields, areas of interest in our study. Such characteristic of the database at issue was the reason why it has been chosen for our review.

3.1.1.2 Eligibility Criteria

To be included in this review, the found articles needed to be relevant to the mission and the aim of the ODIN project. For this reason, the search was mainly directed towards articles that focus on the topics expressed by the Use Cases of the project, such as robotics, Internet of Things, healthcare and hospital environment, logistics management, health workers, data collection and disaster preparedness and management. Other eligibility criteria were journal articles written after the year 2000 and in the English or Italian language.

3.1.1.3 Search

The Scopus search was performed using keywords carefully selected after the investigation of the Use Cases. Specifically: *IoT, IoT Healthcare, Drugs Robotics, Emergency, Disaster, Clinical Workflow, Surgery, Logistics, Data Collection, Staff, and Medical record*. All the above followed by the words “*semantic ontology*”. The search was limited to documents produced after the year 2000, written in English and in Italian. It only included documents that are either articles or reviews. The subject areas were the following: *computer science, engineering, medicine, social sciences, decision sciences, multidisciplinary, business, management and accounting, health professions, environmental science, pharmacology toxicology and nursing*. This means that one of the main elimination criteria was to select and exclude all those articles that regarded ontologies that could not be found publicly or that represented a domain beside the point of the project. No grey literature was searched or included.

3.1.2 Ontology Search Engines

Given the aim of this research, which is gathering information and material semantic representation of healthcare structure, the search has not only involved the literature area, but also the ontology domain. The literature search covered made it possible to identify and select a set of ontologies that could provide a semantic representation of the main topics and needs expressed by the Use Cases of the project. The search for such ontologies was carried through more than one method of search. The first method has been to analyse the collection of articles found through literature search with the intention to identify in the content of such documents those ontologies made public online by the creators. That was possible because some of the major ontologies are used in more than one field, therefore it has been possible to select articles describing them and their main applications. This method, even though it was found straightforward and easy, involved a struggle, meaning that most of the articles that were found running the literature search through the aforementioned databases cite and describe ontologies that were developed locally and were not released publicly or ontologies that are now outdated, therefore not relevant to our study. The second employed method was made possible thanks to the analysis of the found articles that, through citation, led to the discovery of very useful and convenient tools that allowed us to directly identify suitable biomedical ontologies. These tools are the following two databases: **Ontobee**⁵ and **BioPortal**⁶. Examples of references in the literature that led to the discovery of said tools can be found in the review's sources, such as [8] and [9], that employ the database Ontobee, or [10], that mentions BioPortal instead. The eligibility criteria employed are the following: the ontologies selected regarded main topics expressed by the ODIN project's Use Cases, therefore mostly biomedical ontologies and those concerning the aspects describing the generic hospital environment. Furthermore, the found ontologies had to fulfil an additional requirement to be selected, which was to be represented and developed in one of the most common markup schemes: either OWL, RDF or RDFS. A brief explanation of the two databases is given in the following subsections.

3.1.2.1 Ontobee

Ontobee is a linked data server designed for ontologies [11] that provides the query, visualization and comparison of different ontologies and ontology terms. It represents the default server for biomedical ontologies in the Open Biological Ontology (OBO) Foundry⁷, a group of researchers that aim at establishing a set of principles to follow when developing ontologies for the biological sciences. This means that the majority of the ontologies that are published on Ontobee are those developed by the OBO Foundry itself: 131 ontologies out of the 180 total. The default ontology format used in Ontobee is OWL. It was first released in 2008 and through the years it has evolved, making it possible to easily access the stored ontology [12]. It features multiple query methods, and among them, the most used for this research were both the search via keyword and the direct selection of the desired item from the list of all the available ontologies provided by the server. Once the ontology is selected, Ontobee amongst the accessible information provides the ontology's Internationalized Resource Identifiers (IRI), a description and the OWL file to download. These are the pieces of information that we collected to conduct our research.

3.1.2.2 BioPortal

BioPortal is an open repository of biomedical ontologies delivered by the National Centre for Biomedical Ontology (NCBO)⁸, which was formed as part of the National Centers for Biomedical Computing network founded by the National Institutes of Health (NIH). The goal of NCBO is to support biomedical researchers by providing online tools such as BioPortal, which contains ontologies concerning fields such as anatomy, chemistry and health [13]. The biomedical ontologies that are available on this particular database may be represented in OWL, RDF, OBO

format or the Protégé⁹ frame language [14] and, although it does not directly provide the ontologies' Uniform Resource Identifiers (URIs), it allows to download the file when a license is not needed. A feature of the website that has proven itself useful to the conduct of this research, is the possibility to visualize the hierarchical structure of the ontology directly on the website, without the need to use a platform like Protégé, making it very immediate to browse through the ontology's classes and entities.

3.2 RESULTS

The literature search led to a total of 1523 articles. The process of selection of the literature material that was included in this review is shown in the flow diagram represented in Figure 13. This graphical representation indicates that out of the items obtained from the first search, only 110 have been considered relevant after reading the title. The abstracts of the articles belonging to this set of items were then read and therefore an additional screening took place, which resulted in the exclusion of 32 items. Finally, after reading the full text of the remaining articles, 22 documents were selected to be included in the review. All the articles included in the review were published between the years 2005 and 2021 and written in English. This conclusive collection contained items that concerned the development, the description and/or the deployment of some of the selected ontologies that regard the themes expressed by the project's Use Cases.

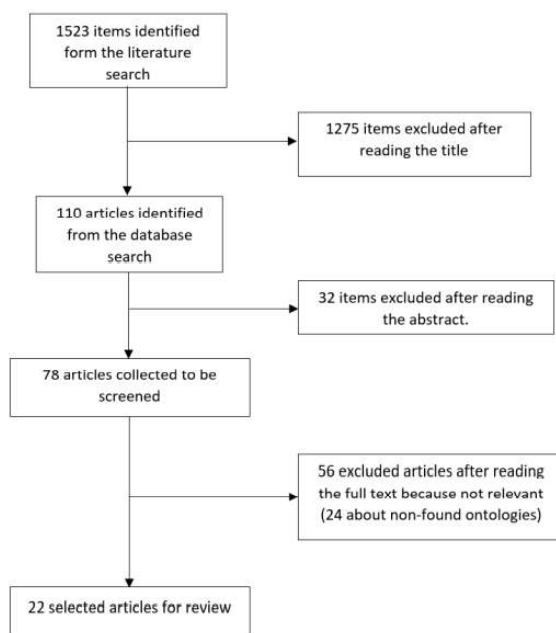


Figure 13. Flow diagram representing the process of selection of the included studies

The main characteristics of the documents selected for this review are displayed in Table 5. The columns of such table are arranged in the following order: the first column presents the first author mentioned in the article, the year of publication and the bibliography reference, the subsequent columns indicate, respectively, the title, the aim of the article under discussion, the ontology mentioned and finally the semantic area around which revolves the scope of the document. As it

is shown in Table 5, the most recent item of the collection of articles that meet our search criteria is [15], being written in 2021, and the least recent are [16] and [17], produced respectively in 2005 and 2006. The remaining items were all produced between the years 2008 and 2020.

Table 5. Summary of the characteristics the selected articles

F. Author				
Year	Title	Aim of the Article	Ontology	Semantic Area
Ref.				
Babcock 2021 [8]	The Infectious Disease Ontology in the age of COVID-19.	Description of IDO and its extensions integrated with the analysis of COVID-19 data: VIDO, CIDO and IDO-COVID-19.	CIDO IDO IDO-COVID-19 VIDO	Disease Vocabulary
Ceusters 2005 [16]	A Terminological and Ontological Analysis of the NCI Thesaurus.	Analysis of the NCI Thesaurus: how the ontological features of the system work together with its terminological parts.	NCIT	Medical Vocabulary
Elsaleh 2015 [18]	IoT-Lite Ontology.	Presentation of IoT-Lite, a lightweight instantiation of the semantic sensor network (SSN) ontology to describe the key IoT concepts that allows interoperability and discovery of sensory data in heterogeneous IoT platforms.	IoT-Lite	Technology
Ei-Sappagh 2018 [19]	SNOMED CT standard ontology based on the ontology for general medical science.	Development of an upper-level ontology to be used as the basis for defining the terms in SNOMED CT.	SNOMEDCT SPM	Medical Vocabulary
Gibaud 2018 [20]	Toward a standard ontology of surgical process models.	Presentation of the OntoSPM Collaborative Action, which serves as a platform developing ontologies in the domain of surgery, focusing on surgical process modelling in the context of Surgical Data Science.	Onto-SPM	Medical Procedure

Glockner 2017 [21]	Lose ODP - an ontology design pattern for logistics services.	Presentation of an ontology design pattern for logistics services.	LoSe ODP	Services
Hakimi 2020 [22]	The Devices, Experimental Scaffolds, and Biomaterials Ontology (DEB): A Tool for Mapping, Annotation, and Analysis of Biomaterials Data.	Description of DEB, an open resource for organizing information about biomaterials, their design, manufacture and biological testing.	DEB	Medical Vocabulary
Hanna 2013 [9]	Building a drug ontology based on RxNorm and other sources.	Description of the building and the structure of the Drug Ontology.	DrOn RxNorm	Drugs Vocabulary
He 2014 [23]	OAE: The Ontology of Adverse Events.	Design of OAE to address adverse events by providing logically well-formed definitions and an associated structured classification.	OAE	Medical Vocabulary
Hicks 2016 [24]	The ontology of medically related social entities: Recent developments.	Description of OMRSE and its recent developments.	OMRSE	Human Role
Ison 2013 [25]	EDAM: An ontology of bioinformatics operations, types of data and identifiers, topics and formats.	Presenting EDAM, an ontology of bioinformatics operations, types of data and identifiers, data formats and topics with the goal of creating machine-understandable annotations for use within resource catalogues.	EDAM	Medical Data
Janowicz 2020 [26]	BOT: The building topology ontology of the W3C linked building data group.	Introduction of BOT, that provides a high-level description of the topology of buildings including storeys and spaces, the building elements they contain, and their web-friendly 3D models.	BOT	Buildings

Lin 2020 [27]	CTO: A community-based clinical trial ontology and its applications in PubChemRDF and SCAIView.	Development of a clinical trial ontology with the goal of aligning and expanding terminologies used by clinical trial registries.	CTO	Medical Vocabulary
Maitra 2021 [15]	Using Ethnographic Methods to Classify the Human Experience in Medicine: A Case Study of the Presence Ontology.	Creation of a conceptual framework to describe and classify data about presence, the domain of interpersonal connection in medicine.	PREO	Human Role
McMurray 2015 [28]	Ontological modelling of electronic health information exchange.	Description of the conceptual framework of health system information exchange and its related ontology.	HEIO	Medical Data
Moreno-Conde 2019 [29]	ITEMAS ontology for healthcare technology innovation.	Definition of a formal representation to specify the most relevant concepts associated with HTI in the Spanish healthcare system.	ITEMAS	Technology
Prestes 2013 [30]	Towards a core ontology for robotics and automation.	Presentation of the current results of the newly formed IEEE-RAS Working Group, named Ontologies for Robotics and Automation and introduction of a core ontology that encompasses a set of terms commonly used in Robotics and Automation.	CORA	Technology
Rahman 2020 [31]	A light-weight dynamic ontology for Internet of Things using machine learning technique.	Proposal of a dynamic ontology to achieve semantic interoperability among heterogeneous devices and applications.	OneM ² M IoT-Lite SSN	Technology

Robinson 2008 [32]	The Human Phenotype Ontology: A Tool for Annotating and Analysing Human Hereditary Disease.	Development of HPO with the goal of covering all phenotypic abnormalities that are commonly encountered in human diseases.	HPO	Disease Vocabulary
Robinson 2010 [33]	The Human Phenotype Ontology.	Development and description of HPO to capture phenotypic information.	HPO	Disease Vocabulary
Santana 2016 [34]	The ontology for the telehealth domain – TEON.	Description and Presentation of TEON, elucidating its main use-case, its applicability and potential to improve information exchange, interoperability and decision support.	TEON	Technology
Zeng 2006 [35]	Exploring and developing consumer health vocabularies.	Presentation of the point of view that CHV development is practical and necessary for extending research on informatics-based tools to facilitate consumer health information seeking, retrieval, and understanding.	OCHV	Medical Vocabulary

4 SELECTED ONTOLOGIES

Following the analysis carried out as described in the previous, 97 ontologies have been selected, among these 52 are potentially useful for the objectives of the Odin project. But after an even more detailed analysis we have concluded that the ontologies that contribute to make more understandable and achievable the Odin ontology, are those described in the following subsections. Figure 14 shows what has been expressed.

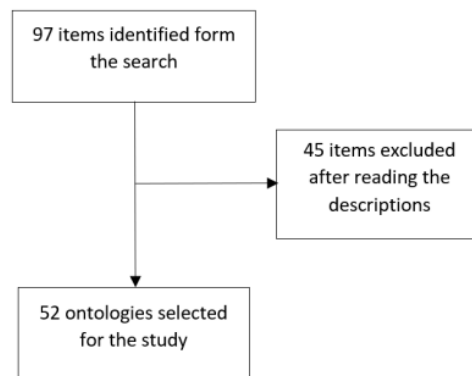


Figure 14. Found Ontologies

4.1 EXISTING ONTOLOGIES SELECTED FOR THE DEVELOPMENT OF ODIN ONTOLOGY

The following subsections provide a summary of the most relevant ontologies out of the selected ones.

4.1.1 BOT – Building Topology Ontology

BOT originated from the need for the implementation of web-based applications to enhance the Building Information Modelling (BIM) methods. The Building Topology Ontology defines the relationships between the components of a building and is used in the construction industry to promote the integration of linked data in the design, planning, constructing, and maintaining a building. The classes of the ontology are the ones showed in Figure 15:

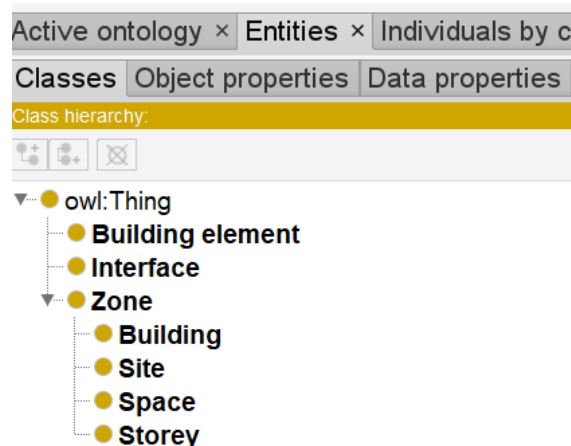


Figure 15. BOT Ontology classes

A bot:Zone is “A part of the physical world or a virtual world that is inherently both located in this world and has a 3D spatial extent; Sub-classes of bot:Zone include bot:Site (A part of the physical world or a virtual world that is inherently both located in this world and having a 3D spatial extent. It is intended to contain or contains one or more buildings.), bot:Building (An independent unit of the built environment with a characteristic spatial structure, intended to serve at least one function or user activity [ISO 12006-2:2013]) , bot:Storey (A part of the physical world or a virtual world that is inherently both located in this world and having a 3D spatial extent.), or bot:Space (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.)”¹⁰. The class Zone is the main class of the BOT ontology.

Moreover, the BOT also includes some important object properties indicated in Figure 16.

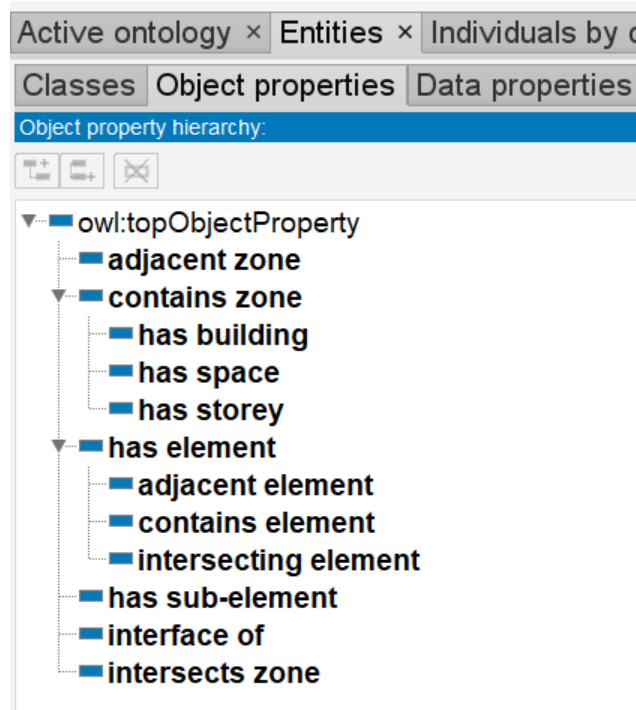


Figure 16. BOT Ontology object properties

Regarding the Odin ontology, the Building Topology Ontology has been selected for the need to clearly define the organization of the spaces and map the hospital structure, for example the San Carlos Clinical Hospital (SERMAS). Linking the classes and the object properties of the BOT, as shown in Figure 17 and Figure 18, it is possible to create a map, at a semantic level, of the building. This is a significant aspect for Odin’s objectives since we may declare the medical equipment that are present in a room.

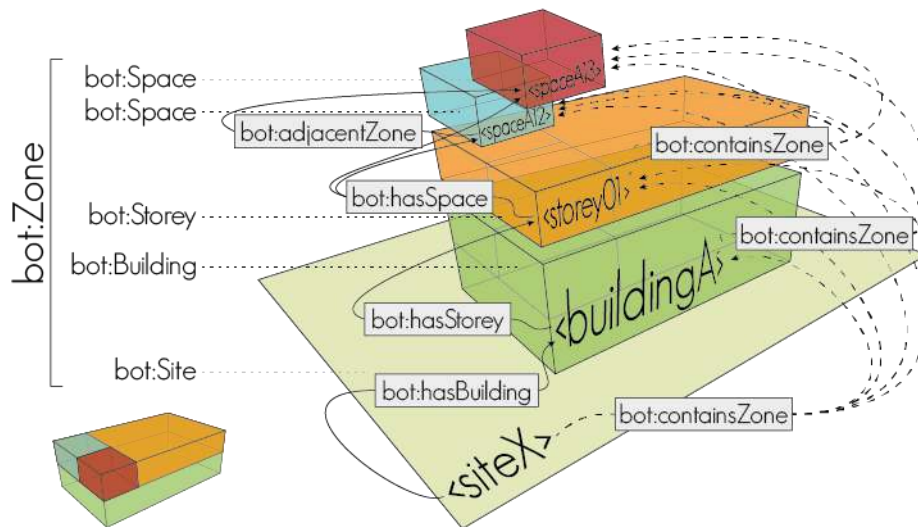


Figure 17. BOT Ontology - Examples of object properties linking classes

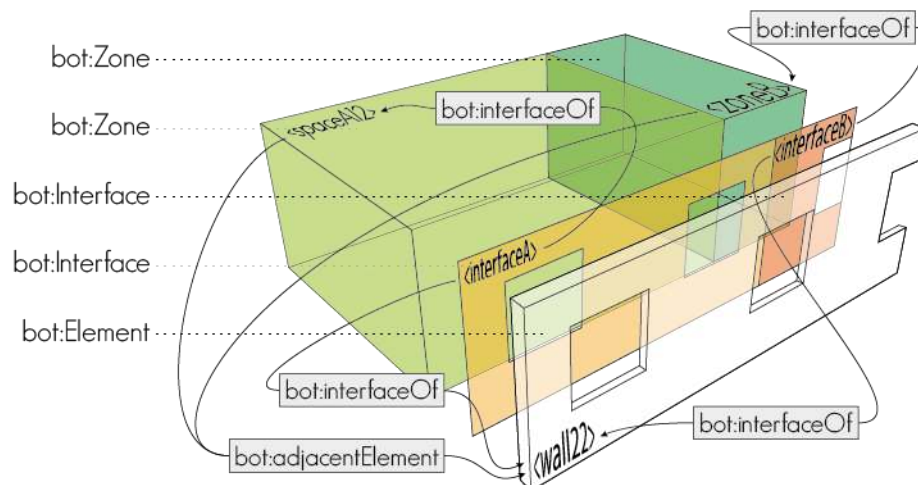


Figure 18. BOT Ontology - Example of object properties for the class bot:Interface

4.1.2 Organization Ontology

The Organization Ontology is a core ontology for generically representing organizational architectures and roles across a multitude of domains and designed to allow domain-specific extensions to add classification in the core ontology to provide a further level of specification. The areas represented by the ontology are the following: organizational structure, reporting structure (memberships, roles and relationships), location information and organizational history. This representation does not offer specific details of the different types of organizational structures, therefore, for this purpose, it is necessary to create extensions vocabularies. The Organization Ontology’s classes are Change Event, Formal Organization, Membership, Organizational

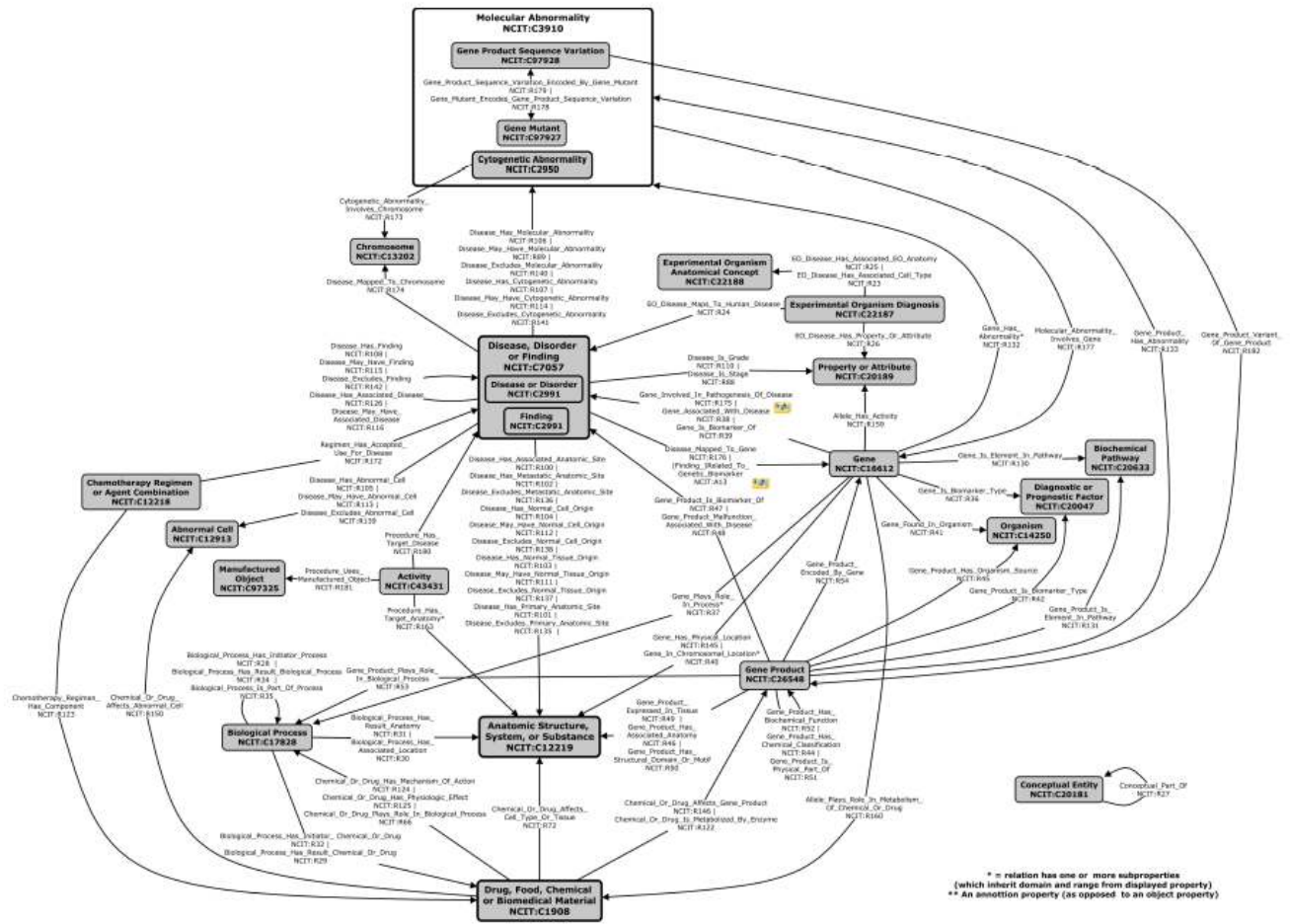


Figure 20. NCIT Ontology

4.1.4 SNOMED – Systemized Nomenclature of Medicine Ontology

SNOMED-CT Systemized Nomenclature of Medicine Clinical Terms Ontology is a clinical healthcare terminology system used for electronic healthcare records, The ontology includes concepts representing diagnosis, procedures, physical objects, body structures, and many other information about health records [38]. The main component types, Figure 21, of the ontology are:

- *Concepts*, a numeric code with clinical meaning that is not human comprehensible, but it is machine readable.
- *Descriptions*, there are two types of descriptions, the FSN-Fully Specified Name that is a description of meaning, and the synonym.
- *Relationships*

Regarding Odin Ontology the SNOMED-CT it is of relevant importance for the description of the hospital structure, i.e. the description of departments and units of the hospital facility.

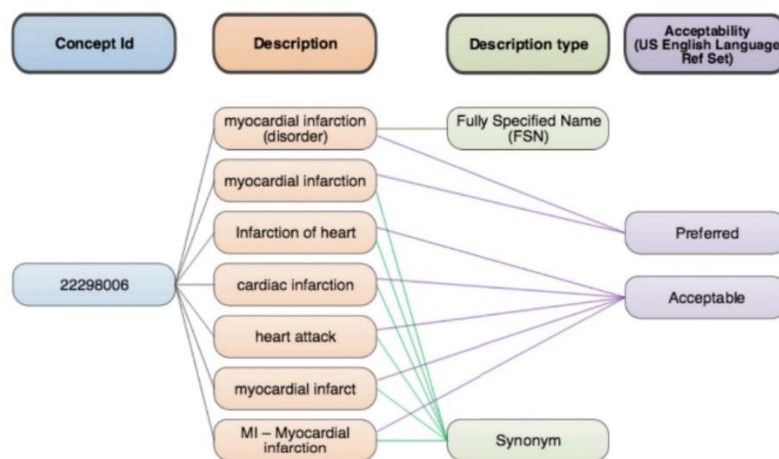


Figure 21: SNOMED-CT main types of components

4.1.5 ICD9CM – International Classification of Diseases 9

The International Classification of Diseases (ICD) is a classification system that organizes diseases and injuries into groups based on defined criteria. International Classification of Diseases 9th revision Clinical Modification - ICD9CM describes in numerical or alpha-numerical codes the medical terms in which the diagnoses of disease or trauma, other health problems, causes of trauma and diagnostic and therapeutic procedures are expressed [39]

The main classes of the ontology are Diseases and Injuries, and Procedures whose hierarchies are shown in Figure 22 and Figure 23.

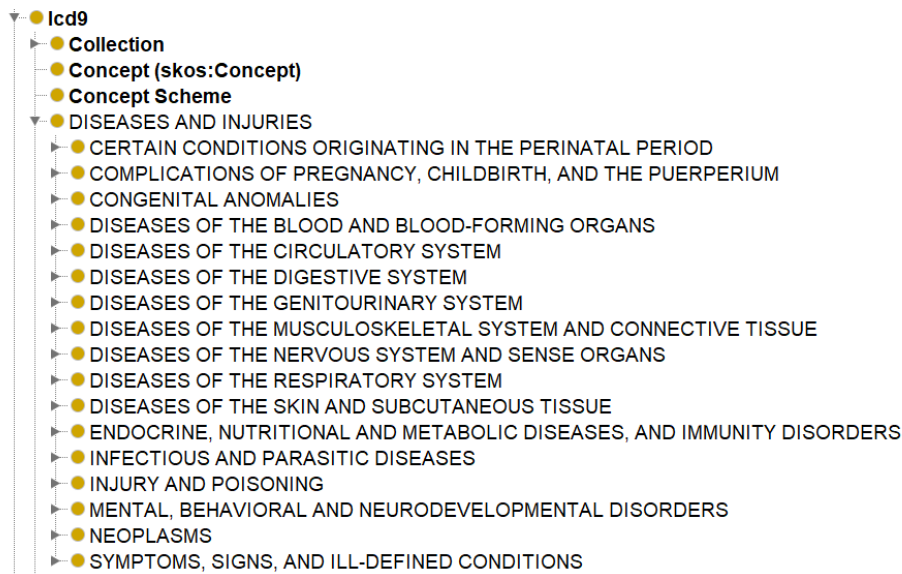


Figure 22: ICD9CM Diseases and Injuries class and subclasses

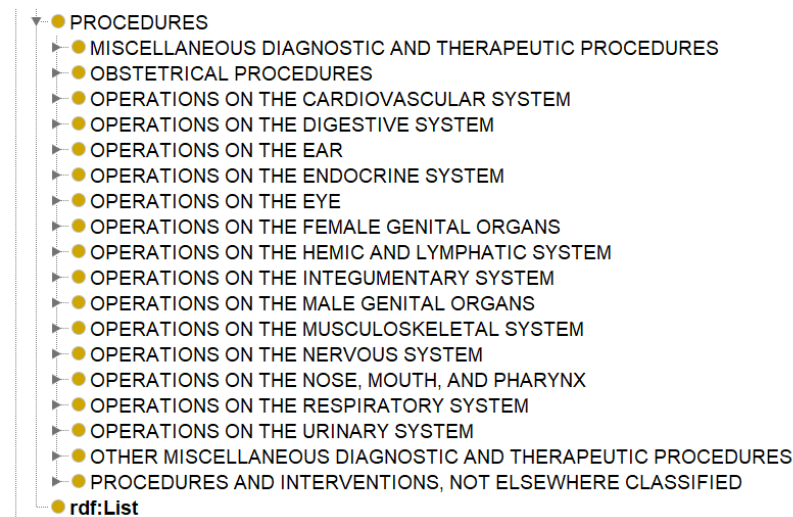


Figure 23: ICD9CM Procedures class and subclasses

4.1.6 CORA – Core Ontology for Automation and Robotics

CORA-Core Ontology for Robotics and Automation is defined by the IEEE 1872-2015 standard from the IEEE Robotics and Automation Society. The main aim of the ontology is to provide a semantic representation of the knowledge in the domain of robotics and automation. The result is a unified representation of a common set of definitions and relations that allow for the reasoning and communication of knowledge in this field. This ontology represents the fundamental concepts of the domain and serves as a base for more specific semantic representations. Its main concept is Robot, which is related to most of the remaining terminology through the subclasses of Device (a robot is a device) and Agent (a robot is an agent) [40]. Figure 23 provides a graphic representation of the taxonomy of the main concepts in CORA. This ontology has been selected for the Odin ontology to describe the Robots and Devices.

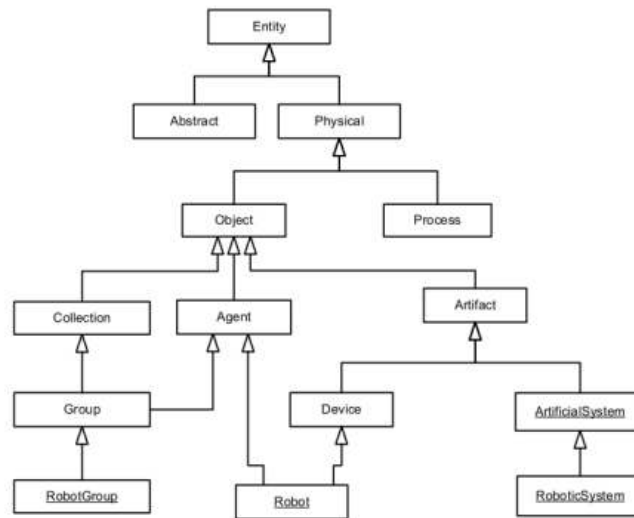


Figure 24. CORA Ontology

4.1.7 AI – Artificial Intelligence Ontology

Because AI is such an important aspect of the ODIN project, the AI-Artificial Intelligence ontology was chosen to construct the ODIN ontology. The Artificial Intelligence ontology is a comprehensive model of AI activities and applications. Engineering and Technical are the two main classes shown in Figure 25, with the one describing AI and its subfields and the second displaying the techniques employed in this subject.

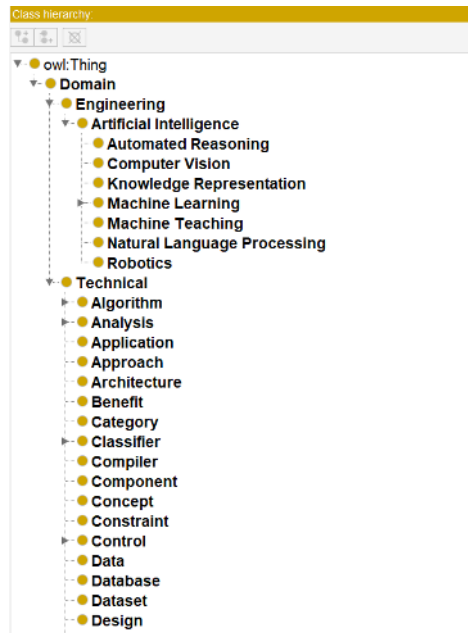


Figure 25: AI Ontology classes and subclasses

4.1.8 WoT – Web of Thing Ontology

The Web of Thing-WoT Ontology is an RDF axiomatization of the Web of Thing. The main aim of the ontology is to provide a semantic representation of knowledge in the domain of Web of Thing. This ontology represents the domain's fundamental concepts and serves as a foundation for more detailed semantic representations. The main classes are Thing, Interaction Pattern and Data Schema. Figure 26 provides a graphic representation of the taxonomy of the main concepts in WoT.

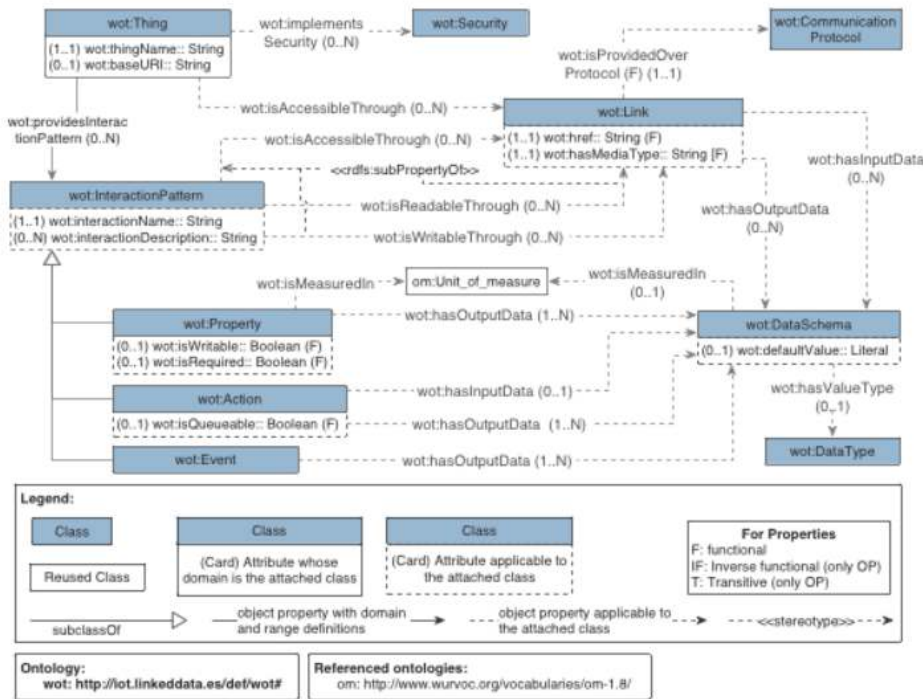


Figure 26: Overview of the WoT Ontology

5 ODIN ONTOLOGY

The main objective of the ODIN ontology is to provide the data model for an open digital platform to support services and Key Enabling Resources (KERs), enhanced by the Internet of Things (IoT), Robotics and Artificial Intelligence (AI) to empower workers, medical locations, logistics and interaction with the territory.

The introduction of ODIN technologies aims to allow real-time management of medical devices thanks to a combination of AI, IoT, robots, sensors, wearable devices for workers and the semantic solution for web architectures. The goal is to promote a better harmonization and standardization of Health Technology Management (HTM) within European hospitals. ODIN's goal regarding this area of intervention is to fill the lack of real-time exchange of information among staff. This has been an issue which in past years has caused the majority of adverse events in hospitals. This technology will be beneficial also in terms of disaster preparedness for future adverse events that could cause a great influx of patients and that would force hospitals to adapt their resources and layout.

At a semantic level, all this translates into the creation of an ontology capable of defining, through its classes and properties, any hospital structure, allowing for reuse of the ontology. Subsequently it has been extended by creating individuals capable of representing key points and pre-determined objectives. Starting with a description of the classes that make up the ontology and then defining what properties link the classes together, the next subsections explain the approach used and the decisions made to achieve the project's purpose. In order to have a representation of the San Carlos Hospital scenario, the individuals, or instances of the classes, are made explicit.

Protégé software was used to construct the ODIN ontology. Protégé is the most comprehensive ontology editor and is an open source project created by Stanford University. The ODIN project's ontology is based on version 5.5.0. The graphical user interface is simple and easy to use. The entities page is the main tab of Protégé, where you can see all of the classes and properties defined by the ontology. Other tabs, such as OWLVIZ and ONTOGRAF, provide a graphical and intuitive presentation of the classes, individuals, and relationships between them, allowing users to have a different representation than the entity tab. Both tabs display the asserted and inferred versions. On the basis of the concepts stated, a semantic reasoner validates the consistency of the produced ontology and extracts the implicit information. The ontological models created for ODIN were validated using the HermiT reasoner [41].

5.1 ODIN Ontology design and realization

The created ontology is the product of the previous section's analyses. Version 1 of the ODIN ontology is shown in Figure 27. It is evident from this that the selected existing ontologies were crucial to the ODIN's realization. The current ODIN ontology is documented and available at <http://192.167.217.252/>. Terms defined in the ontology are defined by Internationalized Resource Identifier-IRI in the namespace <https://odin-smarthospitals.eu/odin> which will be abbreviated to the prefix **odin**: in the next sections.

The result is an ontology with 11 super classes with numerous child classes, Device, Domain, Element, ICD9, Interface, Measurement, Occupation, Organization, Sites of Care Delivery, Unit of Measure, WoT. It also includes many object and data properties, as well as several individuals.

To define medical devices, a new ODIN EMDN ontology was created, representing each and every device class in the new European Medical Devices Nomenclature (EMDN), as detailed in section 5.2, in order to have any existing medical device available. This choice is, per se, a great achievement, considering that to the best of our knowledge there was no existing ontology ready to represent all the possible medical devices on the European market.

The classes, properties, and individuals of the ontology are explained in more detail in the next subsections, while in section 9 it is described how the ontology has been applied to a specific pilot and use case.

The main prefixes used are related to existing ontology as follows:

- NCIT Ontology includes prefix **Thesaurus:** and **obo:**
- SNOMED Ontology includes prefix **snomed:**
- CORA Ontology includes prefix **sumo-cora:** and **cora-bare:**
- AI Ontology includes prefix **technical:** and **ai:**
- BOT Ontology includes prefix **bot:** and **building:**
- WoT Ontology includes prefix **wot:**, **om:** and **core:**
- ICD9CM Ontology includes prefix **ICD9CM:**
- NCIT Ontology includes prefix **Thesaurus:** and **obo:**
- ORG Ontology includes prefix **org:**
- OdinEMDN Ontology includes prefix **odinemdn:**

5.1.1 Phase 0: Class Definition

The ODIN superclasses have been defined in accordance with the ontology domain and the goals to be reached, as shown in Figure 28.

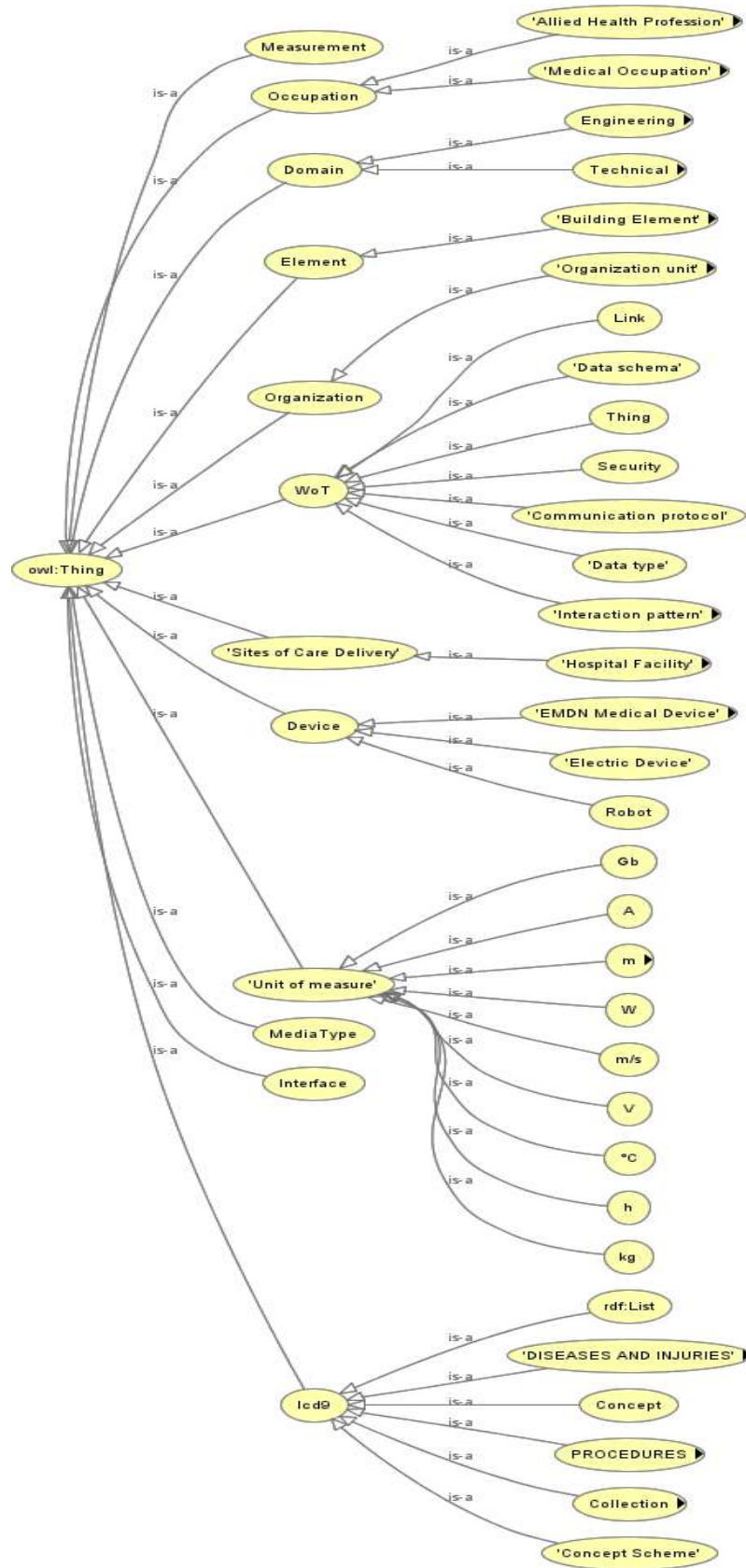


Figure 28: ODIN Ontology classes

Any tool capable of executing a given task is defined as a `sumo-cora:Device`, Figure 29, a superclass with the following subclasses:

- `sumo-cora:Electric Device`
- `cora-bare:Robot`
- `odin:EMDN Medical Device`

A copier or digital camera are examples of Electrical Devices. All electro-medical devices are found in the child class `odin:EMDN Medical Device`. Finally in the `cora-bare:Robot` subclass, any automatic mechanical operator can be defined.

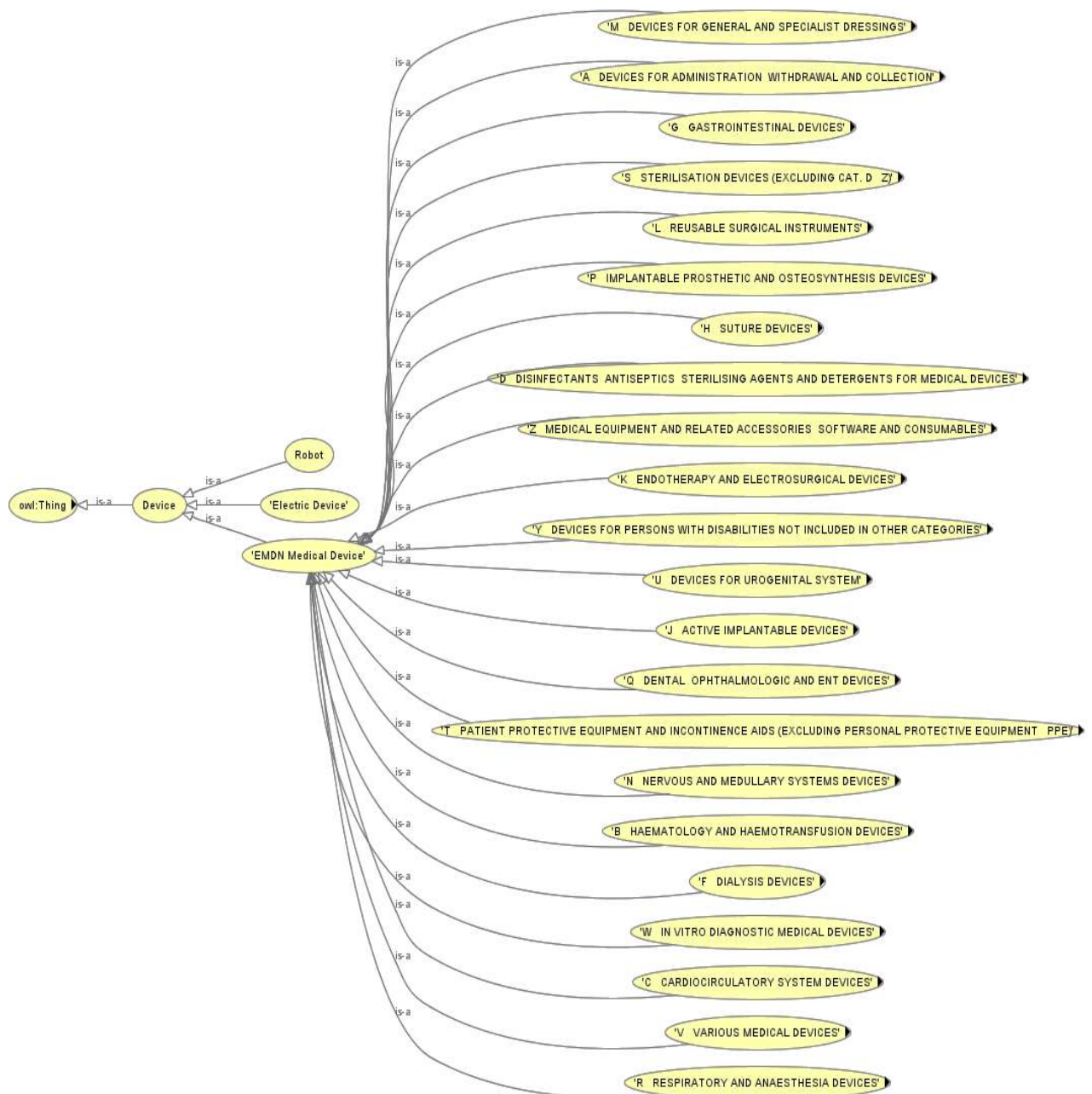


Figure 29: Device class of the ODIN Ontology

The technical:Domain class introduces **as** subclasses, Figure 30, Figure 31, Figure 32:

- ai:Engineering
- technical:Technical

Artificial intelligence systems are classified according to their ability to adapt and work independently using these subclasses.

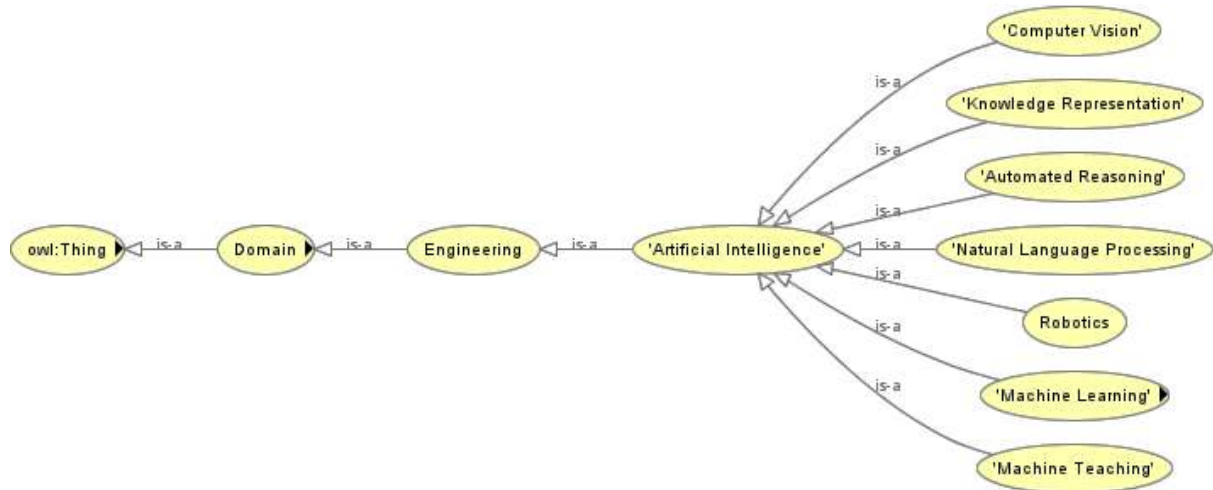


Figure 30: AI ODIN's Ontology

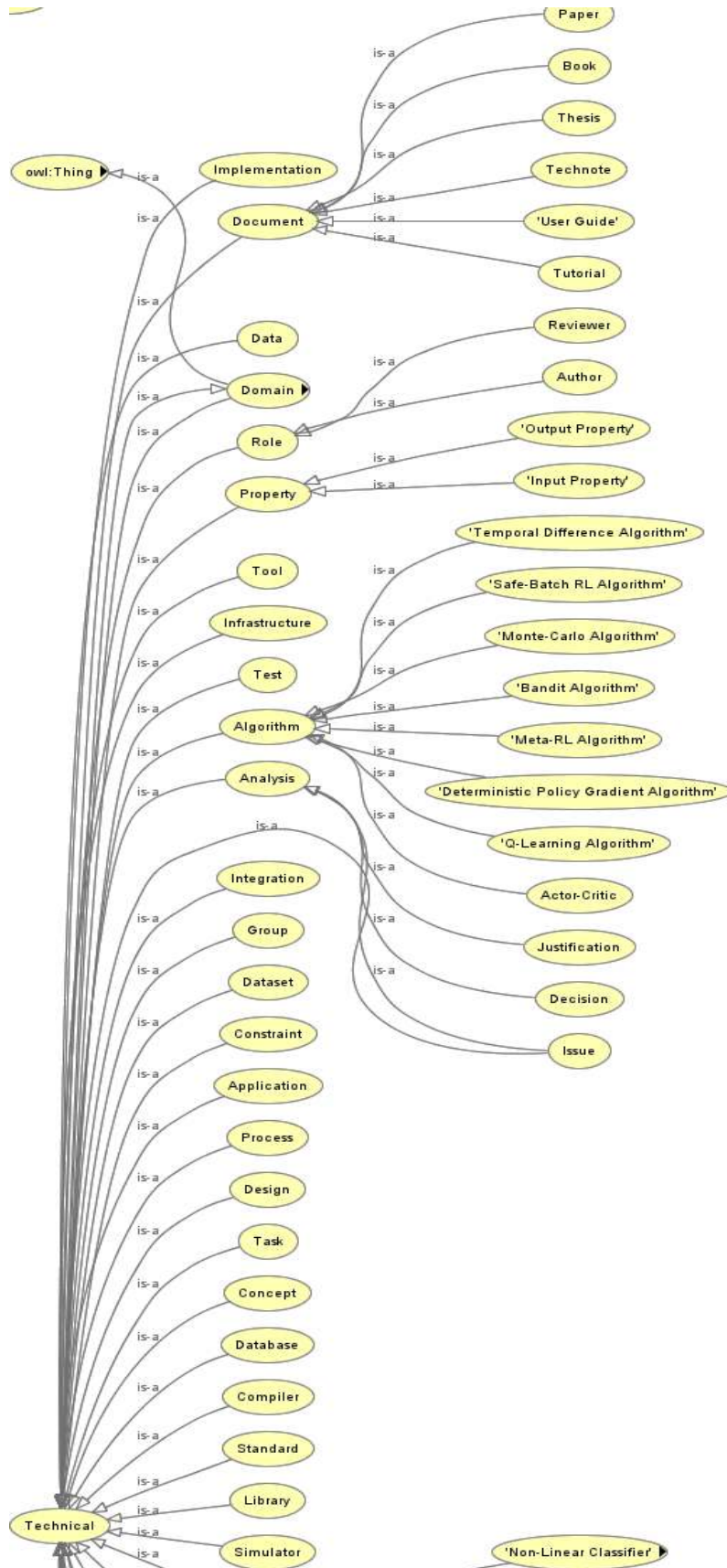


Figure 31: AI ODIN's Ontology

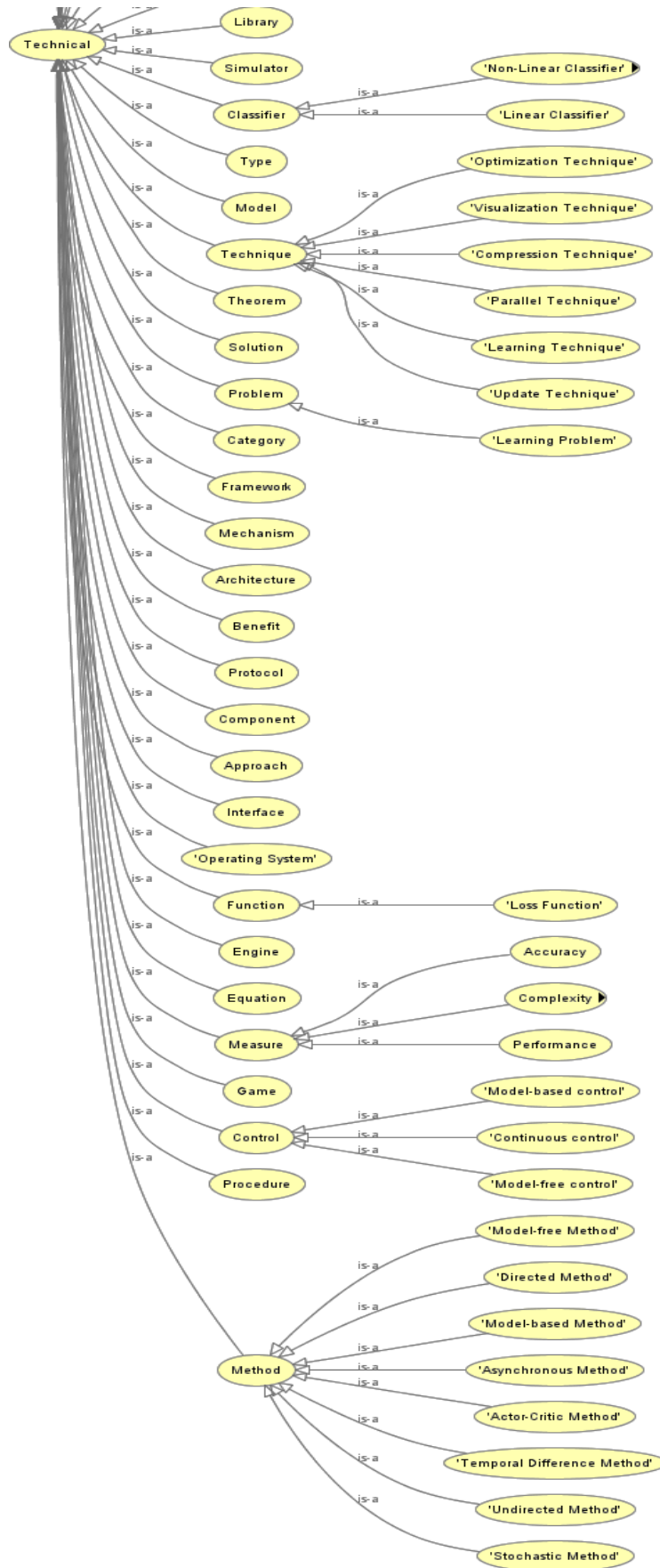


Figure 32: AI ODIN's Ontology

The building is defined by the classes building:Element bot:Interface and Thesaurus:Sites of Care Delivery, Figura 33, Figura 34, and their subclasses. A Thesaurus:Site of Care Delivery, for example, specifies any facility that delivers health services; as a result, a structure of this sort is referred to as a Hospital Facility. Buildings, storeys, and spaces make up the Hospital Facility, which is why they are its subclasses.

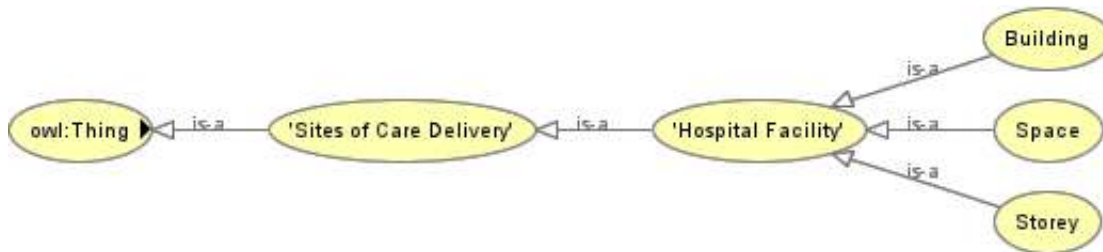


Figure 33: ODIN Site of Care Delivery class

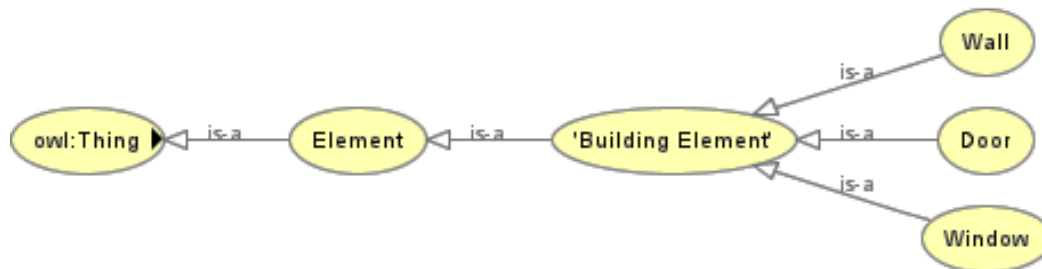


Figure 34: ODIN's ontology Element class

The om:Unit of Measure and core:Measurement classes come from WoT ontology. They define the units of measurement for a property, together with their subclasses, Figure 35. The subclasses of om:Unit of Measure, in particular, have been established to describe the robots in the ontology and their technical specifications. Because each class is required to represent situations relating to IoT platforms and application domains in regarding future expansions, the WoT Ontology has been integrated into ODIN.

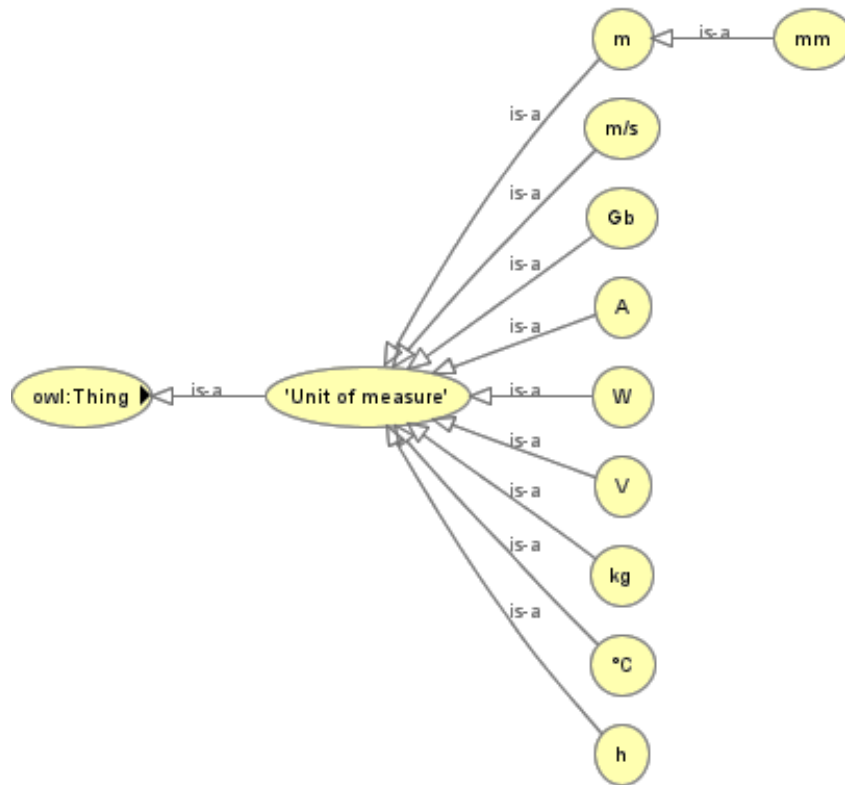


Figure 35: Unit of Measure class of ODIN Ontology

The `odin:lcd9` class defines the documentation linked to the ICD9CM ontology, whose primary classes are ICD9:DISEASES AND INJURIES and ICD9:PROCEDURES, with their subclasses, Figure 36. It was considered appropriate to import the whole ontology because it will be helpful in future expansions.

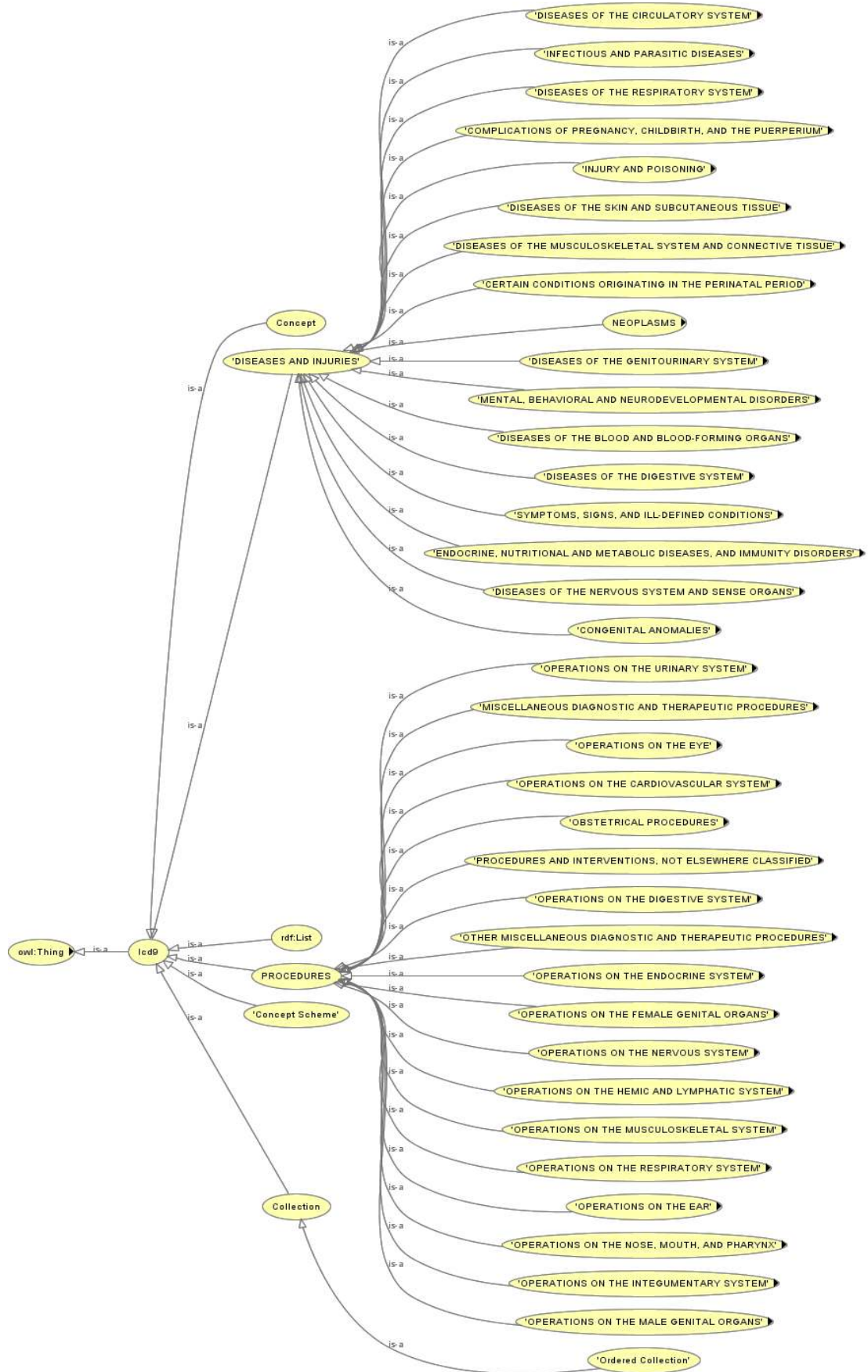


Figure 36: ICD9 class of ODIN Ontology

Any occupational role that can be founded in a hospital is under the class obo:Occupation. Its subclasses are:

- obo:Allied Health Profession
- obo:Medical Occupation

As a result, these can be used to define hospital staff, assisting in the development of the Odin as an exhaustive ontology. Figure 37 depicts what has been expressed.

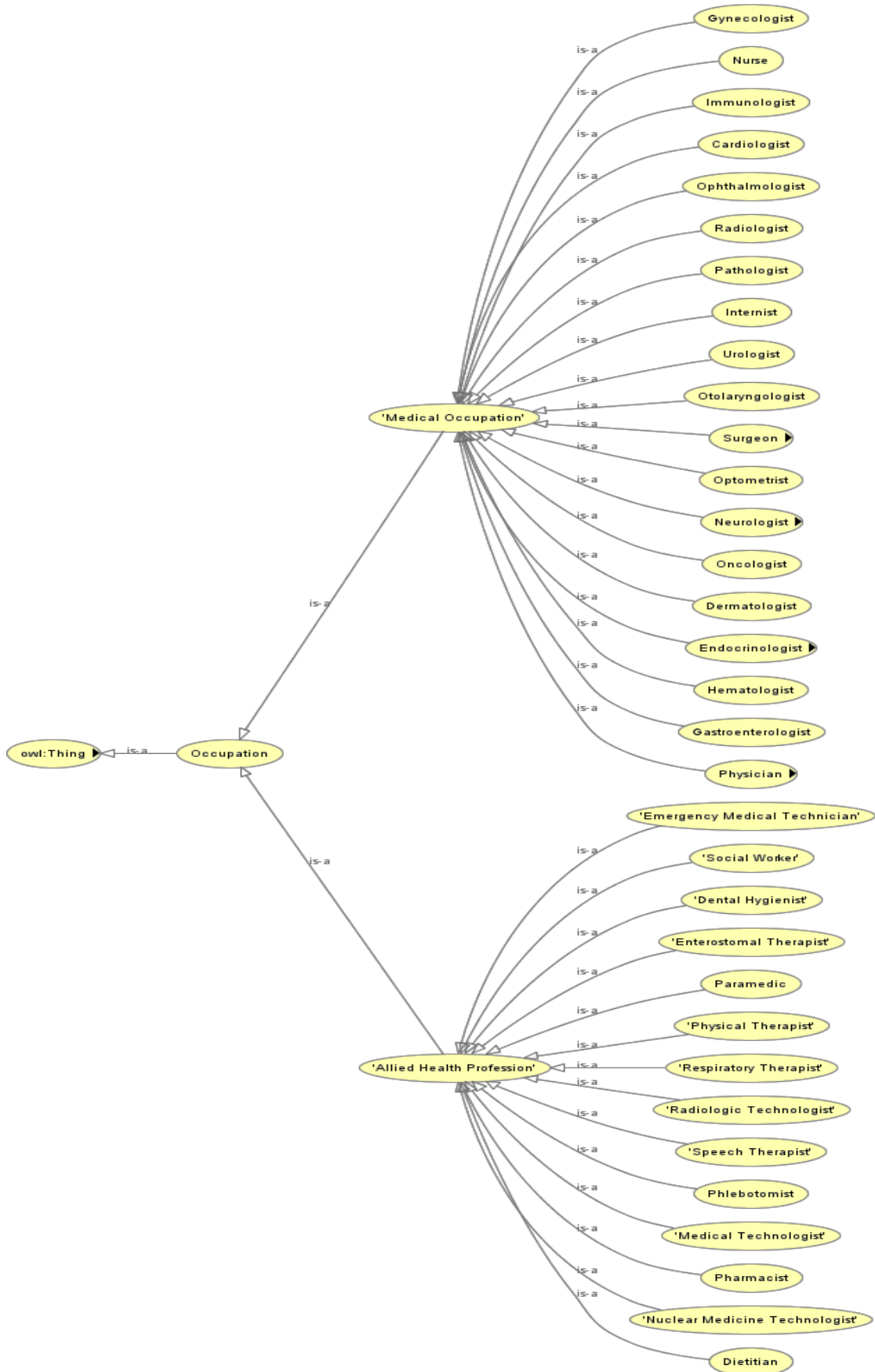


Figure 37: Occupation class of ODIN

Finally, the org:Organization class and its subclass org:Organization Unit have been chosen by the Organization ontology. An organizational unit, such as a department or a department unit, represents a component of an organization.

This class is used in conjunction with the SNOMED ontology to define all the departments and units that make up a hospital. These classes were found to be necessary for the definition of the hospital organizational structure during the creation of the ODIN ontology. For a more complete explanation, see Figure 38.



Figure 38: Organization class

5.2 OdinEMDN – European Medical Device Nomenclature Semantic Ontology

Medical devices are the final essential component for realizing the ODIN ontology. It is essential to explicitly identify where the medical devices are located, who utilized them, and their condition of usage in order to have a real-time flow of information. As a result, an ontology that incorporates all present medical equipment is clearly required in order for them to be utilised in the future. Because this was not available, the OdinEMDN ontology was established using the EMDN-European Medical Device Nomenclature [42], which was developed by the European Commission for the classification and registration of medical devices in the EUDAMED database.

The alphanumeric structure of the EMDN, Figure 39, which is formed in a seven-level hierarchical tree, distinguishes it. It divides medical devices into three hierarchical levels [43]:

- Categories, represented by a letter
- Groups, represented by two digits
- Types, represented by a sequence of number that can be maximum 13

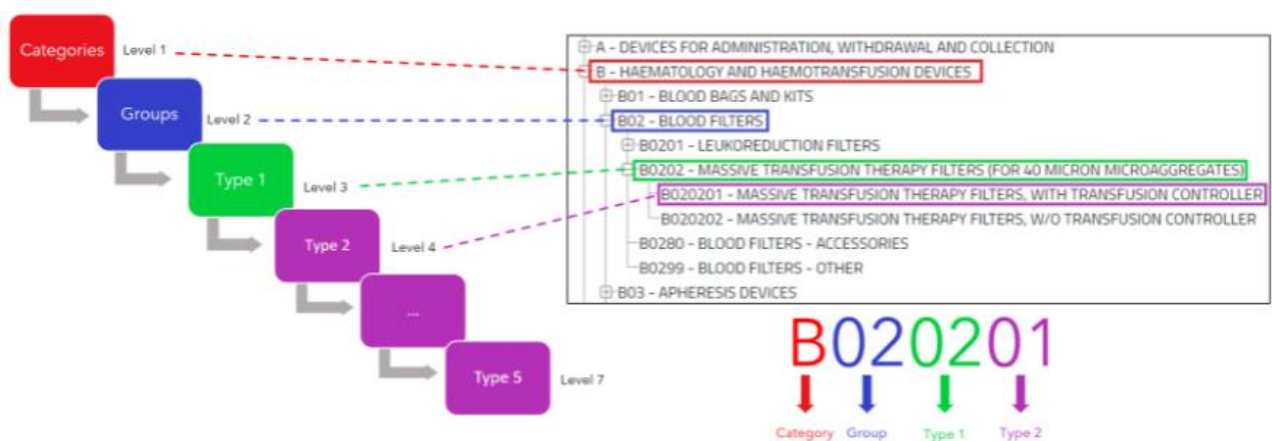


Figure 39: EMDN medical devices alphanumeric structure

5.2.1 OdinEMDN Ontology

The necessity for all medical devices to be available in order to use them in the Odin ontology led to the development of the OdinEMDN ontology. The approach and method used to create the ontology are described in the subsections below. The European database was obtained as an excel file, and transformation rules were built using the Protégé Cellfile plugin, which allows the program to generate the ontology's axioms. As a result, classes were generated that will become part of the ODIN ontology, as stated in section 5.1. Following that, as indicated in subsection 5.2.5, an object property was established that would allow the usage of all medical devices in the Odin ontology.

5.2.2 Phase 0: EMDN File Creation

The new EMDN version can be downloaded as an excel file, which contains the whole list of medical devices that have been appropriately adjusted to reach the final ontology.

5.2.3 Phase 1: Ontology Development

The classes were defined using an excel spreadsheet downloaded from the European Commission's website. The CellFile plugin, Figure 40, in Protégé allows users to download spreadsheet data into OWL ontologies. The transformation rules that link the excel data to the ontology are used to produce new axioms. The Mapping Master DSL, a domain specific language that extends the Manchester OWL Syntax, is used to build the transformation rules [44].

- A reference to a column or to a row is denoted as @A*, therefore this case refers to all the column cells
- Class: @D * SubClassOf: @A * asserts that all cells in column D are classes and these are also child classes of the equivalent cells in column A
- Annotations: rdfs: label @A or rdfs: comment @A * assign a comment and a label to the classes

The complete transformation rules used for the classes of the OdinEMDN ontology are shown in Figure 41, 42.

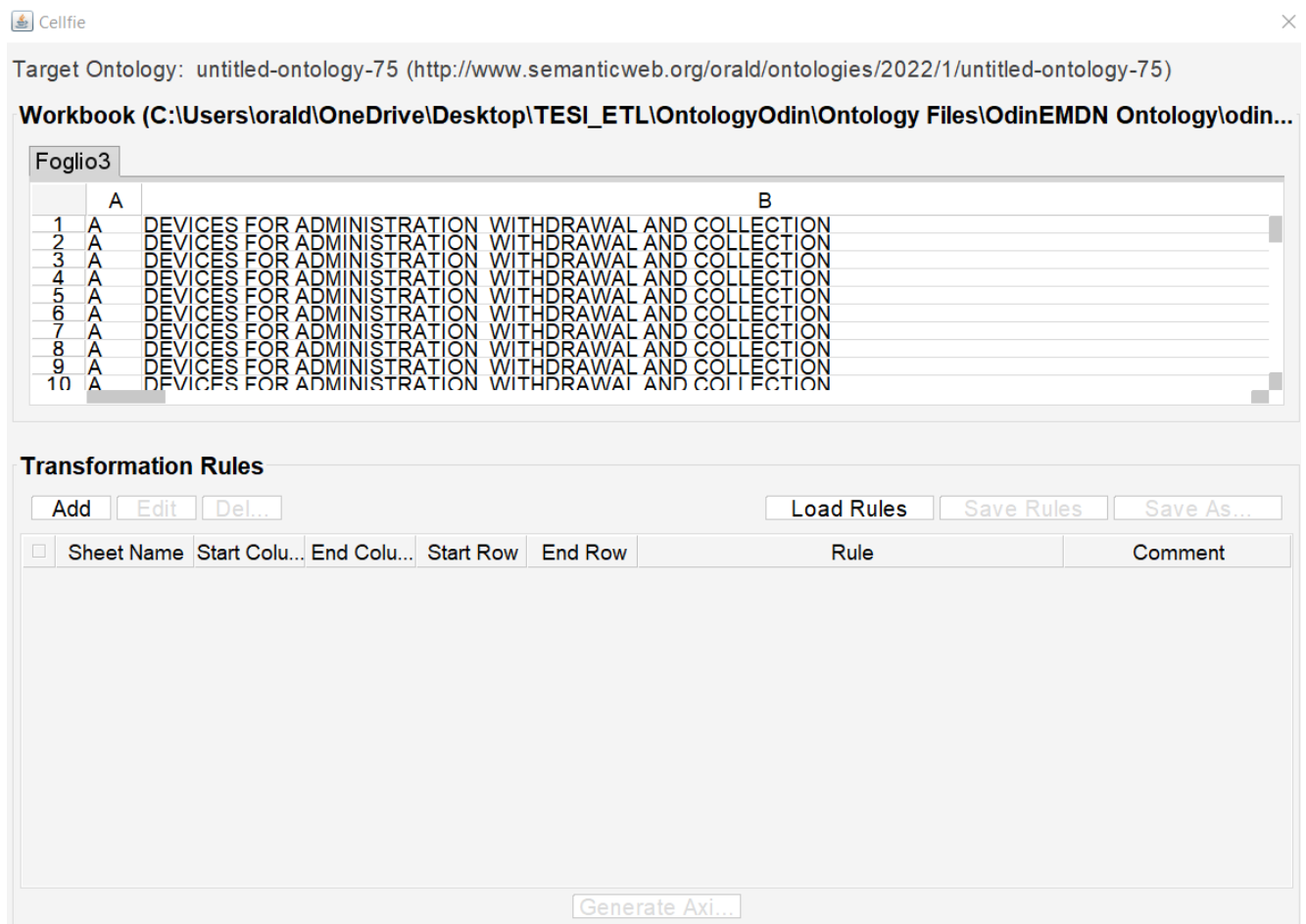


Figure 40: Cellfile plugin in Protégé

Add			Edit			Del...			Load Rules		Save Rules		Save As...	
<input checked="" type="checkbox"/>	Sheet Name	Start Colu...	End Colu...	Start Row	End Row	Rule		Comment						
<input checked="" type="checkbox"/>	Foglio3	D	D	1	+	Class: @D* Annotations: rdfs:label @F* Annotations: rdfs:comment @W*								
<input checked="" type="checkbox"/>	Foglio3	J	J	1	+	Class: @J* Annotations: rdfs:label @L* Annotations: rdfs:comment @W*								
<input checked="" type="checkbox"/>	Foglio3	A	A	1	+	Class: @A* Annotations: rdfs:label @C* Annotations: rdfs:comment @W*								
<input checked="" type="checkbox"/>	Foglio3	M	M	1	+	Class: @M* Annotations: rdfs:label @O* Annotations: rdfs:comment @W*								
<input checked="" type="checkbox"/>	Foglio3	J	J	1	+	Class: @J* SubClassOf: @G*								
<input checked="" type="checkbox"/>	Foglio3	D	D	1	+	Class: @D* SubClassOf: @A*								
<input checked="" type="checkbox"/>	Foglio3	G	G	1	+	Class: @G* SubClassOf: @D*								
<input checked="" type="checkbox"/>	Foglio3	P	P	1	+	Class: @P* Annotations: rdfs:label @R* Annotations: rdfs:comment @W*								
<input checked="" type="checkbox"/>	Foglio3	P	P	1	+	Class: @P* SubClassOf: @M*								
<input checked="" type="checkbox"/>	Foglio3	M	M	1	+	Class: @M*								

Generate Axi...

Figure 41: Transformation rules

<input checked="" type="checkbox"/>	Foglio3	M	M	1	+	Class: @M* SubClassOf: @J*			
<input checked="" type="checkbox"/>	Foglio3	S	S	1	+	Class: @S* SubClassOf: @P*			
<input checked="" type="checkbox"/>	Foglio3	S	S	1	+	Class: @S* Annotations: rdfs:label @U* Annotations: rdfs:comment @W*			
<input checked="" type="checkbox"/>	Foglio3	G	G	1	+	Class: @G* Annotations: rdfs:label @I* Annotations: rdfs:comment @W*			

Generate Axi...

Figure 42: Transformation rules

5.2.4 Class Definition

The Cellfile plugin develops the axioms as a result of them, and the construction of the classes may be seen in the class section shown in Figure 43. Figure 44 illustrates a graphical representation of the classes with OntoGraf.

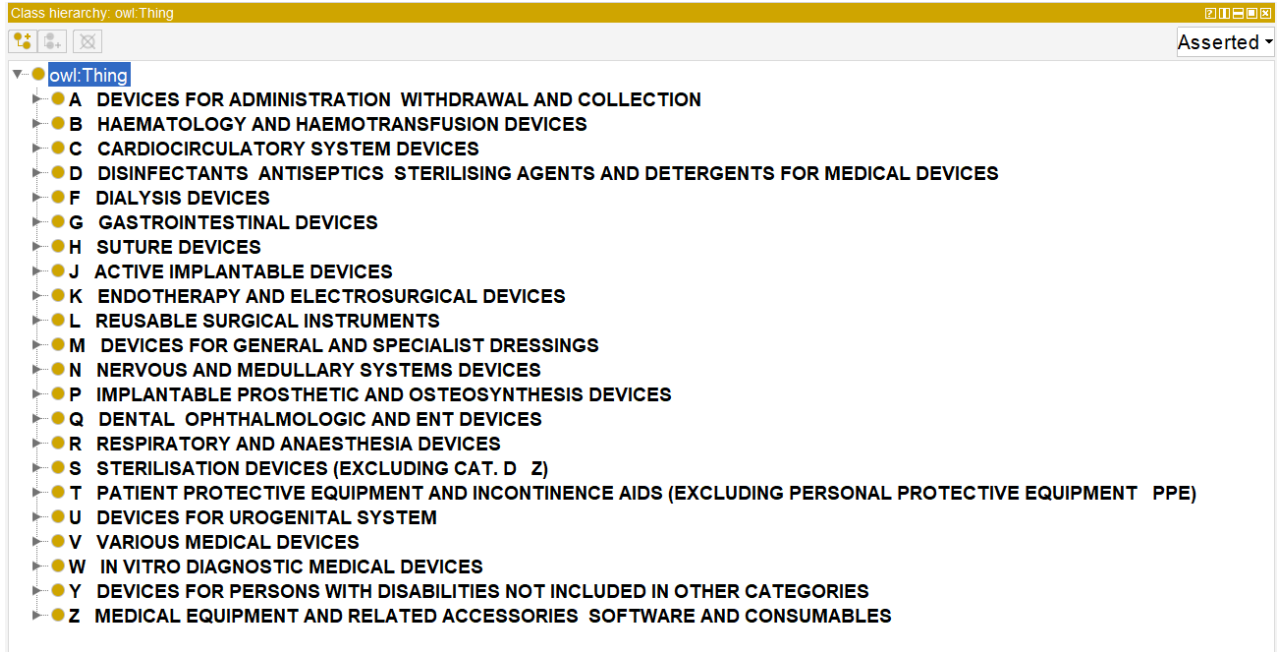


Figure 43: Odin EMDN Ontology classes

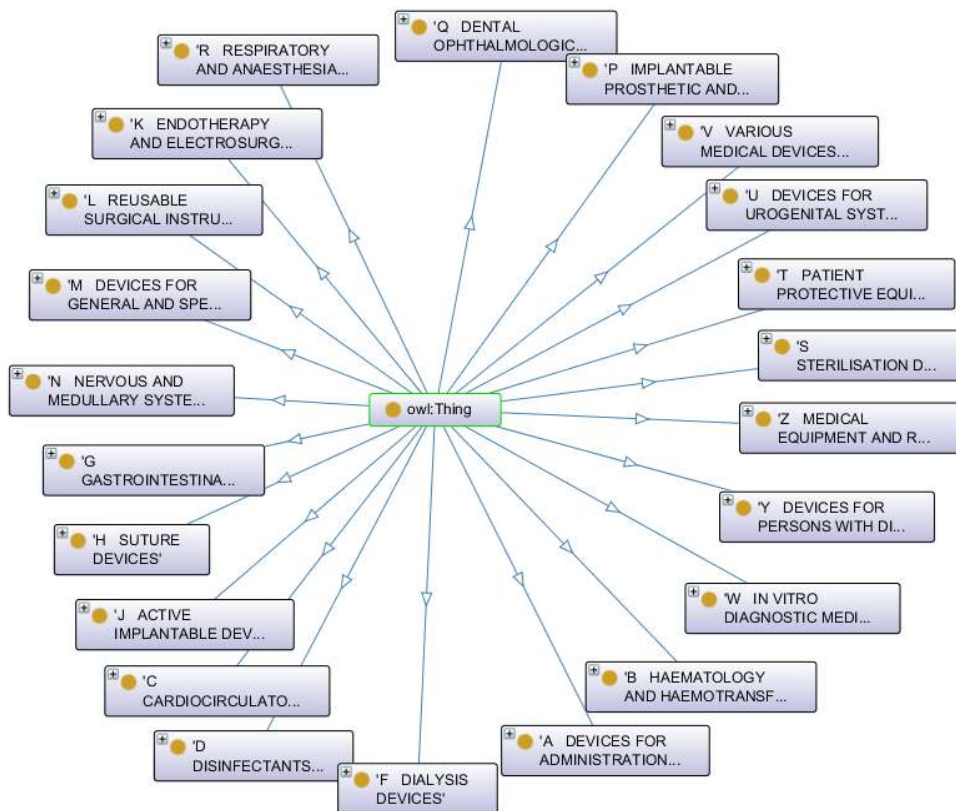


Figure 44: Odin EMDN Ontology classes, OntoGraf view

5.2.5 Object Properties Definition

Following the definition of the classes, the ontology is completed with the object property **odinmdn:has_medical_device**, which establishes a relationship between any space or building that contains a medical device. In the ODIN ontology, the **domains** are the classes Hospital Facility and Organization Unit, the **range** is the class EMDN Medical Device. It is an important property since it will aid in determining which rooms or units have one or more medical equipment in the ODIN ontology.

6 COLLECTION OF DATASETS FROM THE PILOTS

When hospitals will need to input data into the ODIN platform, either in the form of resources, as configuration of KERs, or just plain facts. This task may seem at first daunting, or trivial, and this impression will depend upon the knowledge of the available data, as well as the feasibility of the integration. In this section we offer a methodological approach to collect data and making it ready to be consumed by ODIN resources.



Figure 45 Dataset collection ETL methodology

The ETL (Extract, Transform, Load) methodology (see Figure 45) is divided into 4 phases. Each of the phases will help in the dataset acquisition ending in the data being accessible by any ODIN resource (process which will be explained in section **Error! Reference source not found.**), each phase will be explained in the following sections.

6.1 Data identification

The first phase consists of identifying the available data. Even if we are very well aware of the existence of datasets, sometimes we may be surprised how data can be found in plain sight, namely as presumed facts. The objective of the phase is to systematically analyse the available data and produce a list of potential data to be collected.

One way to identify data is to review the possible media (and format) the data may be in:

- File tables, in formats such as CSV or excel.
- Relational Databases, typically compatible with the SQL (Structured Query Language), and there are many dialects, which correspond with the different database management implementation, such as MySQL, Postgres, MariaDB, SQLite.
- Non-Relational Databases, these are a broad category of data management paradigms which do not rely on tables and relations between them. Table 6 shows some examples by types.
- API endpoints, modern applications have APIs for interoperability, these APIs may be REST (Representational State Transfer) which often offer JSON (Java Script Object Notation) representations of data. Older APIs may be based on XML (eXtensible Markup Language) using formal protocols such as SOAP (Simple Object Access Protocol).
- Standardised formats, there are many formats for many purposes, but in accordance with the presented ontology there are a couple of very relevant standards.
 - FHIR (Fast Healthcare Interoperability Resources) is the emerging standard for health care, so many modern systems may find they use this standard for data exchange
 - DWG, a CAD file standard particularly popular for floorplans.

- There are also many Image and video formats, however, and in anticipation of the next phase, we are not going to list them.
- General files (such as Word, PDF documents or Power Point presentations) and webpages, most of this media is intended for human consumption exclusively, which means that automatizing the information extraction from these sources is challenging; to denote this this type of data is labelled as unstructured data.
- Semantic formats (RDF, JSON-LD), in this case the data is already semantic.

Table 6. Some example types of Non-Relational Databases and implementation (source Wikipedia)

Type	Notable examples of this type
Key-value store	Azure Cosmos DB, ArangoDB, Amazon DynamoDB, Aerospike, Couchbase
Object database	Objectivity/DB, Perst, ZopeDB, db4o, GemStone/S, InterSystems Caché, JADE, ObjectDatabase++, ObjectDB, ObjectStore, ODABA, Realm, OpenLink Virtuoso, Versant Object Database, ZODB
Document store	Azure Cosmos DB, ArangoDB, BaseX, Clusterpoint, Couchbase, CouchDB, DocumentDB, eXist-db, IBM Domino, MarkLogic, MongoDB, Qizx, RethinkDB, Elasticsearch, OrientDB
Wide Column Store	Azure Cosmos DB, Amazon DynamoDB, Bigtable, Cassandra, Google Cloud Datastore, HBase, Hypertable, ScyllaDB
Graph database	Azure Cosmos DB, AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso

Another procedure to identify potential sources of data is to analyse the domains, or the logical categories, of the information. Particularly the ontology and terminology (see Section 5) already offer the list of these domains.

6.2 Data valuation

Once a comprehensive list of available datasets and/or their data sources is available, this phase will focus on deciding which of these is valuable enough for continuing the process. As there may be many datasets a cost-benefit analysis must be performed at this stage, since the next steps could imply an unnecessary overhead for data which would have no use in ODIN’s context. Additionally, the system performance when data is harmonized could be hindered if many data sources are connected. Thus, the importance of filtering the relevant information sources.

The criteria to apply are:

- Is the information relevant? Specifically consider how much would each dataset contribute to the use cases related to ODIN. For example, Data could be very valuable if can be fed to High Level AI to produce models.
- Is the information representable? For example, images are not representable in the ODIN ontology.
- Is the information available? This might be obvious since the first step has ensured the information exists, but maybe this information is behind protective measures, making it unavailable.

- Volume of the information. Considerations of how much information needs to be mapped and then connected will affect the costs. Consider the number of mappings as well (see section 6.3), if the volume of information is large but only one mapping is necessary, is much less costly than a large volume of information where each datapoint needs to be individually mapped.
- Quality of the information. Consider if the data to be shared provides from reliable sources, and if the dataset is consistent. E.g. data providing from wearable devices may be abundant, but it needs to be considered if the data points are guaranteed to be from the said user, and in the specified conditions; or datasets that are manually collected consider the possibility of errors in transcription which affect the dataset quality.
- Potential connection mechanism (see section 6.4), the type of connection will affect the overall cost, native being the cheapest, manual being the more costly option.
- Privacy and legal considerations, ensuring for example GDPR rights and regulations are followed and maintained when collecting the data.

6.3 Data mapping

In this phase the objective is to analyse the path between the source's data structures and how it is represented in the ODIN ontology. Once the data is validated, and the data that is deemed suitable for collection, then the process of mapping consists of identifying the source data model and analysing the necessary changes to the data, in order, to be adapted to the ODIN ontology and thus suitable for collection in the ODIN platform.

The process of mapping is to be performed at high level, yet with the understanding of the data models involved and the processes that need to happen, in order, to achieve the mapping. For this reason, participants in the process, may include not just technical personnel adept in each of the data models involved, but also experts in the data being analysed. The experts will help complete the missing semantics from the original data that is typically required for a successful mapping.

The output of the process is a list of instructions, or specifications, describing the process of transforming each data point from the original model to the ODIN ontology. Typical instructions are:

- Static injection, some properties, or metadata, do not appear in the original model because they are all assumed to be a single known value and it is more efficient just not to include it; this is typically referred as semantics. The target model will typically have explicit semantics, which require this value to be included. Therefore, sometimes in semantic data mapping it is necessary to define static values for target model for a given dataset.
- Attribute renaming, when a property of an object (or column in a table) is referred in different ways in the different models.
- Attribute processing, when the target value is a processed value from one or more of the original values. This type of transformation takes the form of a formula.
- Advance processing, when the original data model is not object oriented, the transformation requires advance processing, this is the case for image feature extraction where the original data model is a 2D raster, and the target model is an object with features analysed in the original image.

6.4 Data connection

Once the conceptual mapping is established, the technical connection of the data needs to be implemented. This phase is about the technical methods by which this is achieved. These methods will depend on the type of data, and the data mapping achieved, but other considerations like batch connection (i.e. when a data set is connected for all the current data points at the time of connection) and stream connection (i.e. the data source is connected the data points are transformed as soon as they are available).

The following sections describe general methods in increasing level of complexity and costs.

6.4.1 Direct approach: ODIN native

This is the simplest approach, as almost nothing must be done to connect the data, other than connect interfaces. However, this approach has very strong preconditions to be able to apply it.

The original dataset needs to be already in RDF or FHIR format, and the conceptual mapping must be empty to be able to directly connect original data source to ODIN. When this is possible the original data points can be sent directly to ODIN through its interfaces, in a native way.

6.4.2 Tool Assisted approach

This approach is characterized using tools to assist in the transformation of the data points. These tools might include R2RML (see section **Error! Reference source not found.**), where the conceptual mapping is codified into the proper script so that the mapper or streamer processes can autonomously operate transforming and connecting the data. In the case of R2RML the original data needs to be structured data, either CSV, a relational database or JSON based formats.

There are other tools to aide in content transformation, some may use standardized data formats to generate structured data from them. One example may be the WASP tool which aides in the transformation of floorplans to semantic representation of the building spaces.

The general characteristic of tool assisted approach is that the tools help create an automatized or semi-automatized process of transformation and connection of data.

6.4.3 Automatization approach

When the source data is more complex, or at least collecting data from it is complex, custom code implementation may be required. In this case the purpose is to develop the conceptual mapping into a program which performs all the complex operations needed to extract, process, analyse, structure, transform and send the data.

This approach would be the type of method used for interpretation of medical images, automatically extracting structured data from said images. The implementation techniques of these may go beyond what a mapping assistive tool may be able to offer, such as custom machine learning models.

6.4.4 Manual approach

Finally, when the mapping is impossible to automate, or the valuation of the data determines that given the small size of the source dataset it is not beneficial to invest in an automation process; then the final option is to manually map the data. This option may be the preferred option for unstructured data.

For manually transforming the data, interactive data transformation tools such as Protégé could be used as alternative to raw text editing of the OWL individuals. Protégé is a very generic

software, and as such it may not be suitable to aid in the manual mapping of data, especially if the person mapping is not knowledgeable about semantic technologies. Thus, to have the appropriate experts mapping the data, it may be interesting to develop an ad-hoc application which aids the experts in the manual mapping, offering custom interfaces and functionalities for the mapping.

7 HARMONIZATION OF RESOURCES, ODIN COMMON DATA MODEL

This section covers data harmonization within the ODIN platform from a practical point of view, i.e. which components are involved and which procedures are followed to represent and transfer data within the ODIN platform, in a common data model.

In ODIN, data is provided or generated by *resources (KERs)*. When a resource is registered to the platform, it announces the types of data that it will produce or consume, which allows it to be used by other resources. Resources communicate by sending data adhering a common data model, as a common language between them, therefore proper connectors that transform the original data to this common data model need to be also supplied. After establishing this communication, data and resources are ready to be discovered and used by other resources in custom application scenarios. The following sections describe the above procedures in more detail.

7.1 Resource registration and representation

Resources in ODIN are the bottom-level components that provide functions to support the ODIN use cases:

- Robotic platforms, providing on-site manipulation of physical objects and interaction with humans to facilitate operations;
- IoT devices, providing sensing mechanisms to collect data from the hospital environment;
- Artificial Intelligence (AI) algorithms, providing advanced methods for learning, pattern detection, prediction and optimization;
- Software components, such as front-end and back-end services, providing application-specific processing, user interfaces, APIs, etc.;
- Raw data, in the form of files, databases or APIs, such as equipment information, hospital stock, employee catalogues, organization charts, etc.
- A special kind of resource is the Hospital Information System (HIS), which provides clinical data to the platform, in the form of Electronic Health Records, possibly in a variety of formats (e.g. those reported in Section **Error! Reference source not found.**).

To use one of these resources in the ODIN platform, the resource needs to be registered to the system. Resource registration involves mapping the resource to the ODIN ontology and registering it to the catalogue of available resources. Resource registration involves specifying, among others, the following information:

- An identification and description of this specific resource, so that it can be referred to by others.
- Which resource type this resource is, by mapping it to the existing types of entities of the ODIN ontology, e.g. if it is a motion sensor, a particular type of robot, or an AI algorithm for crowd detection.
- Which types of information this resource produces, e.g. a motion sensor could produce information of type “detected motion”, along with its ID and measurement timestamp. The types of information are mapped to the available types of information available to the ODIN ontology.

- Which types of information this resource consumes, e.g. an AI algorithm for crowd detection can use information from motion sensors and cameras. These types are again mapped to the available types in the ODIN ontology.
- Any resource-specific configuration information, such as the floor or particular location in which the resource is installed, the IP address it has been assigned, etc. Some or all of this information will be also mapped to the ODIN ontology, to create semantic relationships between resources, such as a motion sensor entity belonging to a specific floor entity, or two motion sensors being adjacent to each other within the hospital.

To register this information, the resource manager component that handles registration needs to be in communication with the ODIN data model, to provide the end-user with the available entities and relationships, so that the user can input the necessary information, filling the respective forms and fields.

The outcome of resource registration is a *Resource Descriptor*, which is the collection of all the above information in a common record describing the resource with respect to its type and the information it handles. The Resource Descriptor is stored in the Resource Directory, which is a directory built on top of the ODIN ontology to hold information about the resources available at the hospital. The Resource Descriptor also contains other information not directly relevant to the ODIN data model, such as any APIs that this resource should expose to the outside world through ODIN's API gateway, or permissions for making this resource available to other ODIN instances. The overall process of resource registration is shown schematically in Figure 46.

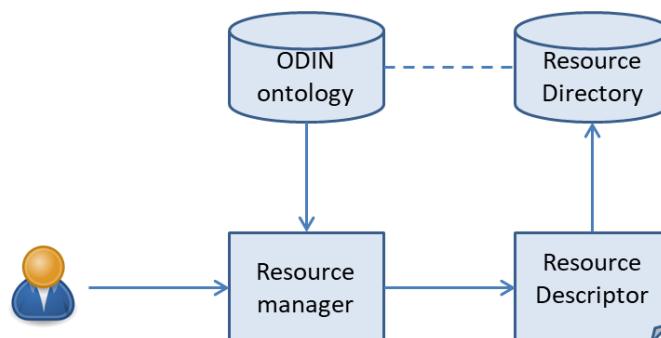


Figure 46: Resource registration process.

7.2 Transforming data to the common data model

Once a resource is registered, it can be connected to the ODIN platform and used within application scenarios, communicating with other resources, with the platform and with the end-users. However, connection to the ODIN platform means that the data model used by the resource is transformed to the common data model used by the ODIN platform. This level of abstraction ensures that any resource can be connected to ODIN and communicate with other resources, regardless of how information is represented within the resource.

To achieve this level of abstraction, all resources are connected to the main communication bus, the *Enterprise Service Bus (ESB)*, through appropriate *connectors* that perform, among others, *semantic translation*. This part of the ODIN architecture can be seen in Figure 47. Each resource has an appointed connector that performs the necessary operations so that the messages produced by each resource are transformed to the common ODIN data model before entering the ESB, or inversely, the messages received by the resource from the ESB are transformed to the resource's internal data model before being processed by the resource.

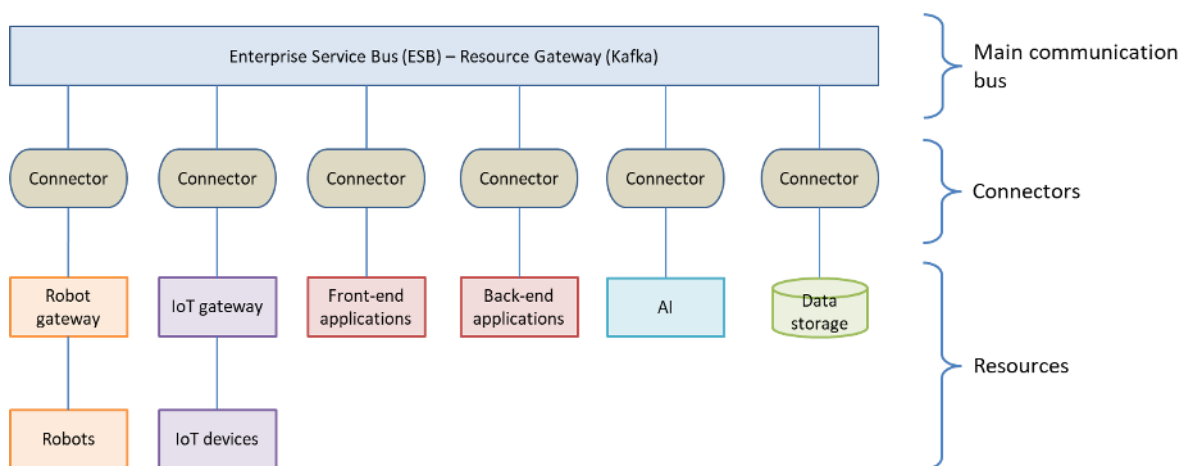


Figure 47: Resource communication within the ODIN architecture.

Each connector performs two levels of operations, as depicted in Figure 48: semantic translation and transport services. Apart from semantic translation, which is relevant to the ODIN ontology and will be presented in more detail below, transport services provide the necessary message transformations and encapsulations so that the messages produced by the specific protocol used by the resource (e.g. MQTT, SOAP, HTTPs, etc.) are transformed to the messages that can be delivered using the protocol of the ESB (e.g. Kafka).

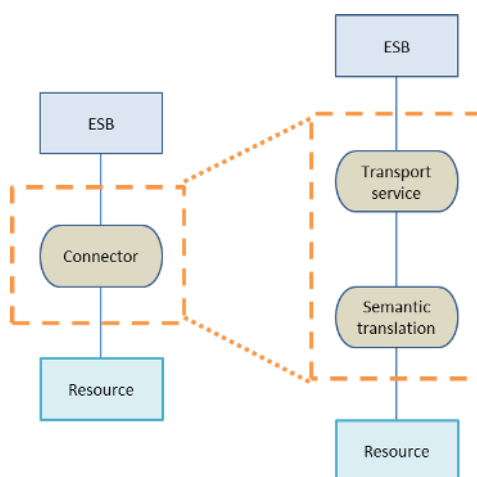


Figure 48: Each connector performs semantic translation and transport to the ESB message format.

Focusing now on the semantic translation part, each resource may use its own internal data model for representing its data, e.g. a specific JSON format, or MQTT messages of a custom comma-separated format and nomenclature. On the other hand, the ODIN platform uses the ODIN common data model as a common language between resources. The ODIN common data model is a manifestation of the ODIN ontology using a data representation standard that can support all relevant information. A possible candidate for the ODIN data model is the FHIR specification, described in Section **Error! Reference source not found.**, which, through its extensions, can support a wealth of information.

To transform data from the resource’s internal data model to the ODIN data model, a mapping needs to be specified, as described in Section 6.3. As mentioned in Section 6.4, this mapping can be implemented either automatically or manually. The purpose of the semantic translator inside the connectors of Figure 48 is to support automatic transformation so that each time the

resource sends or receives data to/from the bus, it is automatically translated to the ODIN data model.

Automatic semantic translation can be implemented using one (or more) of several tools available online. These tools accept as input the input data, along with the instructions of how to map them to the output format. These instructions are usually written in a convenient language, such as R2RML (see Section **Error! Reference source not found.**). The tool then automatically transforms the input data to the output harmonized data model, according to the instructions, as shown in Figure 49.

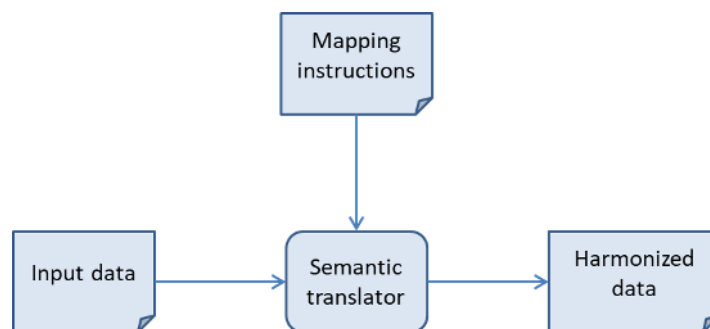


Figure 49: Automatic semantic translation and data harmonization process.

Table 7 lists a number of existing tools that perform automatic semantic translation, following the process of Figure 49. Automatic translation in the ODIN platform can make use of one (or more) of these tools to perform translation. In cases when the data is in complex data formats that are difficult or impossible to automatically translate, ad-hoc translator scripts may be instead used.

Table 7: Existing tools for automatic semantic translation.

Name	Description	Input formats	Output formats
RMLMapper ¹	RMLMapper executes RML rules to generate high-quality linked data from multiple originally (semi-)structured data sources. RMLMapper loads all data in memory.	CSV, JSON, XML, SQL databases	NQuads (default), Turtle, Trig, Trix, JSON-LD, HDT

¹ RMLMapper, <https://github.com/rmlio/rmlmapper-java>

RMLStreamer²	RMLStreamer generates RDF from files or data streams using RML. The difference with other RML implementations is that it can handle big input files and continuous data streams, like sensor data.	CSV, JSON, XML, SQL Databases	RML rules
CARML³	CARML is a Java library that transforms structured sources to RDF based on RML mapping.	Database sources, XML files	RML rules
Morph-KGC⁴	Morph-KGC constructs RDF graphs from data sources with R2RML and RML mapping languages. It is based on the Pandas library and makes use of mapping partitions to reduce execution times and memory consumption.	Relational databases: MySQL, PostgreSQL, Oracle, Microsoft SQL Server, MariaDB, SQLite. Tabular files: CSV, TSV, Excel, Parquet, Feather, ORC, Stata, SAS, SPSS. Hierarchical files: JSON, XML.	RML. Output RDF serializations: N-Triples, N-Quads

² RMLStreamer, <https://github.com/RMLio/RMLStreamer>

³ CARML, <https://github.com/carm/carm>

⁴ Morph-KGC, <https://github.com/oeg-upm/Morph-KGC>

SDM-RDFizer⁵	SDM-RDFizer interprets RML mapping rules to transform (un)structured data to RDF graphs.	CSV, JSON, RDB, XML	RML (TriplesMap)
RocketRML⁶	RocketRML is an implementation of the RDF mapping language (RML) in Javascript.	XML, JSON, CSV	RML

Semantic data transformation is directly linked to the Resource Descriptor of each resource, as stored in the Resource directory; the target information of the mapping should be the one that is described in the Resource Descriptor, in order for the resource to be properly used by other resources.

7.3 Data model-related ODIN services

Mapping resources and the associated data to the ODIN ontology through the Resource Descriptor and semantic translators is an important part of the ODIN platform, providing a level of abstraction on top of resources that harmonizes their semantics. This semantic abstraction not only allows communication between diverse resources, but also enables a rich set of services for discovering and using data and resources, and combining them together. Some of these services are the following.

- **Resource Directory:** The Resource Directory is the place where the Resource Descriptors of the available hospital resources are stored. The Resource Directory is accessible to the end-users, for examining the available resources, but also to other ODIN components, which can query for specific resources to retrieve relevant information.
- **Resource communication:** Semantic translation enables resource communication on top of the ESB. Resources can publish messages adhering to the common data model, and can subscribe to messages of other resources, without internally dealing with different representation formats. These details are hidden in the semantic translation connectors. Moreover, the semantic abstraction of the resource data allows other resources to more easily discover relevant topics (e.g. listen for motion data, irrespectively of the specific types of motion sensors deployed).

⁵ SDM-RDFizer, <https://github.com/SDM-TIB/SDM-RDFizer>

⁶ RocketRML, <https://github.com/semantifyit/RocketRML>

- **Semantic resource discovery:** Semantic resource discovery means that resources can discover other resources by querying the Resource Directory using semantic queries, i.e. searching for resources fulfilling specific semantic criteria. For instance, a resource might search for AI models performing crowd prediction, for motion sensors near a robot, for smart boxes inside patient rooms, etc. In such queries, the types of resources (e.g. “motion sensor”, “crowd prediction”, “smart box”) correspond to semantic types rather than specific devices of particular vendors, while relations such as “near”, “inside”, etc. have corresponding definitions inside the ODIN ontology.
- **Resource federation:** Semantic abstraction is also important for resource federation, i.e. allowing resources of one ODIN instance to be shared with another ODIN instance. ODIN instances can search for another instance’s resources of particular types or relationships using semantic queries, as if they were searching for local resources.
- **Resource Choreographer:** The Resource Choreographer is a component of the ODIN platform that allows the end user to design operational workflows, connecting the available resources to perform a particular task. As an example, one could connect the output of a motion sensor to the input of an AI algorithm that performs human detection. It is important for the Resource Choreographer to be able to find the available resources in the Resource Directory, and to be able to link their inputs and outputs together, based on their semantic representations in the Resource Descriptors and the corresponding entities of the ODIN ontology. Using the semantic abstractions offered by the mappings to the ODIN ontology, the output of one resource can be linked to the input of another, if their types and semantics are compatible.

The above non-exhaustive list of services indicates the importance of the ODIN ontology in the ODIN platform, and the merits of semantic abstraction and harmonization to the discovery of resources and their usage in applications.

8 MAKING USE OF THE ONTOLOGY

8.1 IoT

In the IoT field, there exists several problems that the proposed ontologies can help ODIN to achieve its objectives:

- Perception of the data
- Processing the data
- Automation
- Reasoning & Learning
- Taking decisions

8.1.1 Perception and processing of data

There exists lots of IoT device vendors and platforms that offer devices and services with limited functionalities around the objective of the device or the platform they sell. Those devices and platforms usually have their own way to name the same functionality, property or definition. Not only talking about devices addressing different problems (measuring light, CO2 or taking pictures for example) but also when they are about the same domain. Some devices may expose interfaces to read the temperature, and the interface may deliver the measurements in a format and content that is very different from one another.

This problem is known as fragmentation, where the devices and platforms treat the same problem with different information and capabilities. Thus, using different types of devices or platforms becomes a very challenging problem in terms of integrating them in a platform like ODIN.

So, perception of data and processing are 2 problems that derive or at least are more difficult because of fragmentation.

8.1.2 Automation

With a common ontology, and creating the right mechanisms, ODIN can minimize those problems. Through the use of Thing Descriptors, ODIN can describe IoT resources with a common scheme and language. From ODIN's point of view, all the resources will be interoperable with the rest layers (robots, AI, management, etc). WP4 works around how to connect different resources to ODIN and how to expose them in a semantically interoperable way. The component in charge of providing that interoperability is the Thing Descriptor Module, described in D4.2 Implementation of local CPS-IoT RSM Features, Section 4 Resource Descriptor. It has been also discussed in Section 7. Using Web of Things, FIHR and other technologies such as OpenAPI, IoT resources will expose its attributes, services and information under WP3 T3.2 ontology.

Once the ontology and the resources, attributes and services are available in a digital twin representation, the rest of ODIN problems can be solved with more clear efforts.

Automation is a very broad task, but when it comes to the ODIN domain, it can be summarized in the following actions:

- Resource and Service Discovery: thanks to the ontology, it can automate the tasks to publish and discover what resources are connected to ODIN and what services are offered in a common language.
- Management: under Management there exists many tasks but applying setups, controlling access and starting and stopping resources or services, are the most important to name a few. Without IoT resources using the ontologies, the translation of management commands from ODIN to the devices, would be impossible.

8.1.3 Reasoning, Learning and Taking decisions

In terms of learning and taking decisions, the IoT device community usually cannot provide support by itself, but, with a common model and ontology, it can deliver this power to the upper layers of ODIN. For example, if ODIN defines several temperature rules, with some development, it could detect fires in places where there are no sensors or the fitted sensors are not working, but there are sensors nearby. In this case the ontology model would affect not only to the building with modelled dependencies, but the temperature sensors.

In Figure 49, the house of a monitored patient has been equipped with sensors with smoke detection and temperature sensors.

The house has been modelled with several rooms and in red are marked the sensors. This is the first floor of the house. On the second floor, which is not shown for brevity, there are also sensors but for unknown reasons are not working and that failure has not been detected.

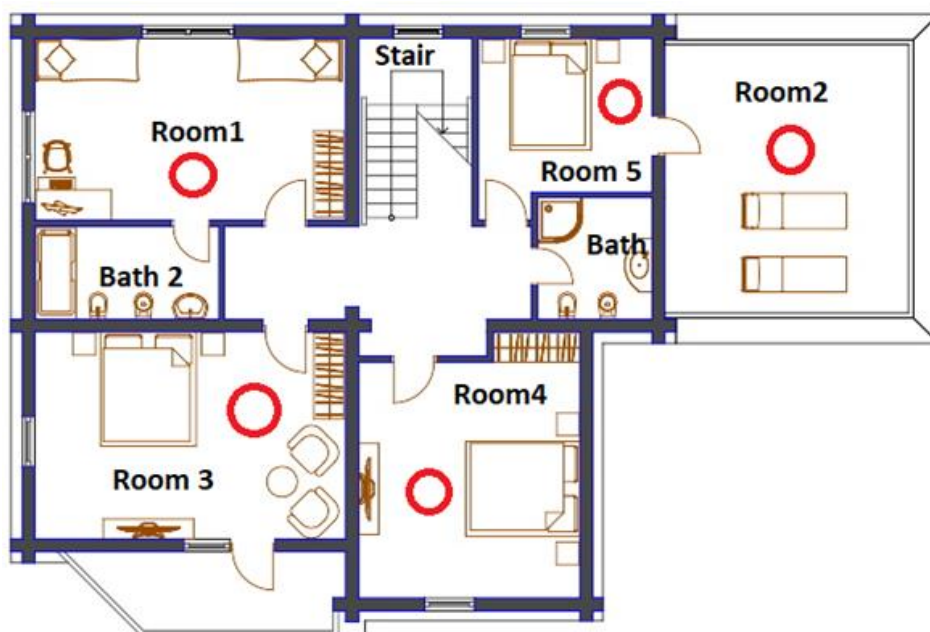


Figure 49: First floor of the house of a monitored patient

If a fire is started on the second floor, smoke will not be detected by the second-floor sensors, neither will the increase of temperature. As smoke travels upwards, the first floor sensors will not detect it, but temperature sensors will register an unusual temperature increase. The cognition layer of ODIN will be able to reason that no smoke is detected but nearby sensors are detecting higher than normal temperatures, therefore a fire is present on the second floor.

Another example, related to Real Time Location Systems (RTLS), can be derived from the same figure. In this case the red circles are gateway sensors that detect smart wrist bands using Bluetooth technology. The RTLS, that is in the IoT domain, would receive several signals, one per gateway, and perform a triangulation algorithm. This would calculate the most probable position of a person. The RTLS only computes coordinates (z, y, z) , in its most basic functionality. The first thing to use the information, would be transform the information using the ontologies selected for ODIN to have the information in ODIN's format.

The position (x, y, z) by itself is not useful as it is difficult to understand and does not allow reasoning. Using the tagged information of the room map with ontologies, and the calculated position from the RTLS, it can be inferred the room where the person is located. This is useful information. At this moment ODIN could relate the person with the room, which may offer services to be used by the person, or just monitor and record where the person has been. With this information ODIN could compute the time spent at each room, and for example, issue a “fire alert” if fire based parameters are violated.

To conclude with another example, in this case the item under observation could be a robot equipped with an RTLS tag. The robot may have its own location system, but when it comes to detecting the position of the robot, in big spaces, RTLS can provide more value in terms of speed. For example if the robot is unexpectedly moved from one position to another, the robot may not be able to track its position by itself. The robot will take a considerable amount of time to compute its new position, with or without ontologies.. With an RTLS, the robot would be located instantly by ODIN reporting the actual position to the robot, thus it would recover its autonomy. In addition, the position could be used to compute the best route to follow for the robot to achieve its next task. Or more importantly, ODIN could, based on the rules and parameters in place, decide whether that robot is the best unit to perform a task or assign another robot..

As it can be seen, at this level, where reasoning, learning and decisions are taken, the IoT cannot offer too much but to implement the ontologies so upper layers can do their work.

8.2 Robots

In this deliverable SSSA has jointly worked together with UPM and UoW to discuss the ontology structure and how to make it available for each of the robotic agents developed within WP5, described in D5.7 and employed in the ODIN project. Keeping the focus on the robotic framework, the ontology provides the representation of the surrounding environment the robotic agents must work into. Such representation must be considered dynamic and fast-scalable. Starting from meaningful static datasets the ontology can be adapted to the dynamic environment retrieving the information provided by the perception modules embedded within the different agents. The communication flow among robots and semantic ontology is bi-directional *i)* from the robotic agent to the ontology in order to provide information of the environment and related to both logistic aspect and environmental monitoring to dynamically update the ontology; and *ii)* from the ontology to the robot to enable the robot awareness and reasoning algorithms regarding the demanded task. Such communication will be performed throughout logic programming languages for sending queries and retrieving information in a deterministic way and will be implemented in the next months.

Partners are still working on the type of the information (*i.e.*, images, cloud points, mission details) to share accordingly to the needs of the use cases of the ODIN project but keeping a general approach.

Moreover, all the robotic technologies developed within WP5 uses two communication protocols:

- i) a client-service and/or publisher-listener infrastructure provided by the Robotic Operating System (ROS) framework;
- ii) publisher-listener infrastructure throughout Message Queuing Telemetry Transport (MQTT – standard ISO protocol) implemented using the Particle microcontrollers embedded into the robotic platform.

MQTT offers several advantages in respect to a general HTTP service which is unidirectional, and synchronous. Assuming that the other components of the ODIN project rely instead on a KAFKA communication protocol a custom “connector” will need to be developed to create a bridge between the aforementioned communication protocols that will handle the translation.

Figure 50 shows the whole architecture of the ODIN project with a special focus (red box) on the robotic part and its connection with the Resource Gateway (Enterprise Service Bus - ESB) that bridges all the single modules developed within the ODIN project (*i.e.*, Platform services, Resource management, Semantic Interoperability).

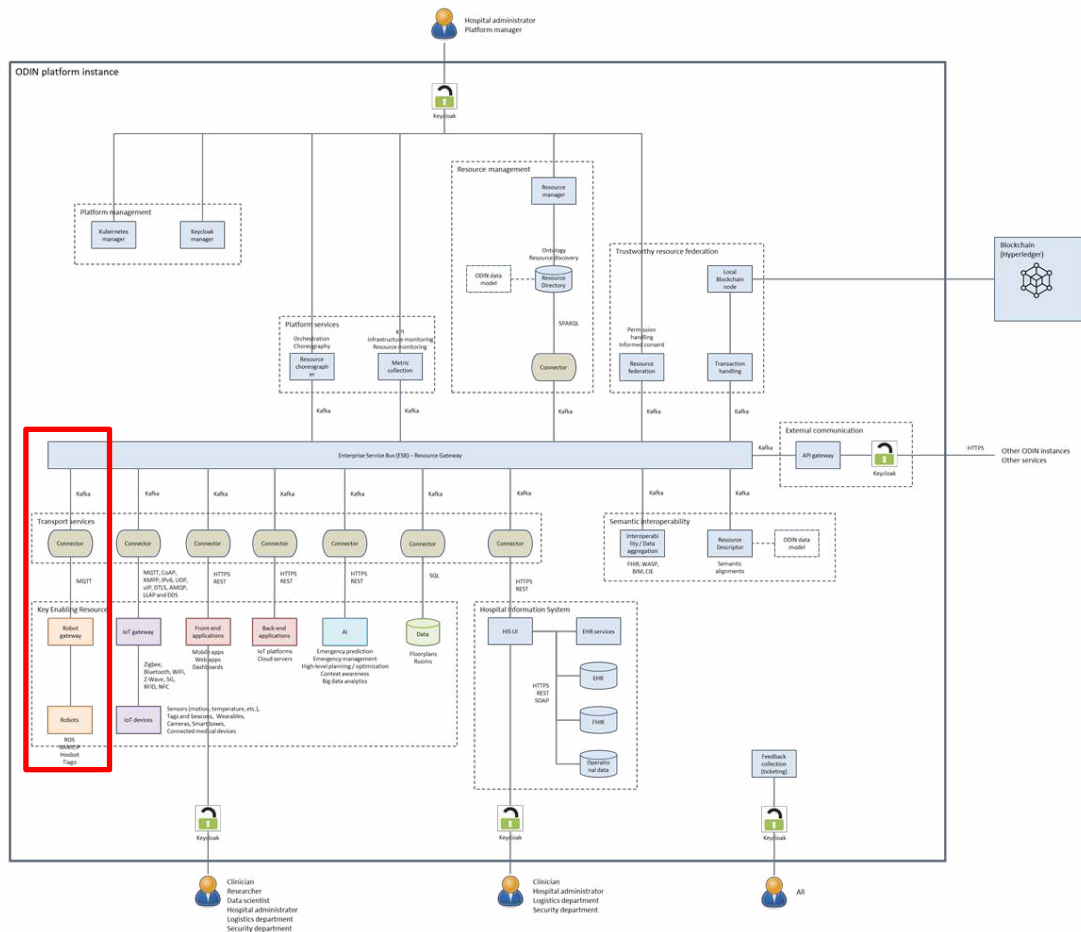


Figure 50 - ODIN architecture

8.3 AI

8.3.1 Uses of ontology for AI

The major benefit of ontologies in the field of AI is that they provide a general structure of the knowledge encompassed in the project. They can be used as master data management systems that consider elements that are not present in the data itself, such as processes, relationships, workflows. Understanding the relationships between the different concepts allows a more precise identification of solutions to problems and provides knowledge of how the products and devices of the project are related.

All this information can be used as input data for AI models so that complex systems are used as a source of knowledge rather than isolated data. Otherwise the ontology is very useful to explain in a human-understandable way the complex systems and projects that are interconnected in the ODIN project.

In this way ontology brings value to both humans and machines. For humans it simplifies and makes data exploration more coherent and easier, and for machines it broadens the scope of the data and increases its quality for training datasets.

8.3.2 Ontology enrichment with AI

The use of Machine Learning to build Ontologies or a subset of Ontologies has been common in the recent years. There are some examples of artificial intelligence models that are based on ontologies that are able to find hidden relationships in the knowledge.

A simpler way to improve the knowledge collected by the proposed ontology is to add abstract content related to AI models. Adding new classes and relations that include algorithms in the ontology can complete the knowledge graph with non-touchable things that are also part of the project ecosystem.

Classes:

- Algorithms: A set of rules that precisely defines a sequence of operations, which would include all computer programs, including programs that do not perform numeric calculations.
- Applications: the action of putting something into operation.
- Dependencies: What are the prerequisites for solving an algorithm or problem and what are the requirements for a tool to be executed.

Relations:

- Input: Features
- Output: support to decision, predictions, classification, segmentation

9 CASE STUDY: HOSPITAL CLINICO SAN CARLOS

On 13-14 October 2021, a T3.2 technical meeting was held in Rome, organized by UoW (task leader) and hosted by UCBM, to discuss about the opportunity of designing the first version of the ontology. This would focus on specific pilots and use cases, in order to test on actual data. At the end of the meeting, it was agreed to focus on Use Case 2 (UC2), Clinical Engineering. It was also agreed to focus on the Spanish pilot (SERMAS). The ontology V1 is therefore focused on the Hospital Clínico San Carlos.

On November 23 there was a technical visit to the hospital in Madrid. It was agreed to start focusing on two specific departments: the Haemodynamics Unit and the Emergency Room.

9.1 DESCRIPTION OF THE PILOT

SERMAS-Servicio Madrileño de Salud is the largest agency in the Spanish National Health Service and is in charge of the Madrid Regional Health System's public health services. SERMAS represents Hospital Clínico San Carlos HCSC) in the ODIN project. The San Carlos Clinical Hospital consists of seven institutes:

- "José Botella Llusia" Women's Health Institute
- Institute of Laboratory Medicine
- Institute for Childhood and Adolescence
- Institute of Psychiatry and Mental Health
- Institute of Neuroscience
- Institute of Oncology
- Cardiovascular Institute

It is a university hospital that provides medical, surgical, and central services, and also includes areas of nursing units [45].

9.2 DATASETS

The hospital website was primarily used to describe the pilot, followed by the maps obtained from the pilot, as well as the inventory of supplies and instrumentation. Meanwhile, images of the building where interventional haemodynamics or cardiology is located were obtained during the technical visit to the site, which helped to approximately describe the areas.

9.2.1 PLANS

The Haemodynamic Unit and Emergency Room maps have been supplied. The rooms have been numbered to make them easier to find, as seen in Figures 51, 52, and 53.

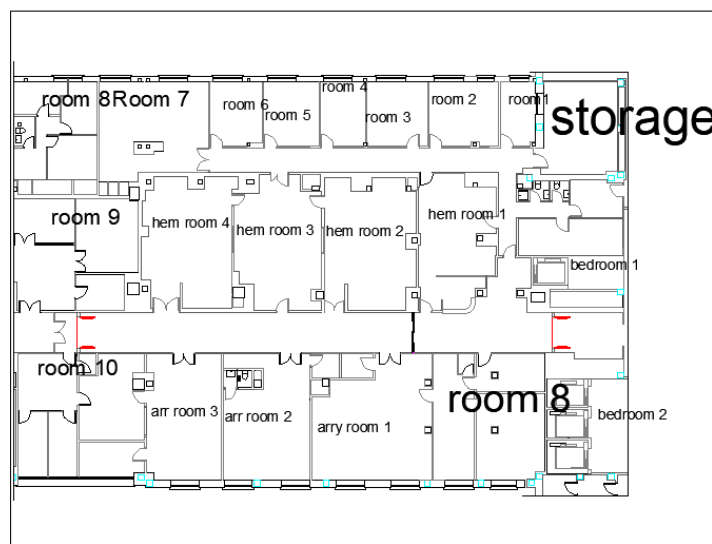


Figure 50: Haemodynamic Map

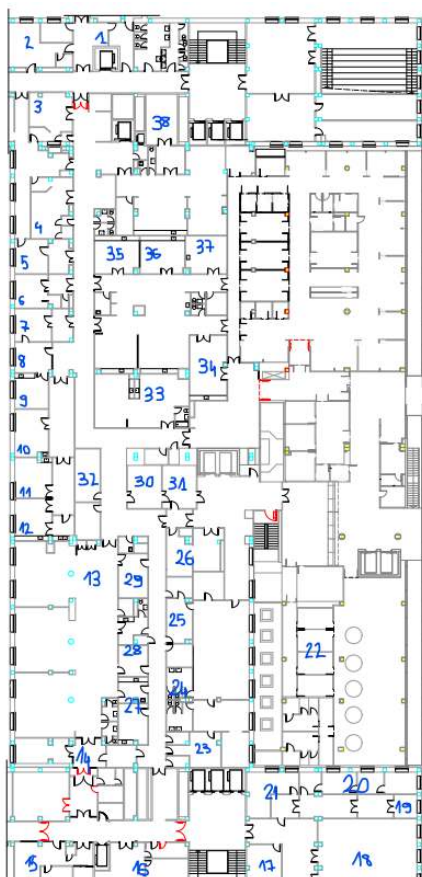


Figure 51: Emergency Map 1

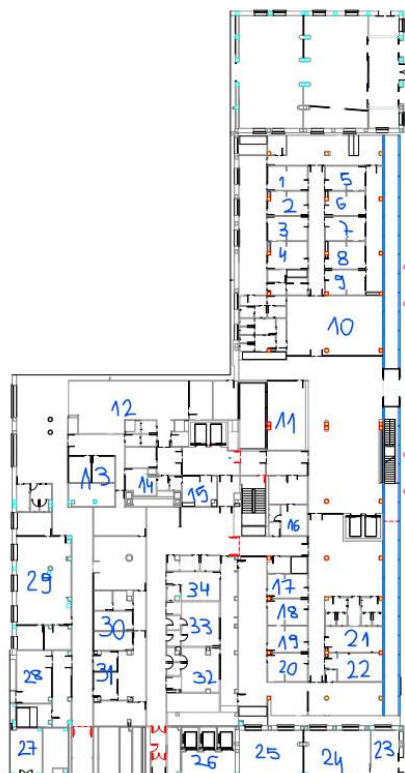


Figure 52: Emergency Map 2

9.2.2 EQUIPMENT AND CONSUMABLES INVENTORY

Some Excel files containing the inventory for equipment and consumables were received from the pilot (see Figures 54 and 55). These files contain a list of medical and electrical devices for which an EMDN code was assigned in order to categorise the device in the ODIN ontology. The Cellfile plugin in Protégé has been used to import the data. Because the devices are stated to belong to the arrhythmia, hemodynamic, or emergency units in these files, they have been defined in ODIN in the same way.

A	B	C	D	E	F	G
Código	N. Expediente	Equipo	Marca	Modelo	Número Serie	Unidad
05488	1999-6-001.1	ELECTROCARDIOGRAFO	HEWLETT PACKARD	PAGEWRITER 100 M1772A	CNC4229064	URGENCIAS
06554	2003-6-009	PULSIOXIMETRO	DATEX OHMEDA	3800 PLUS	FBXG00371	URGENCIAS
07202	1997-6-033	COPIADORA PLACAS	AGFA	DRYSTAR 2000	9804/2334	URGENCIAS
07203	1997-6-033	COPIADORA PLACAS	AGFA	DRYSTAR 2000	9804/2333	URGENCIAS
07261		ESFIGMOMANOMETRO				URGENCIAS
07265		ASPIRADOR SECRECIONES	ORDISI		15368	URGENCIAS
07737	1998-6-796	OTOFTALMOSCOPIO	RIESTER	RI-FORMER	10547	URGENCIAS
07777		ESFIGMOMANOMETRO				URGENCIAS
07873		ESFIGMOMANOMETRO				URGENCIAS
07874		ESFIGMOMANOMETRO				URGENCIAS
07877		NEBULIZADOR	CARBURROS METALICOS	2002HF	HF584661	URGENCIAS
07878		KIT OTORRINO	HEINE			URGENCIAS
07881		BASCULA TALLIMETRO				URGENCIAS
07910		BASCULA PESA BEBE	SECA	SP	161064	URGENCIAS

Figure 53: Consumables and Equipment data

A	B	C	D	E	F	G
Código	N. Expediente	Equipo	Marca	Modelo	Número Serie	Unidad
110797	CESION	ECOGRAFO	SIEMENS	X 300	345967	UNIDAD DE ARRITMIAS
111826	CESION	navegador cardio electromagnetico	ABBOT	EE3000	13643158	UNIDAD DE ARRITMIAS
82965	2005-8-056	MONITOR P.N.I.	GE HEALTHCARE	DINAMAP PROCARE 300 NELLCOR	AAW05360171SA	UNIDAD DE ARRITMIAS
92386	2006-1-055	RX. SALA HEMODINAMICA	PHILIPS	ALLURA XPER FD 10'		UNIDAD DE ARRITMIAS
105065	2013-1-073	FOTOCOPIADORA	RICOH	MP301SPF	W913PA01548	UNIDAD DE ARRITMIAS
110583	CESION	ACCESORIO SISTEMA DE NAVEGACION 3D	BOSTON SCIENTIFIC		FG0-S00565	UNIDAD DE ARRITMIAS
110584	CESION	SISTEMA DE NAVEGACION 3D	BOSTON SCIENTIFIC	RYTHMIA	S00582	UNIDAD DE ARRITMIAS
112602	2018-8-00104	FOTOCOPIADORA	RICOH	MP 3555	C368J600117	UNIDAD DE ARRITMIAS
115388	CESION	SISTEMA DE NAVEGACION	BIOSENSE WEBSTER	CARTO 3 V6	11392	UNIDAD DE ARRITMIAS
117594	CESION	SISTEMA DE ANESTESIA COMPLETO	GENERAL ELECTRIC	CARESTATION	SM72013002SNA	UNIDAD DE ARRITMIAS

Figure 54: Consumables and Equipment data

9.2.3 ORGANIZATION CHART

The organization chart for San Carlos Clinical Hospital was inferred from the official HCSC website. Unfortunately, a complete list of all personnel of the haemodynamics and emergency departments could not be found, hence the site only lists the members in charge of these units. As a result, the heads of the departments and units involved were added to the ODIN ontology. The information gathered is depicted in Figure 56, Figure 57, Figure 58.

Hospital Clínico San Carlos | CIUDADANOS | PROFESIONALES | COMUNICACIÓN | NOSOTROS

Profesionales > Servicios médicos > Cardiología

Secciones

- Servicio
- Paciente
- Asistencia
- Formación
- Investigación
- Profesionales
- Contacto

Cardiología

Compártelo en [d](#) [f](#) [t](#) [e](#) [s](#)

Servicio

El servicio de Cardiología del Hospital Clínico procede de la unión de los servicios de Cardiología y Cardiología Intervencionista, que se integraron en el Instituto Cardiovascular en el momento de su constitución, en mayo de 1998, y se convirtieron en un único servicio en junio de 2003.

Por otro lado, en el año 2010 se integra funcionalmente también en el Instituto Cardiovascular el servicio de Cardiología del Hospital Carlos III, complementándose la actividad asistencial, docente e investigadora de ambos centros, en el marco de un único equipo de trabajo.

Localización servicio
Hospitalización: Segunda Norte y Segunda Sur

Figure 55: Cardiology Department description

Hospital Clínico San Carlos | CIUDADANOS | PROFESIONALES | COMUNICACIÓN | NOSOTROS

Profesionales > Servicios médicos > Cardiología

Secciones

- Servicio
- Paciente
- Asistencia
- Formación
- Investigación
- Profesionales
- Contacto

Localización servicio
Hospitalización: Segunda Norte y Segunda Sur

Descripción

Organización

El Servicio, dentro de la estructura del Instituto Cardiovascular se estructura por Unidades:

- Cardiología Clínica
- Unidad de Cuidados Agudos Cardiológicos
- Electrofisiología y Arritmias
- Hemodinámica y Cardiología Intervencionista
- Imagen Cardiovascular
- Rehabilitación Cardíaca
- Investigación Cardiovascular
- Área de Enfermería

Paciente

Figure 56: Cardiology Department units



Figure 57: Head of Hemodynamics Unit

9.3 MAPPING THE PREMISES IN THE ODIN ONTOLOGY

General instructions for transforming a dataset into a correct ontological version have been defined in section 6.3. Then, as has been described in section 6.4, there are different approaches to connect data. The following subsections contains details on how the Haemodynamics Unit and the Emergency Department has been mapped.

9.3.1 Phase 1: Properties Definition

The object and data properties are required to indicate how objects can interact with one another, as well as the attributes and parameters that they can have. The properties defined in ODIN that have been used for the UC2 are described in the following subsections, the created object properties proved necessary to describe mainly robots, the hospital facility and the facility organization. Instead the data properties declared are important for the description of the robots.

9.3.2 Phase 1.1: Object Properties

The properties of Table 8, which include their description as well as their domain and range, have been defined to connect the class instances.

Table 8: Table of Odin's Ontology Object Property

Object Property	Domain	Range	Annotations
odin:carries	Robot	EMDN Medical Device	Relationship between any robot that carries an object from a building or a space to another.

odin:carries from	Robot	Space	Relationship between any robot that carries a device from a space to another one.
odin:delivers to	Robot	Space Building	Relationship between any robot that delivers an object from a building or a space to another one.
odin:has robot	Building Storey Space	Robot	Relationship between any building or storey or space that uses a robot.
odin:has technical specifications	Robot	Autonomy some xsd:decimal Has_manufacturer some rdfs:Literal Communication some rdfs:Literal Payload some xsd:decimal Battery some rdfs:Literal Hard drive storage capacity some xsd:decimal Temperature range some xsd:decimal Connectivity some rdfs:Literal Dimensions some xsd:decimal Speed some xsd:decimal Environment some rdfs:Literal Controller some rdfs:Literal Processor some rdfs:Literal Reachable height some xsd:decimal Weight some xsd:decimal	Relationship between a robot and its technical specifications.

		Elevation range some xsd:decimal	
		RAM some xsd:decimal	
odinemdn:has medical device	Organization Unit Hospital Facility	EMDN Medical Device	Relationship between any building or space and a medical device.
odin:adjacent space	Space	Space	Relationship between two adjacent spaces.
odin:has facing space	Space	Space	Relationship between two spaces one in front of the other.
bot:has building	Sites of Care Delivery	Building	Relationship to buildings contained in a zone.
bot:has element	Hospital Facility	Building Element	Links a zone to a building element.
bot:adjacent element	Building Space	Building Element	Relation between a building or a space and its adjacent element bounding the space.
bot:contains element	Building Space	Building Element	Relation to a building element contained in a building or space.
bot:intersecting element	Building Space	Building Element	A building element that intersects a space.
bot:has storey	Building	Storey	Relationship between a building and a storey.
bot:has space	Building Storey	Space	Relation to spaces contained in a zone.
wot:is measured in	Measurement	Unit of Measure	A relationship identifying the unit of measure of an entity.
org:has unit	Organization	Organization Unit	Indicated a unit which is part of this organization.
org:linked to	Organization	Sites of Care Delivery	Relationship between two organizations.
org:member of	Occupation	Organization	A person that is part of the organization.
org:head of	Occupation	Organization	A person that is head of the organization.

odin:worksIn	Organization Unit	Space	A person that works in a space.
---------------------	-------------------	-------	---------------------------------

9.3.3 Phase 1.2: Data Properties

The data properties have been defined in this ODIN ontology V1 with the goal of describing the technical specifications of the robots, as they indicate a relationship between an individual and a type of data. The data properties utilized are listed in Table 9.

Table 9: Odin Data Properties

Data Property	Domain	Range
arm reach	Is measured in some mm	xsd:decimal
autonomy	Is measured in some h	xsd:decimal
battery	Is measured in some V	rdfs:Literal
communication	Device	rdfs:Literal
connectivity	Device	rdfs:Literal
controller	Device	rdfs:Literal
dimensions	Is measured in some mm	xsd:decimal
elevation range	Is measured in some mm	xsd:decimal
environment	Device	rdfs:Literal
hard drive storage	Is measured in some Gb	xsd:decimal
has manufacturer	Device	rdfs:Literal
has model name	Device	rdfs:Literal
payload	Is measured in some kg	xsd:decimal
processor	Device	rdfs:Literal
RAM	Is measured in some Gb	xsd:decimal
speed	Is measured in some m/s	xsd:decimal
temperature range	Is measured in some °C	xsd:decimal

weight	Is measured in some kg	xsd:decimal
---------------	---------------------------	-------------

9.3.4 Phase 2: Individuals Definition

Individuals or instances of the classes are the central focus of this first edition of the ODIN ontology. It is possible to determine several situations or use cases using them. More than 600 individuals have been identified to achieve the UC2 aim, which can be divided into four main categories:

- Instances of the **sumo-cora:Device** class

Many individuals belongs to the sumo-cora:Device. Different instances such as printers, cameras, lights and other electronic devices are part of the sumo-cora:Electric Device class.

Individuals such as odin:Catheter Mount, an instance of the odin:R020201 Fixed Catheter Mounts class, or odin:Defibrillator, an instance of the odin:Z12030503 Automatic Defibrillators class, can be found in the class odin:EMDN Medical Device.

Three individuals are defined in the class cora-bare:Robot: odin:RB1 Base Mobile Robot, odin:RB1 Mobile Manipulator, and odin:Tiago robot. These are the robots in charge of transporting medical devices, for UC2 from the hemodynamics storage to various rooms.

- Instances of the **Thesaurus:Site of Care Delivery** class

In this hierarchical system, the individuals illustrated in the Figure 59 and Figure 60 can be located. Figure 40 is an example of the numerous individuals defined under the class bot:Space, which describes the hospital's entire Emergency and Haemodynamics departments.

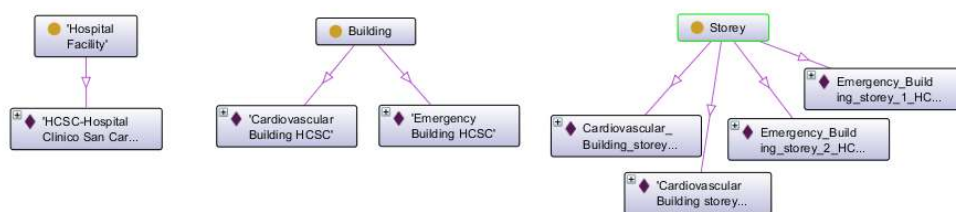


Figure 58: Site of Care delivery class individuals

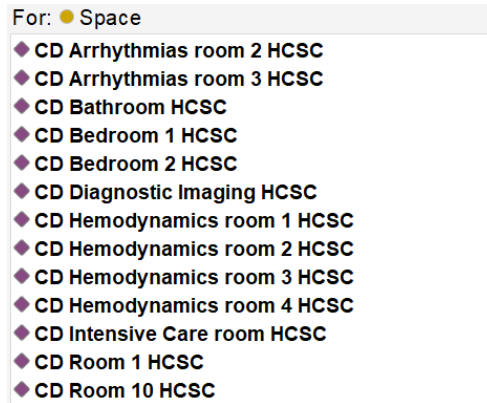


Figure 59: Site of Care delivery class instances

- Instances of the **obo:Occupation** class

The directors of the Hospital Clinico San Carlos' Haemodynamics and Emergency departments are classified as members of this class.

- Instances of the **org:Organization** class

For this version of ODIN there are four individuals here: HCSC Emergency Department, HCSC Arrhythmias Unit, HCSC CD-Cardiology Department, HCSC Haemodynamics Unit.

HCSC Emergency Department is an instance of the class Accident and Emergency Department, while the others are members of Medical Department.

9.3.5 Phase 3: Ontology Development

Users may define the structure of a hospital by introducing the storeys and spaces that make it up. Medical or electrical equipment can be placed and their properties determined, within each space. Medical devices are then utilized by employees to perform certain tasks, such as a surgery that can be declared using the Procedures class. Again, AI allows for the combination of classes that can be used as input to an algorithm. Furthermore, the departments that are present, as well as the employees that are a part of them, might be declared from an organisational point of view. This is one potential use, but the ontology for the ODIN project has been mapped using individuals.

First of all, it was stated that the San Carlos Clinical Hospital has two buildings, Figure 61. The building plans were then created, Figure 62.

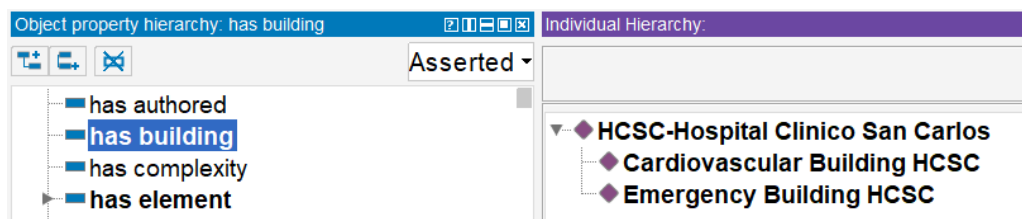


Figure 60: Map of HCSC with the property has building

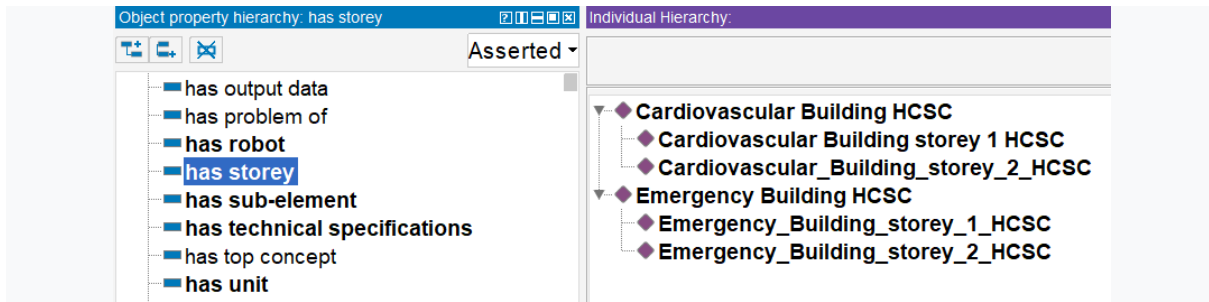


Figure 61: Storeys of the buildings declared with the property has storey

9.3.6 CARDIOLOGY DEPARTMENT

The rooms that make up the floors have been stated with the property bot: has space , Figure 63, after the Cardiology building and its floors have been defined. Later, the characteristics bot:adjacent space, Figure 64, and odin:has facing space, Figure 65, were used to connect these rooms. These signify when two rooms are adjacent to one another, that is, when they share an element such as a wall, and when two rooms are facing one another. Another object property useful for the scope of the ontology is the bot:contains element, Figure 66. This is how the Haemodynamics unit's storey has been defined.

The property odinemdn:has medical device introduces the medical devices that are located in the Department, in according to the equipment and consumable inventory. This can be observed in Figure 67 and Figure 68.

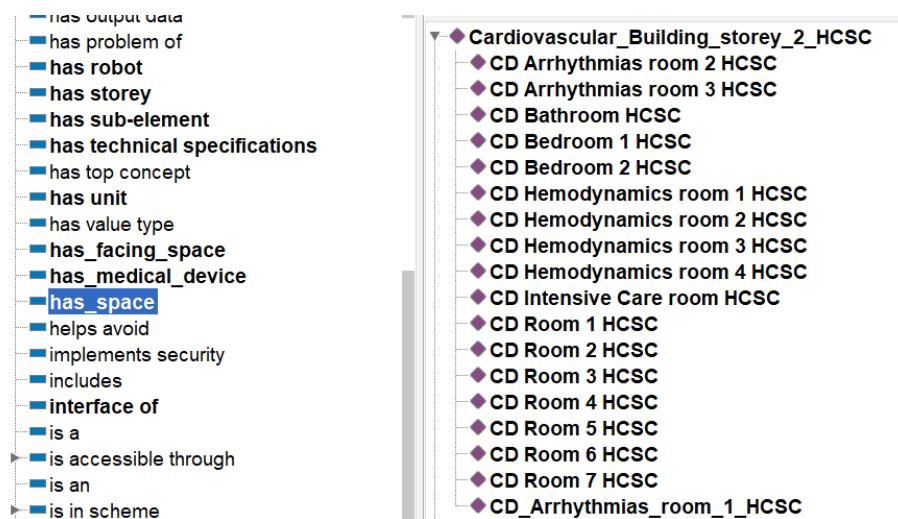


Figure 62: Spaces of the Haemodynamics defined with the property has space

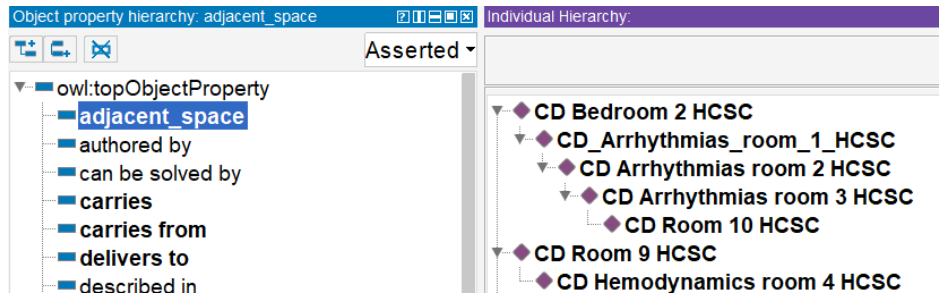


Figure 63: Adjacent spaces in Haemodynamics

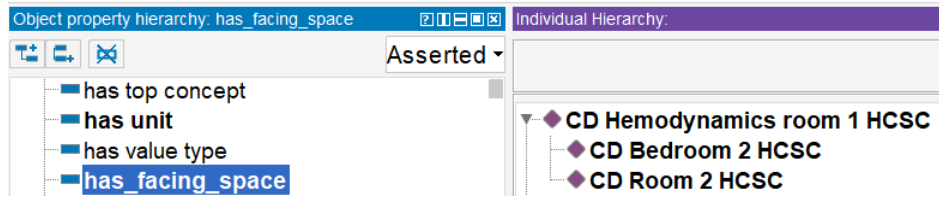


Figure 64: Facing spaces in Haemodynamics

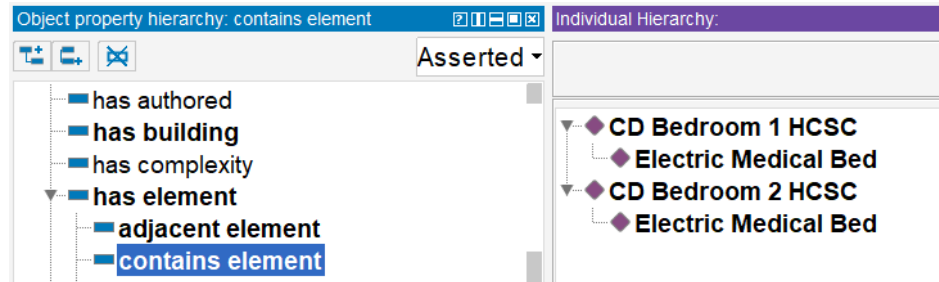


Figure 65: Element contained in some spaces of Haemodynamics

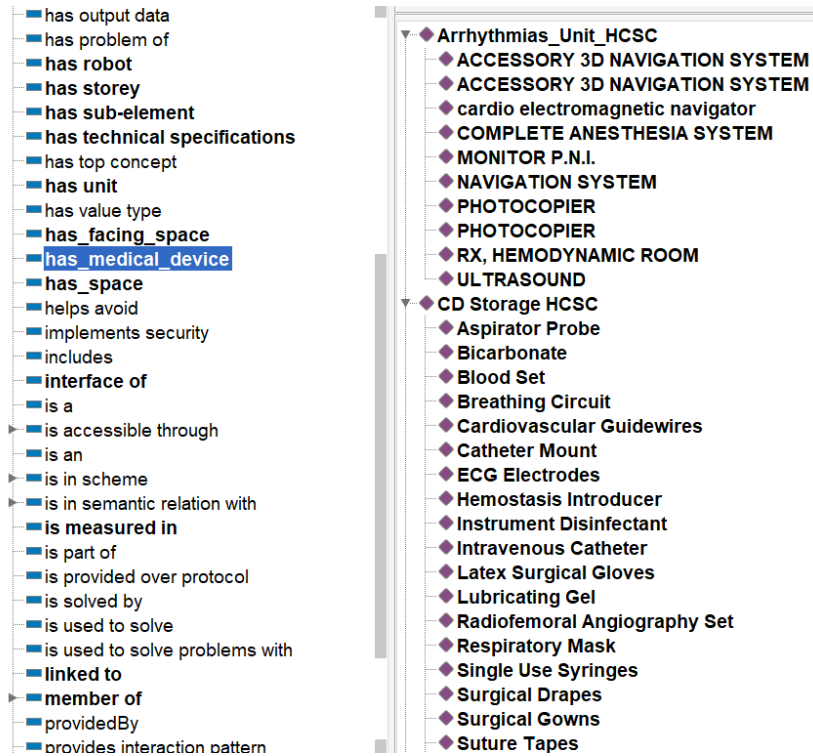


Figure 66: Medical devices located in the Cardiology Department

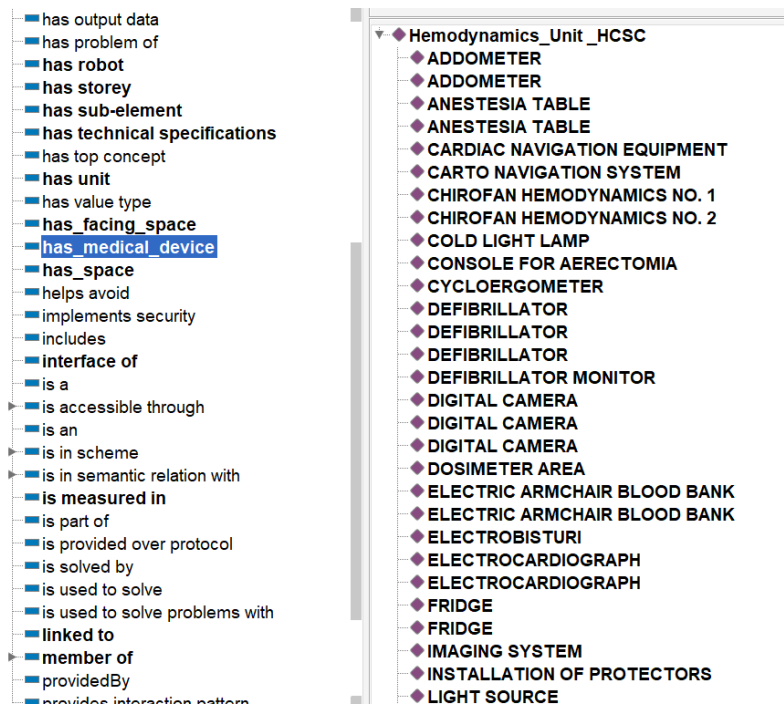


Figure 67: Part of the medical devices in the Haemodynamics Unit

9.3.7 EMERGENCY ROOM

In the same way that the Haemodynamics and the Cardiology department were mapped, we repeated for the Emergency department. The following Figure 69, Figure 70, demonstrate the spaces that are part of the building. Figure 71 and Figure 72 shows the adjacent spaces and Figure 73, Figure 74 the facing spaces. Then the medical devices that are located as shown in Figure 75.

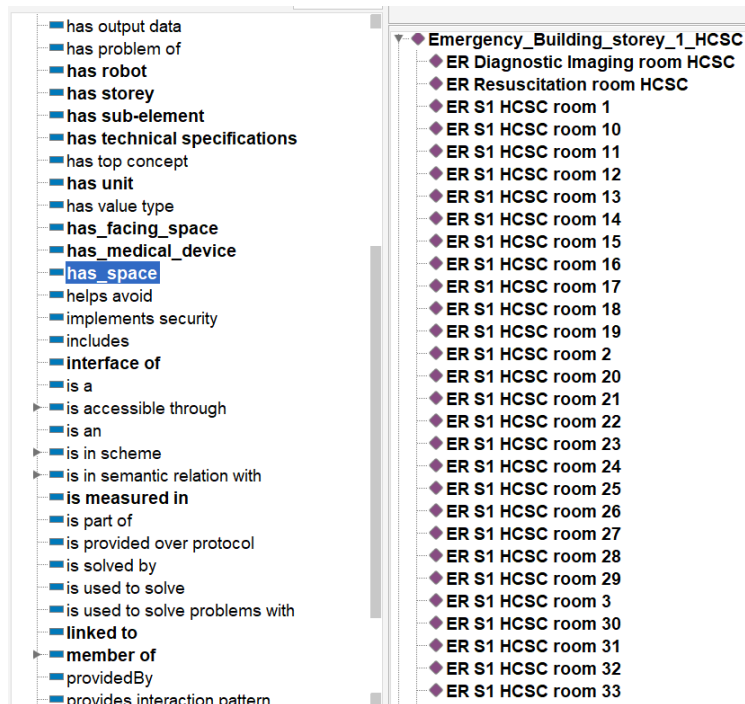


Figure 68: Spaces of the first storey of Emergency

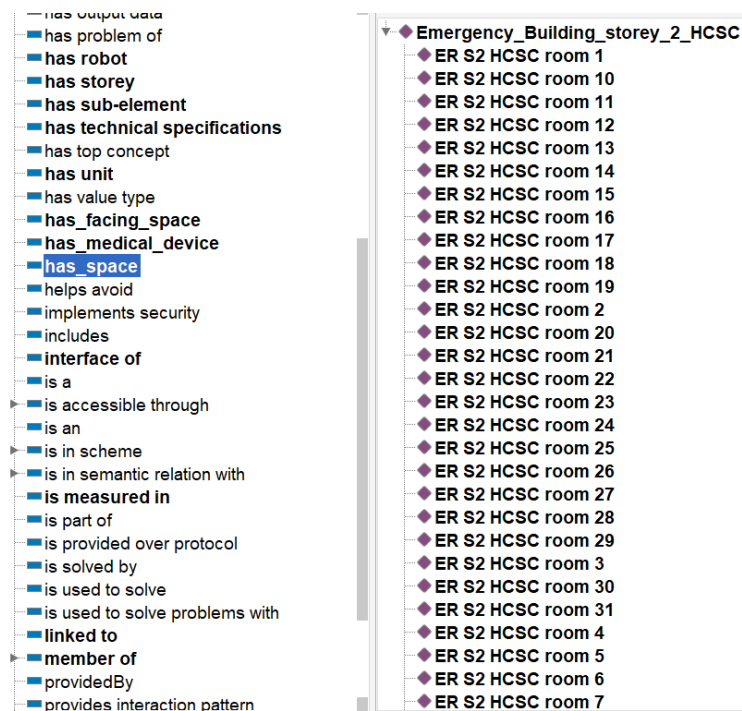


Figure 69: Spaces of the second storey of Emergency

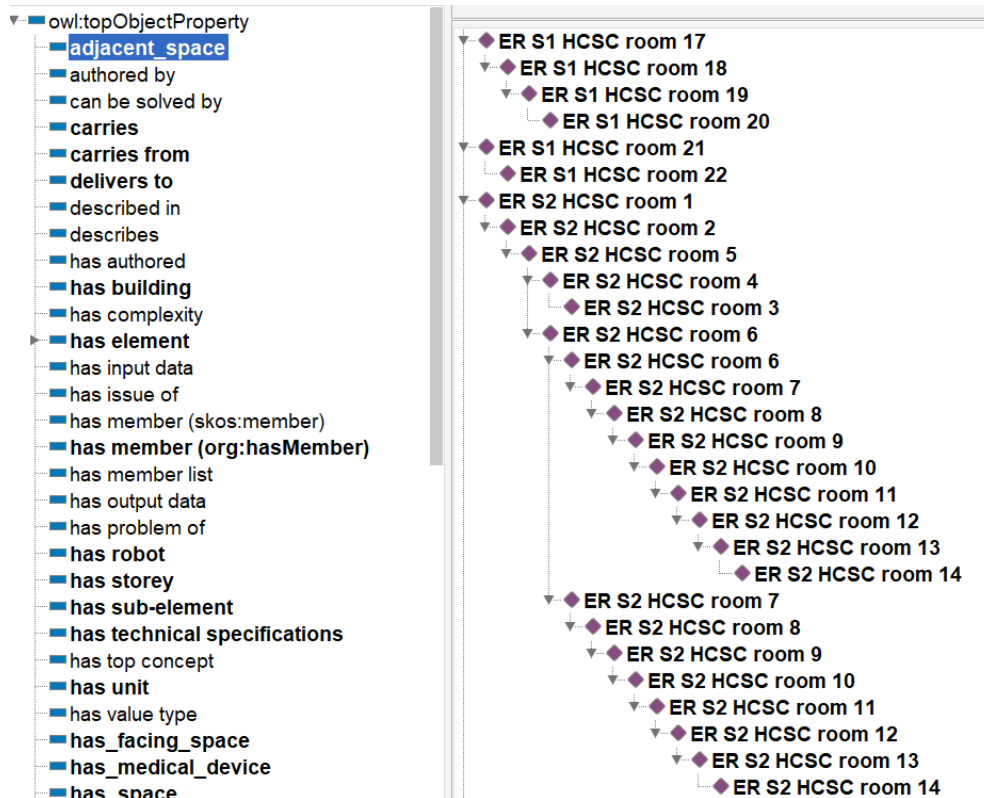


Figure 70: Adjacent spaces in Emergency

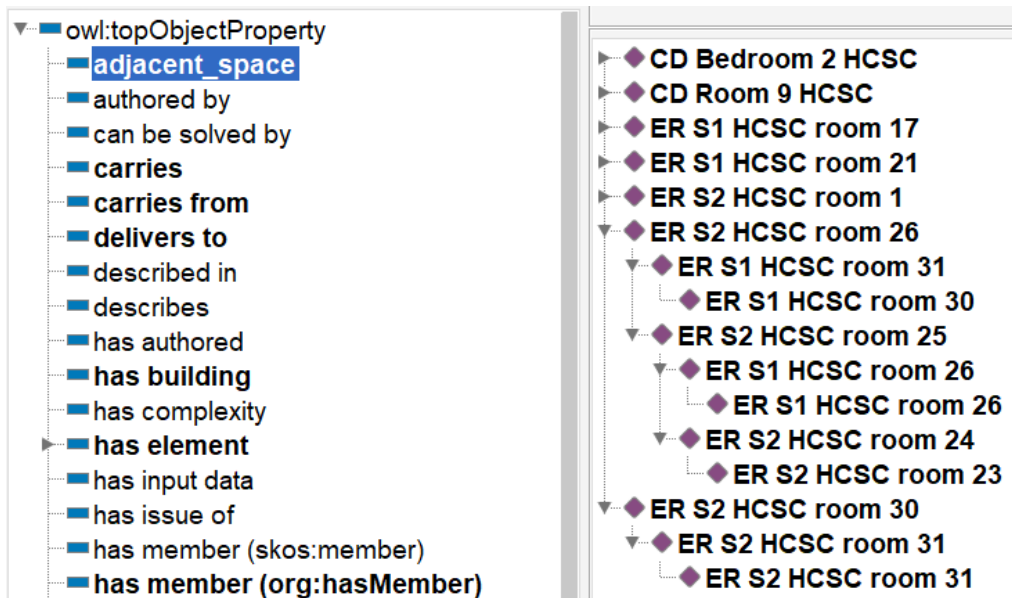


Figure 71: Adjacent spaces in Emergency 2

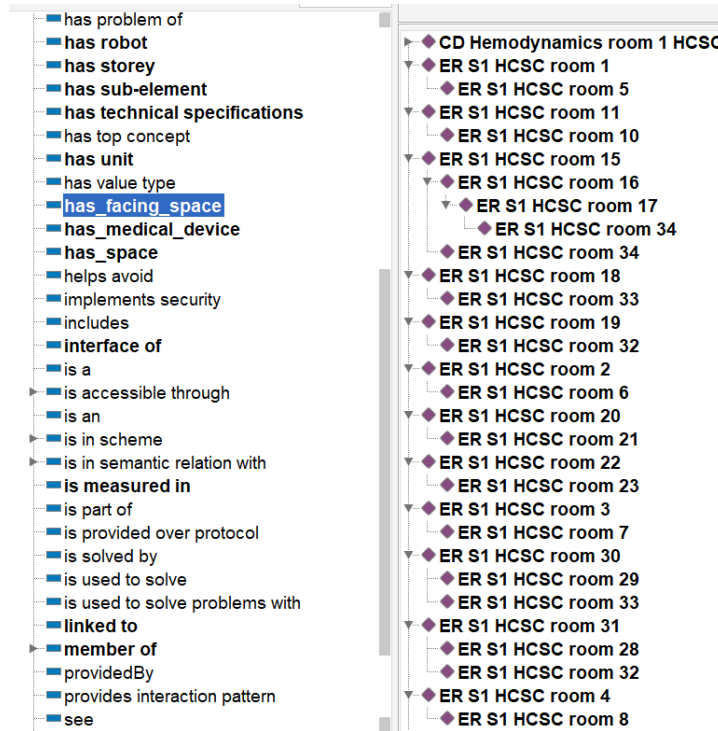


Figure 72: Facing spaces in Emergency

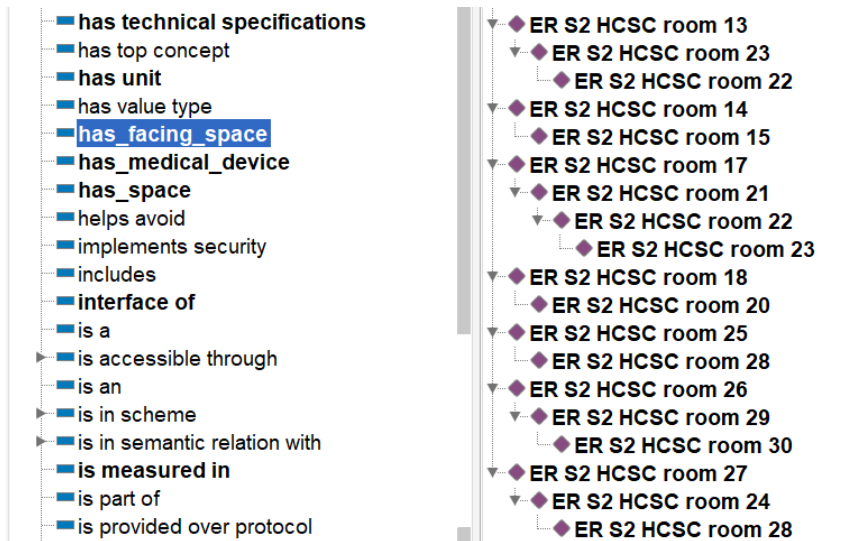


Figure 73: Facing spaces in Emergency 2

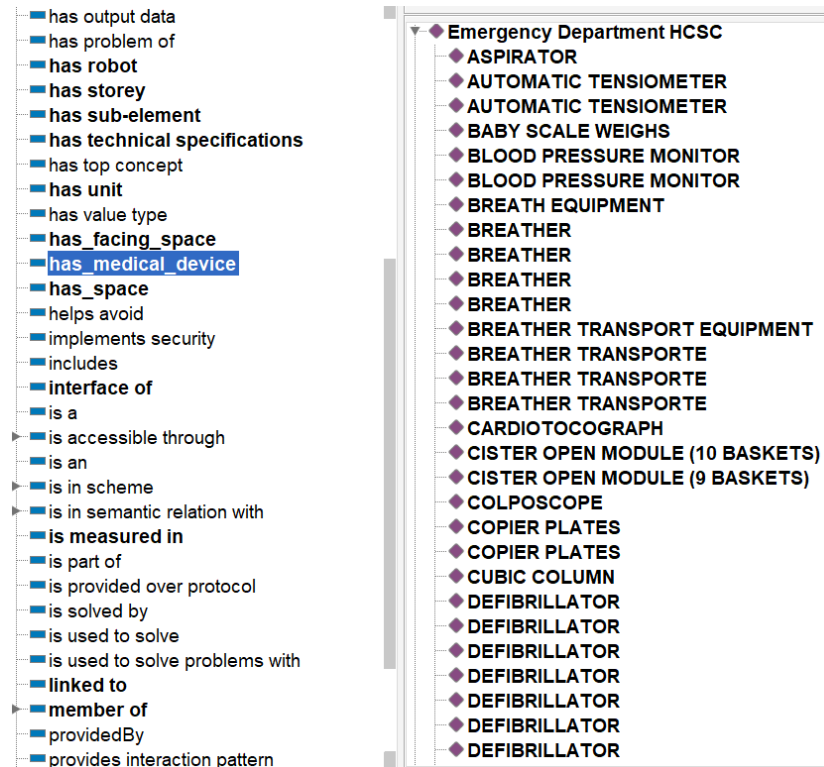


Figure 74: Part of the medical devices located in Emergency

9.4 MAPPING THE ROBOTS AND THEIR TASKS IN THE ODIN ONTOLOGY

The path taken by the robots must be mapped, which is done using object properties:

- **odin:carries**, Figure 76, feature that describes a robot's ability to carry stuff.
- **odin:carries from**, Figure 77, feature that describes a robot's ability to transport goods from one location or building to another.
- **odin:delivers to**, Figure 78, the ability of a robot to bring objects to a location or a building is expressed by this attribute.

Their technical specifications are shown in Figure 79 and Figure 80.

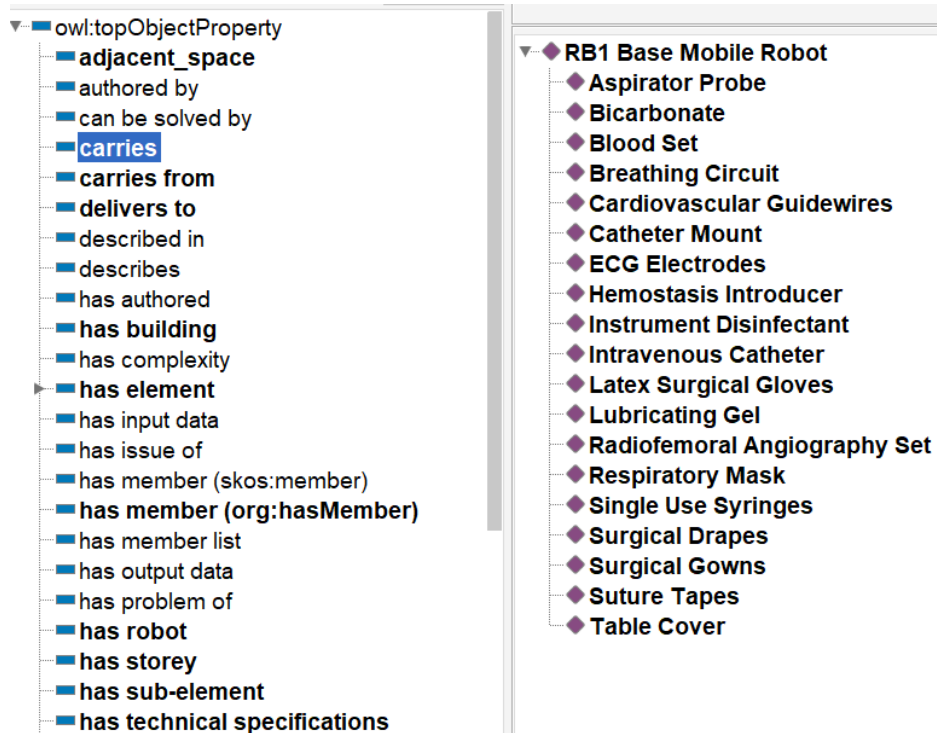


Figure 75: Introduction to devices that the robot carries



Figure 76: A robot carries devices from the CD Storage

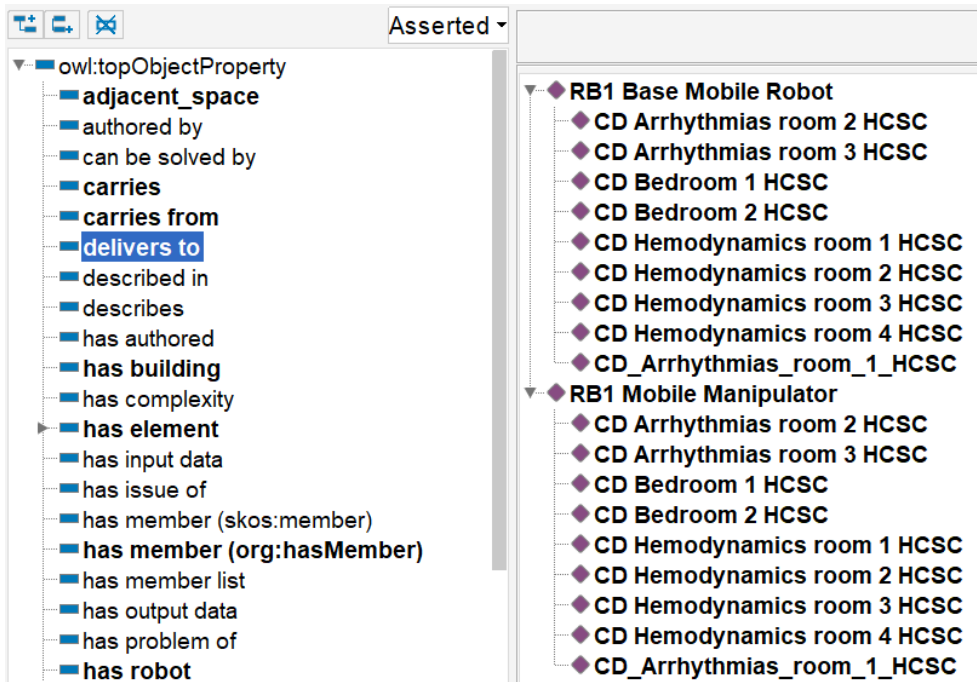


Figure 77: Spaces where the Robot delivers

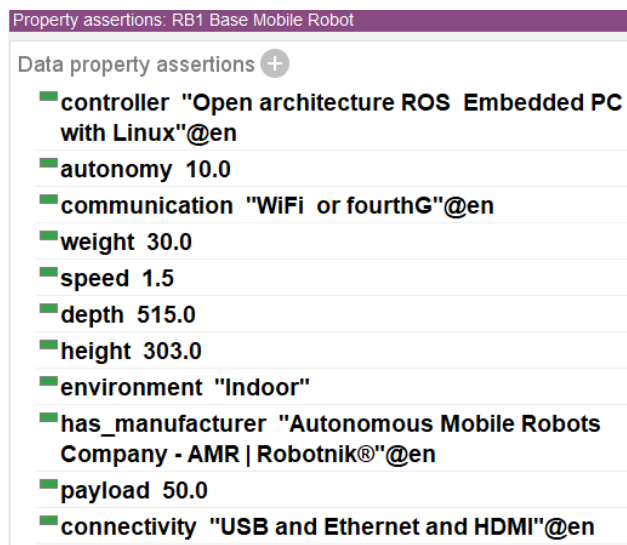


Figure 78: Technical specifications of the Robot

Description: Robot

SubClass Of +

- 'has technical specifications' some ('arm reach' some xsd:decimal)
- 'has technical specifications' some ('elevation range' some xsd:decimal)
- 'has technical specifications' some ('hard drive storage capacity' some xsd:decimal)
- 'has technical specifications' some ('reachable height' some xsd:decimal)
- 'has technical specifications' some ('temperature range' some xsd:decimal)
- 'has technical specifications' some (autonomy some xsd:decimal)
- 'has technical specifications' some (battery some rdfs:Literal)
- 'has technical specifications' some (communication some rdfs:Literal)
- 'has technical specifications' some (connectivity some rdfs:Literal)
- 'has technical specifications' some (controller some rdfs:Literal)
- 'has technical specifications' some (dimensions some xsd:decimal)
- 'has technical specifications' some (environment some rdfs:Literal)
- 'has technical specifications' some (has_manufacturer some rdfs:Literal)
- 'has technical specifications' some (payload some xsd:decimal)
- 'has technical specifications' some (processor some rdfs:Literal)
- 'has technical specifications' some (RAM some xsd:decimal)
- 'has technical specifications' some (speed some xsd:decimal)
- 'has technical specifications' some (weight some xsd:decimal)
- ('carries from' some Space) and ('delivers to' some Space)
- carries some 'EMDN Medical Device'
- Device

Figure 79: Technical specifications of the Robot class

10 CONCLUSIONS

This deliverable illustrates the first nine months of activity in Task 3.2, which started at M3.

After an analysis of the state of the art in terms of existing literature about semantic technology applied to health care settings, a set of existing ontologies has been identified as the groundings to design the ODIN ontology V1. In one case, namely for medical devices, no satisfying ontology could be found, and it was decided to create from scratch an ontology representing the totality of classes of medical devices, as described in the European Medical Devices Nomenclature (EMDN), recently released as a standard nomenclature from the European Commission.

Many new classes and properties were defined to link the many entities that constitute a hospital, from buildings to personnel, from diseases to treatments and equipment, and many more.

Specific qualifying ODIN technologies, such as IoT, Robots and AI devices have also been mapped in this first version of the ODIN ontology. The ontology has been applied to a specific case study, both for proving its efficacy and for refining it.

In the second version of the ontology, which is due ad M24, many other case studies will be mapped, thanks to the new definitions and key performance indicators (KPI) coming from the deliverable 7.1, for each reference use case, as well as concepts identified in the ODIN platform and technologies, some of which have already been discussed, which current V1 does not map.

There are also 2 more topics that will be treated in the future:

- Interoperability, defining how the ODIN Ontology allows the exchange of data and services. This is necessary to cover these aspects for the ODIN technological platform.
- Standardization, in WP8 the OdinEMDN can be taken to a proper standardization.

11 References

- [1] R. J. S. P. a. K. M. P. Kataria, «Implementation of ontology for intelligent hospital wards,» *Proceedings of the Annual Hawaii International Conference on System*, pp. 1-9, 2008.
- [2] G. A. & F. v. Harmelen, «A Semantic Web Primer,» *Strutturare la conoscenza: XML, RDF, Semantic Web*, 2008.
- [3] H. S. ., D. T. a. D. P. K. Arabshian¹, «GLOSERV: GLOBAL SERVICE DISCOVERY USING THE OWL WEB ONTOLOGY LANGUAGE,» pp. 1-6.
- [4] «OWL Web Ontology Language Overview,» [Online]. Available: <http://www.ksl.stanford.edu/people/dlm/webont/OWLOverviewMay12003.htm>.
- [5] «JSON-LD Basic Concepts,» [Online]. Available: <https://www.w3.org/TR/json-ld/#relationship-to-rdf>.
- [6] «FHIR specifications, resource formats,» [Online]. Available: <http://www.hl7.org/fhir/rdf.html#ontologies>.
- [7] «R2RML: RDB to RDF mapping language,» [Online]. Available: <https://www.w3.org/TR/r2rml/>.
- [8] J. B. L. G. C. a. B. S. S. Babcock, «The Infectious Disease Ontology in the age of COVID-19,» *Journal of Biomedical Semantics*,, vol. 12, n. 1, pp. 1-20, 2021.
- [9] E. J. M. B. a. W. R. H. J. Hanna, «Building a drug ontology based on rxnorm and other sources,» *Journal of biomedical semantics*, vol. 4, n. 1, pp. 1-9, 2013.
- [10] «ITEMAS ontology for healthcare technology innovation,» *Health Research Policy and Systems*, vol. 17, n. 1, pp. 1-10, 2019.
- [11] «Ontobee Introduction,» [Online]. Available: <http://www.ontobee.org/introduction>.
- [12] Z. X. B. Z. Y. L. Y. L. J. Z. C. M. M. C. A. R. E. Ong, «Ontobee: a linked ontology data server to support ontology term dereferencing,» *Nucleic acids research*, vol. 45, n. D1, pp. 347-352, 2017.
- [13] «About ncbo,» [Online]. Available: <https://ncbo.bioontology.org/about-ncbo>.
- [14] N. H. S. P. L. W. B. D. M. D. N. G. C. J. D. L. R. N. F. Noy, «Bioportal: ontologies and integrated data resources at the click of a mouse,» *Nucleic acids research*,, vol. 37, n. 2, pp. W170-173, 2009.
- [15] M. R. K. D. M. Z. M. C. H. C. B.-J. R. S. A. Maitra, «Using ethnographic methods to classify the human experience in medicine: a case study of the presence ontology,» *Journal of the American Medical informatics Association*, vol. 28, n. 9, pp. 1900-1909, 2021.
- [16] B. S. a. L. G. W. Ceusters, «A terminological and ontological analysis of the NCI Thesaurus,» *Methods of Information in Medicine*, vol. 44, n. 4, pp. 498-507, 2005.

- [17] Q. T. Zeng and T. Tse, «Exploring and developing consumer health vocabularies,» *Journal of the American Medical Informatics Association*, vol. 13, n. 1, pp. 24-29, 2006.
- [18] T. E. a. P. Barnaghi, «IoT-Lite Ontology,» n. Novembre, 2015.
- [19] F. F. F. A. a. K. S. K. S. El-Sappagh, «“SNOMED CT standard ontology based on the ontology for general medical science,» *BMC Medical Informatics and Decision Making*, vol. 18, n. 1, pp. 1-19, 2018.
- [20] G. F. C. F. G. F. P. G. T. H. C. J. B. Gibaud, «Toward a standard ontology of surgical process models,» *International Journal of Computer Assisted Radiology and surgery*, vol. 13, n. 9, pp. 1397-1408, 2018.
- [21] M. Glöckner and A. Ludwig, «Lose ODP - an ontology design pattern for logistics services,» *Advances in Ontology Design and Patterns*, pp. 131-143, 2017.
- [22] J. L. G. M. K. F. C. D. R. a. M. P. G. O. Hakimi, «The Devices Experimental Scaffolds and Biomaterials Ontology (DEB): A tool for Mapping, Annotation and Analysis of Biomaterials Data,» *Advanced Functional Materials*, vol. 30, n. 16, 2020.
- [23] S. S. Y. L. Z. X. A. G. S. Z. D. J. L. T. C. T. Y. He, «OAE: The Ontology of Adverse Events,» *Journal of Biomedical Semantics*, vol. 5, n. 1, pp. 1-13, 2014.
- [24] J. H. D. W. M. B. a. W. R. H. A. Hicks, «“The ontology of medically related social entities : recent developments,» *Journal of Biomedical Semantics*, pp. 1-4, 2016.
- [25] M. K. I. J. D. B. M. U. H. M. J. M. R. L. J. Ison, «EDAM: An ontology of bioinformatics operations, types of data and identifiers, topics and formats,» *Bioinformatics*, vol. 29, n. 10, pp. 1325-1332, 2013.
- [26] M. H. R. M. L. G. F. S. a. P. P. K. Janowicz, «BOT: The building topology ontology of the W3C linked building data group,» *Semantic Web*, vol. 12, n. 1, pp. 143-161, 2020.
- [27] S. G. Q. L. L. S. M. J. D. E. B. B. S. M. H.-A. A. Y. Lin, «“CTO: A community-based clinical trial ontology and its applications in PubChemRDF and SCAIView,» *CEUR Workshop Proceedings*, vol. 2087, n. 2020.
- [28] L. Z. I. M. a. H. C. J. McMurray, «“Ontological modeling of electronic health information exchange,» *Journal of Biomedical Informatics*, vol. 56, pp. 169-178, 2015.
- [29] «ITEMAS ontology for healthcare technology innovation,» *Health Research Policy and Systems*, vol. 17, n. 1, pp. 1-10, 2019.
- [30] J. L. C. S. R. F. V. A. V. M. A. R. M. A. L. E. Prestes, «Towards a core ontology for robotics and automation,» *Robotics and Autonomous Systems*, vol. 61, n. 11, pp. 1193-1204, 2013.
- [31] H. R. a. M. I. Hussain, «A light-weight dynamic ontology for Internet of Things using machine learning technique,» *ICT Express*, 2020.

- [32] S. K. S. B. D. S. D. H. a. S. M. “. H. P. N. Robinson, «A Tool for Annotating and Analyzing Human Hereditary Disease,» *The American journal of Human Genetics*, vol. 83, n. 5, pp. 610-615, 2008.
- [33] P. N. R. a. S. Mundlos, «The Human Phenotype Ontology,» *Clinical Genetics*, vol. 77, n. 6, pp. 525-534, 2010.
- [34] A. S. C. a. M. D. A. N. F. Santana, «The ontology for the telehealth domain –TEON,» *Journal of the International Society for Telemedicine and eHealth*, n. 2016, pp. 1-8.
- [35] Q. T. Zeng and T. Tse, «Exploring and developing consumer health vocabularies,» *Journal of the American Medical informatics Association*, vol. 13, n. 1, pp. 24-29, 2006.
- [36] «Organization ontology,» January 2014. [Online]. Available: <https://www.w3.org/TR/vocab-org/>.
- [37] «Nci thesaurus,» August 2021. [Online]. Available: ” <https://ncithesaurus.nci.nih.gov/ncitbrowser/>.
- [38] «SNOMED 5 steps briefing,» [Online]. Available: <https://www.snomed.org/snomed-ct/five-step-briefing>.
- [39] «ICD9 for Health Ontology Mapper,» [Online]. Available: <https://bioportal.bioontology.org/ontologies/HOM-ICD9>.
- [40] I. R. a. A. S. S. C. f. S. Activities., «Institute of Electrical and Electronics Engineers., and IEEE-SA Standards Board,» *IEEE Standard Ontologies for Robotic and Automation*, 2015.
- [41] «Protégé documentation,» [Online]. Available: <http://protegeproject.github.io/protege/>.
- [42] «EMDN European Medical Device Nomenclature,» [Online]. Available: <https://webgate.ec.europa.eu/dyna2/emdn/>.
- [43] «EUDAMED Device Nomenclature,» [Online]. Available: <https://ec.europa.eu/tools/eudamed/#/screen/nomenclatures>.
- [44] «Cellfile plugin,» [Online]. Available: <https://github.com/protegeproject/cellfie-plugin>.
- [45] «San Carlos Clinical Hospital,» [Online]. Available: <https://www.comunidad.madrid/hospital/clinicosancarlos/nosotros/oferta-asistencial>.

