This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement Nº 101017331

D3.3 Hospital Knowledge Base and ODIN semantic ontology v2

Deliverable No.	D3.3	Due Date	28/02/2023
Description	This deliverable shows the second 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
Giuseppe Fico	UPM	gfico@lst.tfo.upm.es
Alejandro M. Medrano Gil	UPM	amedrano@lst.tfo.upm.es
Javier del Río Martín	UPM	jrio@lst.tfo.upm.es
Peña Arroyo	UPM	pena.arroyo@lst.tfo.upm.es
Liss Hernandez	UPM	lhernandez@lst.tfo.upm.es
Ilias Kalamaras	CERTH	kalamar@iti.gr
Marcello Chiurazzi	SSSA	marcello.chiurazzi@santannapisa.it

History

Date	Version	Change
28/09/2022	0.1	Initial TOC based on V1(D3.2)
21/12/2022	0.2	Update TOC
31/03/2023	0.3	Updated introduction, Semantic technologies, Method, selected technologies.
05/04/2023	0.4	Updated Abstract, ODIN ontology, conclusions, added Ontology Governance model
05/04/2023	0.5	Sent to peer review
28/04/2023	0.6	Several updates, Section 8 restructuring.
17/05/2023	0.7	Section 8 completed.
18/05/2023	0.8	Applied peer review comments from SSSA, UCBM and ROB
22/05/2023	0.9	Added update on Section 9, and conclusions
23/05/2023	1.0	Final version

Key data

Keywords	Semantic web, ontologies, EMDN, JSON-LD, Turtle, RDF, OWL, SPARQL
Lead Editor	Alejandro Medrano Gil (UPM)
Internal Reviewer(s)	ROB, UCBM, SSSA

Abstract

This Deliverable discusses the second version of the ODIN semantic ontology, a platform that, through the Internet of Things (IoT), Robotics and Artificial Intelligence (AI), optimizes the activities of the hospitals participating in the ODIN project. The analysis of the scientific publications made it possible to identify the articles and other sources 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 V2 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 workers' 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 and as the data model basis for the ODIN platform communications and internal representation and features. A Governance model is provided to enable the update of the ODIN ontology as the needs from ODIN project and platform evolve. A set of use examples of the ontology are provided to demonstrate how to get the most value out of the model through the platform and representation of pilots.

Statement of originality

Table of contents

List of tables

List of figures

1 INTRODUCTION

Ontologies are vocabularies that make explicit the meaning of words and generate, by means of the relations between them, a semantic tree that, as it extends, creates ever finer distinctions. An ontology structurally represents general concepts, single semantic fields, or a whole knowledge domain. In all 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 individuals (e.g. The class Robot represents the concept of Robot, where the HOSTBot prototype deployed in UCBM is an individual of said class). Finally, the classes are related to each other through properties which express the relationships between concepts. Communication is fundamentally the 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 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.

1.1 Deliverable context

This is the second version of the Deliverable D3.2; because it is important to understand the full context of the semantic model for the ODIN platform, and in order to keep the deliverable as self-contained as possible, part of the content from the previous version has been maintained and updated for this version. The most relevant changes are:

- inclusion of SPARQL as an important semantic technology language that will be relevant for ODIN platform.
- Update of the Search method and search engines, so that it includes not just research on ontologies but also practical ontologies.
- Update on the results of the ontologies found, from the updated method (see Table 3-1)
- Replacement of Web of Things model by SSN, as it better represents the IoT domain.
- Inclusion of ontologies to model dataset (DCAT-AP), as datasets are considered KERs like IoT, Robots and AI in v2 of ODIN.
- Update of the ontology design, with the new selected ontologies, and with the input received from V1.
- Update of the examples of ontology usage, to reflect more specific examples in ODIN v2
- Integration of harmonized datasets from case study of pilots with more pilot models

- Section 10 is a new section reflecting the update and governance policies for the ontology, allowing for the management of the ontology in a more dynamical way which adapts to future ODIN needs. -Since this will be the last version of the deliverable, this section puts forward the method by which the updates on the ontology will be agreed and communicated.

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 V2 are provided in this document. First, we provide a brief introduction to the programming languages which are typically used to develop an ontology. Following LOT (Linked Open Terms), a Methodology is a method for developing ontologies (see Figure 1.1), we will review the literature and existing ontologies for existing modes (see section 3) that match the modeling requirements of ODIN project, this version emphasizes technological aspects required by the ODIN platform. The ontologies that have been considered important (see section 4) for the creation of the ODIN Ontology, this is the ODIN, ODINEMDN and ODIN Process Ontologies (see section 5) and their relation are described as part of the second LOT block. The deliverable also provides extensive methodological, and practical approaches to make use of the ontology and semantic technologies, in sections 6, 7, 8 and 9. The last 2 blocks of the LOT method will be discussed in section 10.

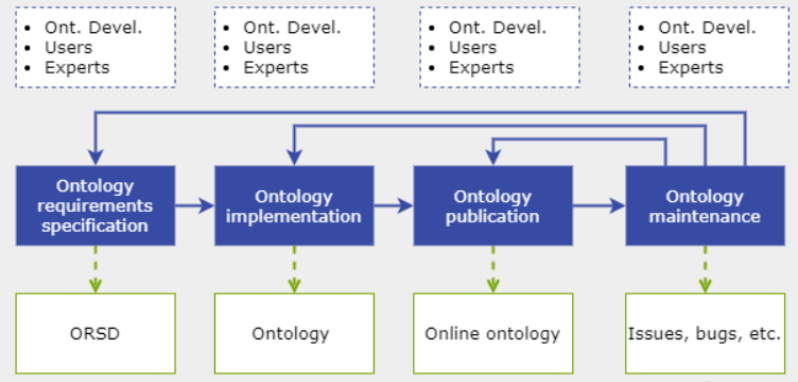

Figure 1.1. LOT (Linked Open Terms) Methodology. Credit[1].

Table 1-1. Deliverable context.

PROJECT ITEM IN THE DOA	RELATIONSHIP
Project Objectives	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. The relevant WP3 objectives to which this deliverable contributes are:

[1] https://lot.linkeddata.es/

	• 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. • 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	The semantic model for Hospital Management, as well as the model for technological management and interoperability offered through ODIN platform, described in this deliverable may be exploited directly, or as assets to further the knowledge domains they constitute as well as material to better disseminate ODIN platform as a whole.		
Workplan	D3.3 is attributed to the task T3.2 of WP3, Platform Semantic Ontology and Harmonized Datasets. It is the second and final version of D3.2.		
Milestones	The D3.3 is a key deliverable for the Implementation (MS3) and Pre-commercial Procurement (MS4) milestones of the project by providing the data model and logical models of the ODIN technological and piloting aspects.		
Deliverables	D3.5	Privacy Security and Trust report	Regarding security mechanisms
	D3.8	Technical Support Plan and Operations	Regarding component documentation and feedback collection.
	D3.11	ODIN platform	Regarding the application of DevOps in the development of the ODIN platform.
	D4.3	ODIN platform components	Regarding the model and usage of core components of the platform as well as the integration of 3rd party KERs
	D4.6	ODIN platform Advanced components	Regarding the model for the usage and representation of

			advanced components such as orchestrator and federation.
	D5.8	ODIN Robotics integration	Regarding the models needed for the representation and integration of robotic platforms in ODIN platform
	D6.1	Data model for AI operations	Regarding the data set models used for AI model training
	D6.6	High-level AI models	Regarding the AI models for representation and integration with ODIN platform.
	D7.1	Use Case definitions and KPIs	Regarding the process and KPI models in order to map to ODIN piloting practices
Risks	The ontology provided in this deliverable can help in minimizing the following risks identified in the Grant Agreement: • Technologies might not available on time. The ontologies provide a structure which enables the development of unknown/unavailable technologies. • Possible Technical problems during component/module development. This deliverable provides all the knowledge needed for developers to overcome development issues using semantic technologies. • Growing complexity of unification procedure. The Ontologies are an excellent form of managing complexity in software development.		

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)[2] with the main goal of enabling computers to support interactions over the network. The Semantic Web, Figure 2.1, 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 Ontology[3]. The Web Ontology Language (OWL) is a logic-based language designed to represent complex knowledge and relationships between items and it is the format used to define ontologies[4], 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] .

[2] https://www.w3.org/

[3] https://www.w3.org/standards/semanticweb/

[4] https://www.w3.org/OWL/

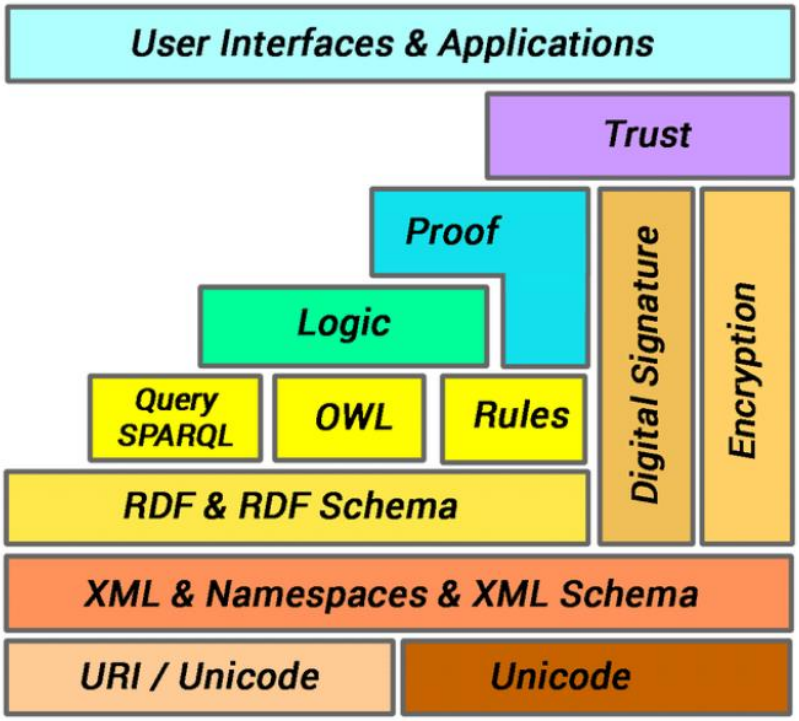

Figure 2.1. Semantic Web stack.

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[5] 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 Internationalized Resource Identifier (IRI);
- Properties: it is an attribute that you want to assign to a resource.

[5] https://www.w3.org/RDF/

- Statement: is the association of the property to the resource, its structure is a triple subject / predicate / object, as showed in Figure 2.2.

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].

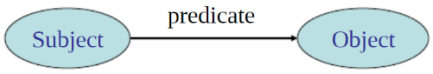

Figure 2.2. The RDF triple.

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 of 2 parts:

- RDF Model and Syntax
- RDF Schema-RDFS

RDF Schema is a simple vocabulary description language, that defines the vocabulary used in RDF data models and extends RDF, as shows Table 2-1. Figure 2.3 is an example of how the RDFS vocabulary can be used to describe something.

Table 2-1. 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, declares 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.

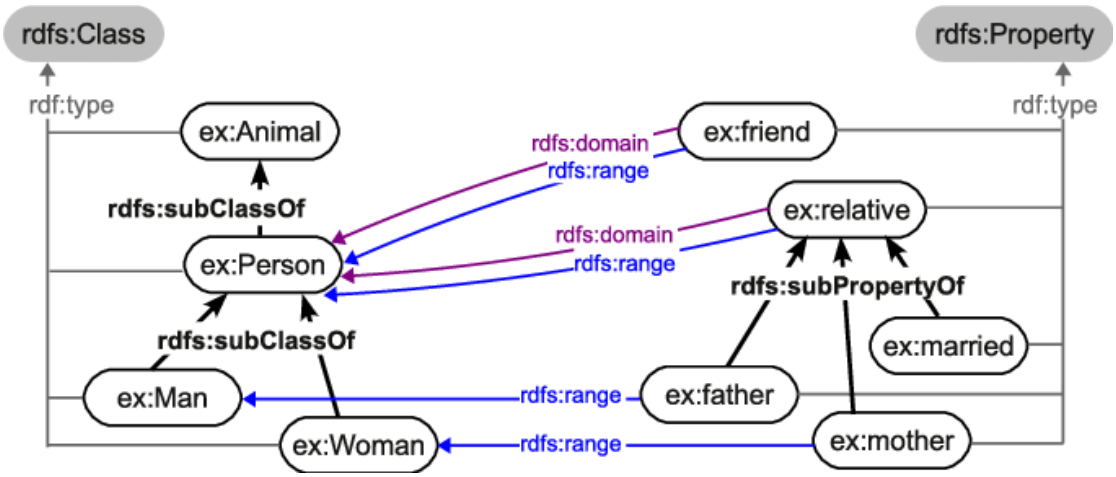

Figure 2.3. Introduction to RDFS.

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 in Figure 2.4.

The following subsections will describe the 3 sublanguages of OWL, Figure 2.5, which are OWL Lite, OWL DL, OWL Full.

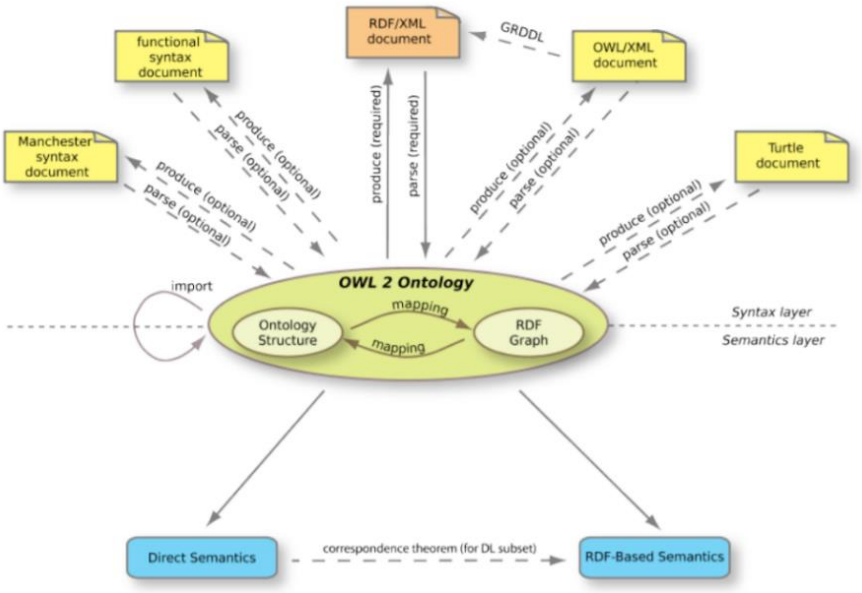

Figure 2.4. Structure of OWL 2.

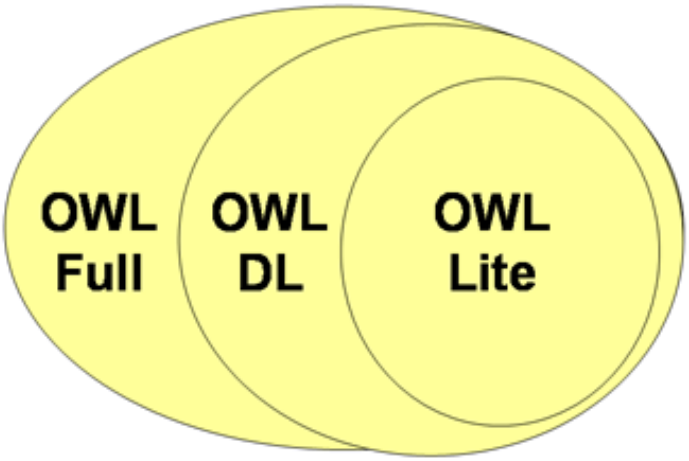

Figure 2.5. OWL variants. OWL Lite language is a subset of OWL DL, which is a subset of OWL Full.

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 versions, 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 2-2.

Table 2-2. 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 explicit those individuals who can be applied a property and who can have a property as value.
Equivalent/Different	owl:equivalentClass owl:equivalentProperty owlsameAs 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 owl:functionalProperty	A property can be defined the inverse or symmetric of another property. A transitive property is a property that can be deduced from the reasoner, for example if the

	owl:inverseFunctionalProperty	couples (Michael, David) and (David, Clara) are ancestors, then for the reasoner (Michael, Clara) are ancestors too. If a property is a functional property, then each individual should have a unique value.
Restrinctions on the type of property	owl:allValuesFrom owl:someValuesFrom	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. 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.
Cardinality restrictions	owl:minCardinality owl:maxCardinality owl:cardinality	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. 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.
Intersection of Classes	owl:intersectionOf	Intersection between classes and restrictions.
Annotation properties	rdfs:label rfds:comment rdfs:seeAlso rdfs:isDefinedBy	This annotation properties allows to better define classes, properties and individuals.
Datatypes	datatypeProperty	A relation between two instances of a class and an RDF or XML data value.

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 2-3.

Table 2-3. OWL DL and OWL Full synopsis.

Class Axioms	owl:oneOf owl:dataRange owl:disjointWith	A class can be described by listing all of its instances, thanks to one of. With dataRange the list is defined by literals.

	owl:equivalentClass rdfs:subClassOf	If two classes are disjoint, an instance of the first class cannot be an instance of the second class as well. Two equivalent classes have the same extensions and therefore the same individuals. With subClassOf can be defined the restrictions, characteristics, that the class must inherit.
Boolean Combinations of Class Expressions	owl:unionOf owl:complementOf owl:intersectionOf	Can represent Boolean combinations 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 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 2.6 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 2.6. A JSON-LD document.

2.4 FIHR

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 2.7 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 in XML (Figure 2.8), JSON (Figure 2.9) or RDF/TURTLE (Figure 2.10) encodings [6].

Structure

Name	Flags	Card.	Type	Description & Constraints
Person	TU		DomainResource	A generic person record Elements defined in Ancestors: id, meta, implicitRules, 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 2.7. 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 2.8. 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 2.9. 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 2.10. 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 relies 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 2.11, 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 these elements produces the triple. Figure 2.12 shows a direct mapping as R2RML.

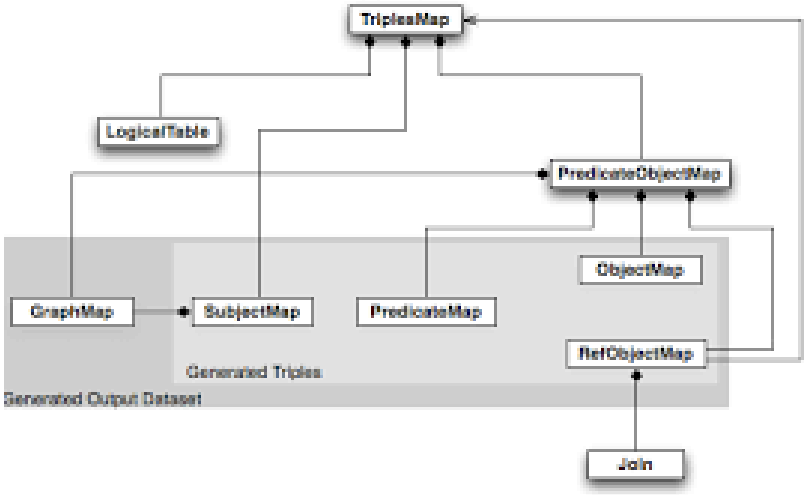

Figure 2.11. R2RML mapping language.

```
@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 2.12. R2RML direct mapping.

2.6 SPARQL

SPARQL (pronounced "sparkle") is a query language used for querying and manipulating data stored in RDF format. RDF is a standard model for data interchange on the web, and it represents information as a graph of nodes and edges.

SPARQL allows users to retrieve and manipulate data from RDF graphs by writing queries that express patterns and constraints over the graph. These queries can be used to retrieve specific pieces of information, make inferences about the data, or perform complex analytics. SPARQL includes features for querying and modifying data, including filtering, sorting, grouping, and aggregating data. It also supports complex graph patterns, subqueries, and federated queries that combine data from multiple sources.

SPARQL is widely used in the Semantic Web community. It is an important component of the linked data ecosystem. It is supported by many RDF databases (also known as triple stores) and query engines. And it has a large and active user community that continues to develop and improve the language.

Wikisource pages for authors of scientific articles

```
#Wikisource pages for authors of scientific articles, ordered by Wikisource language
#added in 2017-09
SELECT DISTINCT ?item ?wikisourceSitelink ?wikisourceLanguage WHERE {
  ?wikisourceSitelink schema:isPartOf [ wikibase:wikiGroup "wikisource" ];
                      schema:inLanguage ?wikisourceLanguage;
                      schema:about ?item.
  ?paper wdt:P31 wd:Q13442814;
         wdt:P50 ?item.
}
ORDER BY ?wikisourceLanguage
LIMIT 300
```

Figure 2.13. Example SPARQL query from wikidata[6]. Showcasing how human-readable the queries can be and how inspired the language is from standard SQL (hence the name similarities).

Some of the basic operations in SPARQL include:

- SELECT: This operation is used to retrieve data from an RDF graph based on a specific set of criteria. The SELECT operation allows users to specify which variables to return and the conditions that must be met for the variables to be included in the query results.
- WHERE: This operation is used to specify the conditions that must be met for a query to return results. The WHERE operation allows users to specify patterns in the RDF graph that must match for a query to succeed.

[6] https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/queries/examples

- FILTER: This operation is used to further refine the results of a query based on specific criteria. The FILTER operation allows users to apply conditions to the results returned by a query, such as filtering by a specific value or range of values.
- ORDER BY: This operation is used to sort the results of a query based on one or more variables. The ORDER BY operation allows users to specify the order in which the query results should be returned.
- GROUP BY: This operation is used to group the results of a query based on one or more variables. The GROUP BY operation allows users to aggregate data and compute statistics over the results of a query.
- LIMIT: This operation is used to limit the number of results returned by a query. The LIMIT operation allows users to specify a maximum number of results to return.
- OFFSET: This operation is used to skip a specified number of results in a query. The OFFSET operation allows users to specify a starting point for the results returned by a query.

While SPARQL can be used to query RDF data that is based on an OWL ontology, the two languages have different roles in the Semantic Web stack (see Figure 2.1). SPARQL is used to query and manipulate data, while OWL is used to represent and reason about the concepts and relationships in a domain. SPARQL and OWL can be used together to support more complex queries and reasoning tasks over RDF data.

In ODIN it is essential to offer query capabilities over the current logical model, in order to update the model with new information, for example when a robot has updated their semantic location (i.e. changed room); or to locate resources according to complex conditions (e.g. find the nearest free robot with interaction capabilities to assist or monitor a particular patient); inference will also be useful in many cases, especially when there are multiple routes to the desired effect (e.g. find the location of an object, when this object is carried by a robot; define the cyber security risks associated with a resource from its description). These complex queries could also be used for metric definition, as well as self-organization and discovery of the platform components themselves.

3 ONTOLOGIES FOR HEALTHCARE AND SUPPORTING TECHNOLOGY

One of the main goals of 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 to know the ontologies useful for the ODIN platform. A fundamental aspect in developing a new ontology, 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.

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 ODIN project.

3.1.1.1 Information Source

The literature search was carried out through the Scopus[7] 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

The articles needed to be relevant to the mission and the aim of the ODIN project to be considered in this review. For this reason, the search was mainly directed towards articles that focus on the technologies required 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 by 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

[7] https://www.elsevier.com/solutions/scopus?dgcid=RN_AGCM_Sourced_300005030

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 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 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. Even though this method was straightforward and easy, 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 based on 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**[8] and **BioPortal**[9]. 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. Aside from these biomedical ontology databases, other ontology and vocabularies search engines and common ontology lists could be located through W3C wiki pages like Ontology Repositories[10], List of ontologies[11], Search Engines[12] and Ontology Dowsing[13]; some of which also deepened in this method of ontology search.

The last method used for searching and locating the latest versions of ontologies is the experience gained through the participation in different research projects and awareness of standardization

[8] https://ontobee.org/

[9] https://bioportal.bioontology.org/

[10] https://www.w3.org/wiki/Ontology_repositories

[11] https://www.w3.org/wiki/Lists_of_ontologies

[12] https://www.w3.org/wiki/Search_engines

[13] https://www.w3.org/wiki/Ontology_Dowsing

activities. One clear example is the usage of HL7's FHIR, which is well within the domain of ODIN project, and common place in the vernacular of its members. This standard is not technically defined as an ontology, and therefore it cannot be found through the previous methods; however, HL7 FHIR does have all the properties and resources of an ontology and can be used in fact as one, it also employs terminologies, and other ontology

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) Foundry[7], 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)[14], 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 Protégé[15] 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.

[14] https://ncbo.bioontology.org/

[15] https://protege.stanford.edu/

3.1.2.3 Linked Open Vocabularies (LOV)

Linked Open Vocabularies (LOV)[16] is a catalogue of open and publicly available vocabularies that can be used for describing resources on the web. A vocabulary is a set of terms and relationships that define the concepts and relationships in a particular domain, or in other words a simple ontology, strictly defining terms and not the relations between them. Vocabularies are an important component of the Semantic Web, as they allow resources to be described using a common set of terms that can be understood by both humans and machines.

LOV provides a centralized repository of vocabularies that are available for use on the web. The catalogue includes information about each vocabulary, such as its name, description, and licensing terms, as well as links to the vocabulary's documentation, download location, and usage statistics. These use statistics help identify the most used vocabularies which usually are the more standardised. LOV is designed to be a community-driven resource, and users can contribute new vocabularies to the catalogue or suggest updates and improvements to existing vocabularies. By providing a comprehensive catalogue of open vocabularies, LOV helps to promote interoperability and standardization in the Semantic Web ecosystem, making it easier for resources to be shared and reused across different applications and domains.

3.2 RESULTS

The main characteristics of the documents selected for this review are displayed in Table 3-1. 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.

Table 3-1. Summary of the characteristics the selected articles.

F. Author Year Ref.	Title	Aim of the Article	Ontology	Semantic Area
Albertoni 2020[17]	Data Catalog Vocabulary	DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web.	DCAT	Dataset cataloging and interoperability
Babcock 2021 [8]	The Infectious Disease Ontology in the	Description of IDO and its extensions integrated with the analysis of COVID-19	CIDO IDO	Disease Vocabulary

[16] https://lov.linkeddata.es/dataset/lov

[17] https://www.w3.org/TR/vocab-dcat-2/

	age of COVID-19.	data: VIDO, CIDO and IDO-COVID-19.	IDO-COVID-19 VIDO	
Becnel 2017 [15]	The Biomedical Research Integrated Domain Group (BRIDG)	Provides a framework for representing and integrating data across different biomedical research domains being a CDISC, HL7 and ISO standard.	BRIDG	Biomedical research interoperability
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
Commission on Professional and Hospital Activities 1980 [17]	International Classification of Diseases, Version 9 - Clinical Modification	The ICD is the international standard diagnostic classification for all general epidemiological, many health management purposes and clinical use.	ICD9CM	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
El-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	Onto-SPM	Medical Procedure

		surgical process modelling in the context of Surgical Data Science.		
Glockner 2017 [21]	Lose ODP - an ontology design pattern for logistics services.	Presentation of an ontology design pattern for logistics services.	LoSe ODP	Services
Guarino 2007 [22]	DOLCE+DnS Ultralite (DUL)	provides a set of upper level concepts that can be the basis for easier interoperability among many middle and lower level ontologies.	DUL	Upper ontology
Hakimi 2020 [23]	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 [24]	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 [25]	The ontology of medically related social entities: Recent developments.	Description of OMRSE and its recent developments.	OMRSE	Human Role
Ison 2013 [26]	EDAM: An ontology of bioinformatics operations, types of data and identifiers,	Presenting EDAM, an ontology of bioinformatics operations, types of data and identifiers, data formats and topics with the goal of creating machine-	EDAM	Medical Data

	topics and formats.	understandable annotations for use within resource catalogues.		
Janowicz 2020 [27]	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 [28]	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 [29]	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
Mallina 2023[18]	HL7 FHIR	The ontology is based on the implicit structures defined in FHIR and is separated into Structural Definition and ValueSet mappings to OWL.	FHIR OWL	Medical Data vocabularies and services
McMurray 2015 [30]	Ontological modelling of electronic health information exchange.	Description of the conceptual framework of health system information exchange and its related ontology.	HEIO	Medical Data

[18] https://w3c.github.io/hcls-fhir-rdf/spec/ontology.html

Moreno-Conde 2019 [31]	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
Pawel Joachimiak 2022[19]	Artificial Intelligence Ontology	This ontology models classes and relationships describing deep learning networks, their component layers and activation functions, as well as potential biases.	AIO	Artificial Intelligence
Prestes 2013 [32]	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 [33]	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.	OneM2M IoT-Lite SSN	Technology
Reynolds 2012[20]	Core organization ontology	Vocabulary for describing organizational structures, specializable to a broad variety of types of organization.	ORG	Human
Robinson 2008	The Human Phenotype	Development of HPO with the goal of covering all	HPO	Disease Vocabulary

[19] https://bioportal.bioontology.org/ontologies/AIO

[20] http://www.w3.org/TR/vocab-org/

[34]	Ontology: A Tool for Annotating and Analysing Human Hereditary Disease.	phenotypic abnormalities that are commonly encountered in human diseases.		
Robinson 2010 [35]	The Human Phenotype Ontology.	Development and description of HPO to capture phenotypic information.	HPO	Disease Vocabulary
Santana 2016 [36]	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 [37]	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 sections, 97 ontologies have been selected; among these 52 were potentially useful for the objectives of the ODIN project. But after an even more detailed analysis we have concluded that the ontologies which contribute to make the ODIN ontology more understandable and achievable, are those described in the following subsections.

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

4.1 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 shown in Figure 4.1:

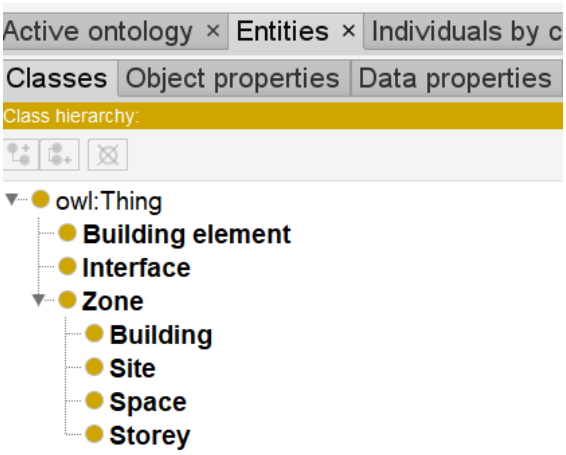

Figure 4.1. BOT Ontology classes.

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 4.2.

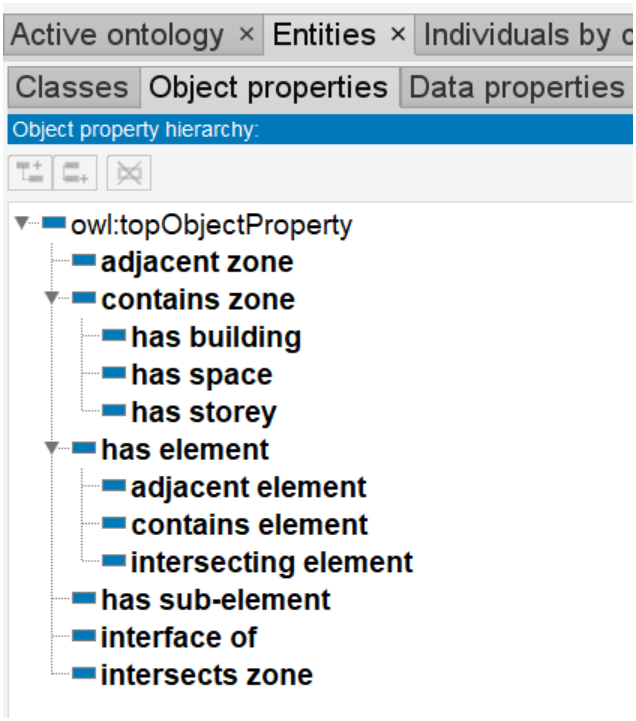

Figure 4.2. BOT Ontology object properties.

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 4.3 and Figure 4.4 it is possible to create a map of the building at a semantic level. This is an important aspect for ODIN's objectives since we may declare the medical equipment that is present in a room.

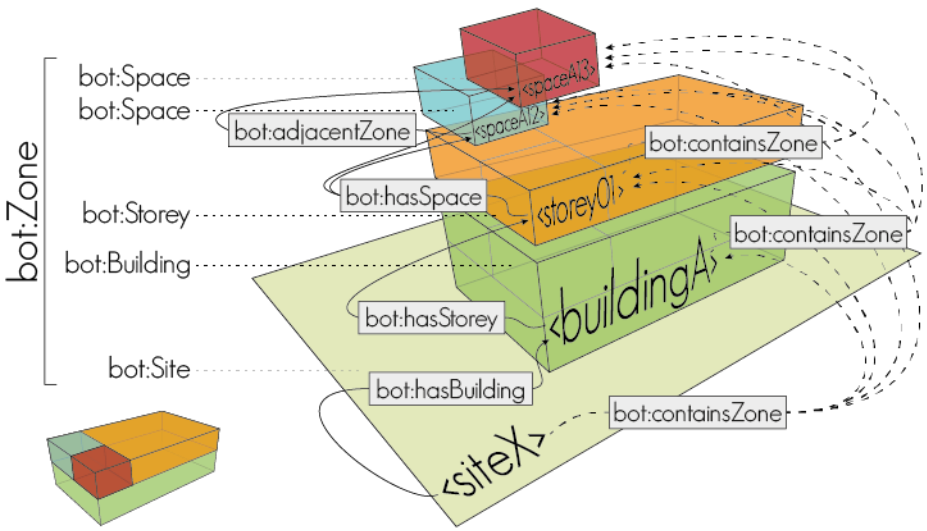

Figure 4.3. BOT Ontology - Examples of object properties linking classes.

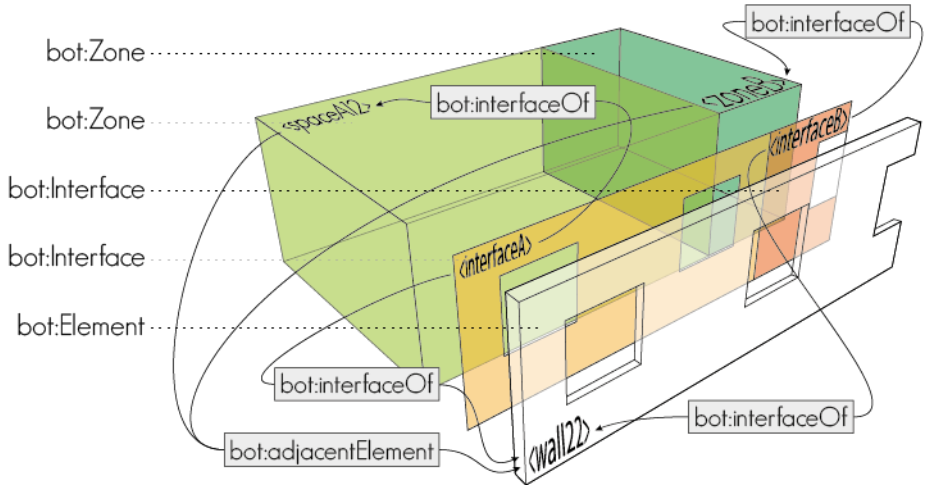

Figure 4.4. BOT Ontology - Example of object properties for the class bot:Interface.

Although it is not an ontology, it is worth mentioning that the open geospatial consortium, and standardization organism for geo-spatial data, proposes an indoor open data model as a standard representation called IndoorGML[21].

4.2 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 45 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 Collaboration, Organizational Unit, Organization, Post, Role and Site. All of the above are then logically relate through a multitude of properties [38], as shown in Figure 4.5. The Organization ontology has been selected for the ODIN ontology for the description of the departments and the units of a hospital environment, and also has been used to describe the organization of a hospital facility.

[21] http://www.indoorgml.net/

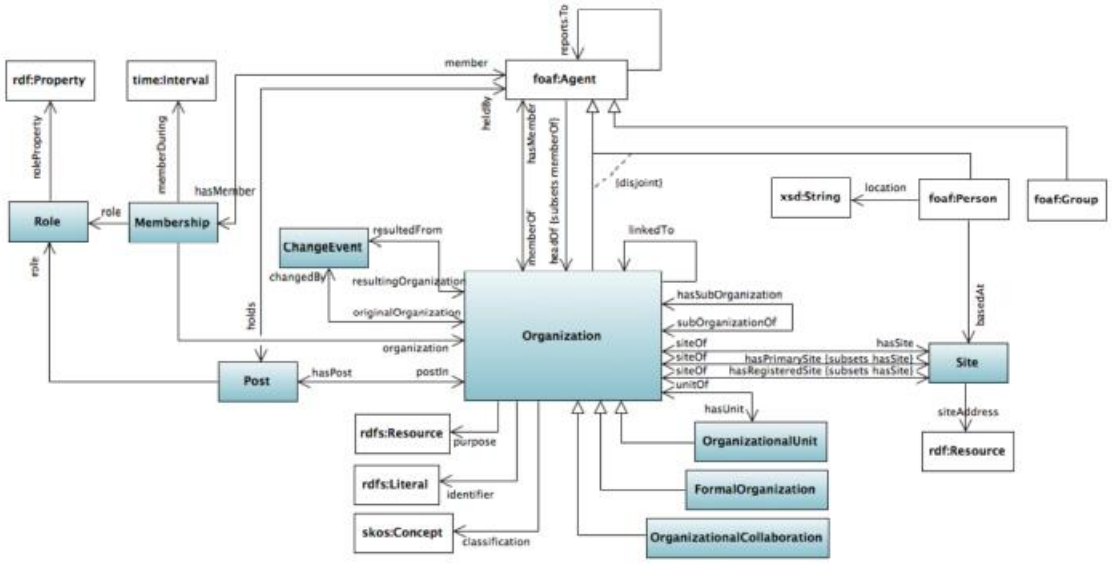

Figure 4.5. Organization Ontology.

4.3 NCIT – National Cancer Institute Thesaurus Ontology

The National Cancer Institute Thesaurus (NCIT) was developed by the National Cancer Institute's Centre for Bioinformatics and Office of Cancer Communications with the main objectives of providing a base terminology for cancer, creating a vocabulary that is understandable by both humans and machines and promoting the introduction of new concepts and relationships derived from research, clinical trials and other information sources. NCIT is a thesaurus that includes broad coverage of the cancer domain, including cancer-related diseases, findings and abnormalities. Being a thesaurus, it is defined as a controlled vocabulary organized as a list of terms and definitions. Its ontological form was developed to allow further integration between resources coming from different sources and it represents a large number of classes, most of which are described, rather than defined. This last consideration, according to [16], involves an issue related to the discrepancy between most of the definitions included in the ontological form and the ones presented by the original thesaurus. Along with this problem, this ontology also presents issues related to terms (e.g. problematic synonyms), and to the ontological representation in OWL, but nevertheless, it still provides a useful connection between the biological and clinical areas and a powerful tool to keep track of updates in these fields of interest. The ontology's domain includes vocabulary for clinical care, transitional and basic research and administrative activities [39].An overview of the ontology is displayed in Figure 4.6. For the ODIN ontology the NCIT ontology has a key role to describe the medical occupations and the allied health professions that are part of a hospital facility.

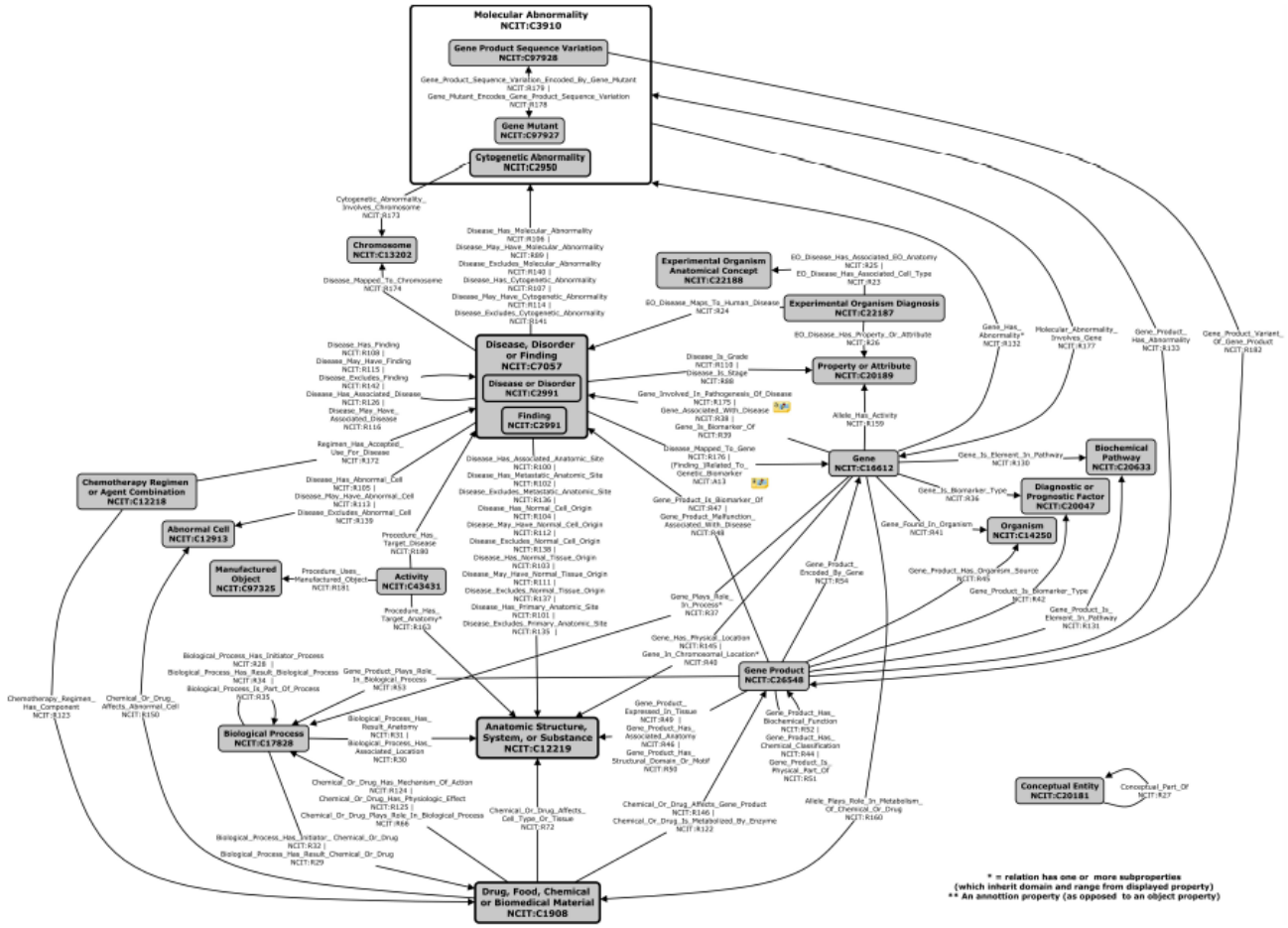

Figure 4.6. NCIT Ontology.

4.4 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 [40]. The main component types, Figure 4.7, 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.

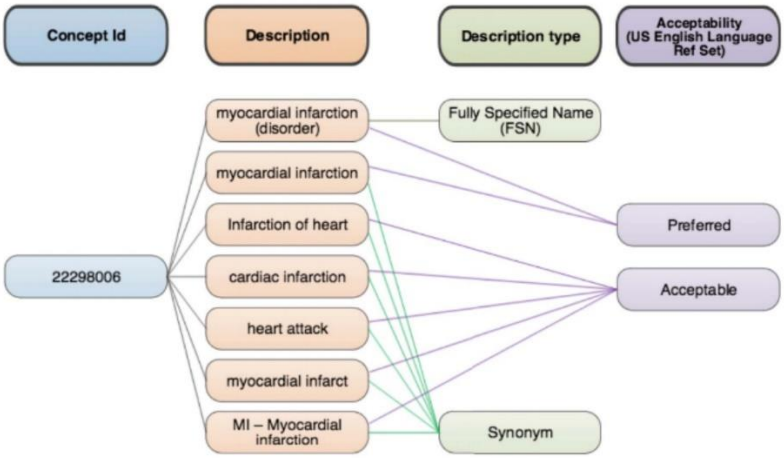

Figure 4.7. SNOMED-CT main types of components.

4.5 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 a defined criterion. 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 [17].

The main classes of the ontology are Diseases and Injuries, and Procedures whose hierarchies are shown in Figure 4.8 and Figure 4.9.

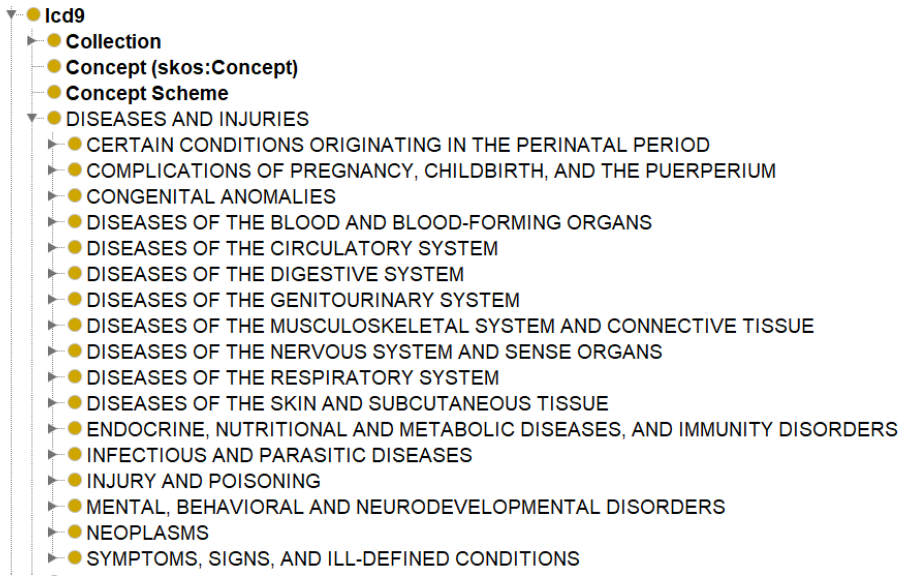

Figure 4.8: ICD9CM Diseases and Injuries class and subclasses.

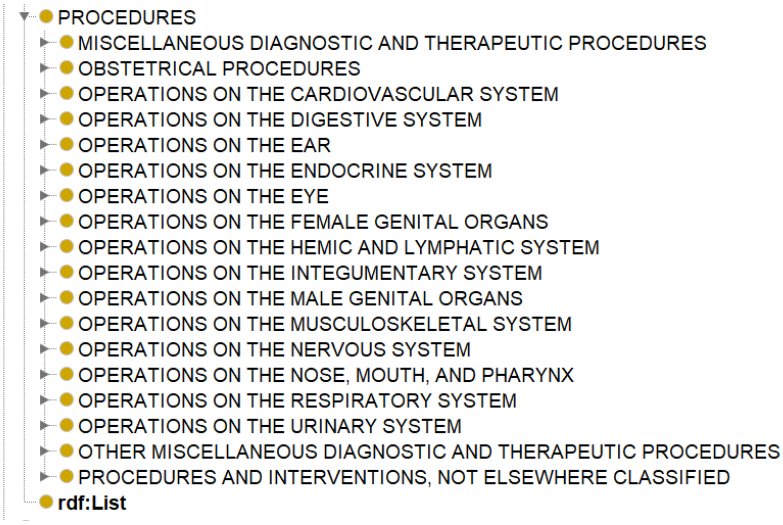

Figure 4.9. ICD9CM Procedures class and subclasses.

4.6 DCAT-AP – Data Catalog Vocabulary Application Profile

The Data Catalog Vocabulary (DCAT)[22] is a metadata standard for describing datasets and data catalogues in the Semantic Web. It provides a set of standard properties and classes for describing the characteristics of a dataset or a collection of datasets, such as the title, description, keywords, publisher, and license of the dataset. DCAT is designed to support the discovery and reuse of datasets across different domains and applications. By providing standardized metadata about the dataset, DCAT can help data publishers to increase the visibility and accessibility of their data, and data users to find and access the data they need for their work.

The DCAT Application profile for data portals in Europe (DCAT-AP) is a specification developed by the European Union based on the Data Catalogue vocabulary (DCAT) for describing public sector datasets in Europe.

Its basic use case is to enable cross-data portal search for data sets and make public sector data better searchable across borders and sectors. This can be achieved by the exchange of descriptions of datasets among data portals.

[22] http://www.w3.org/TR/vocab-dcat/

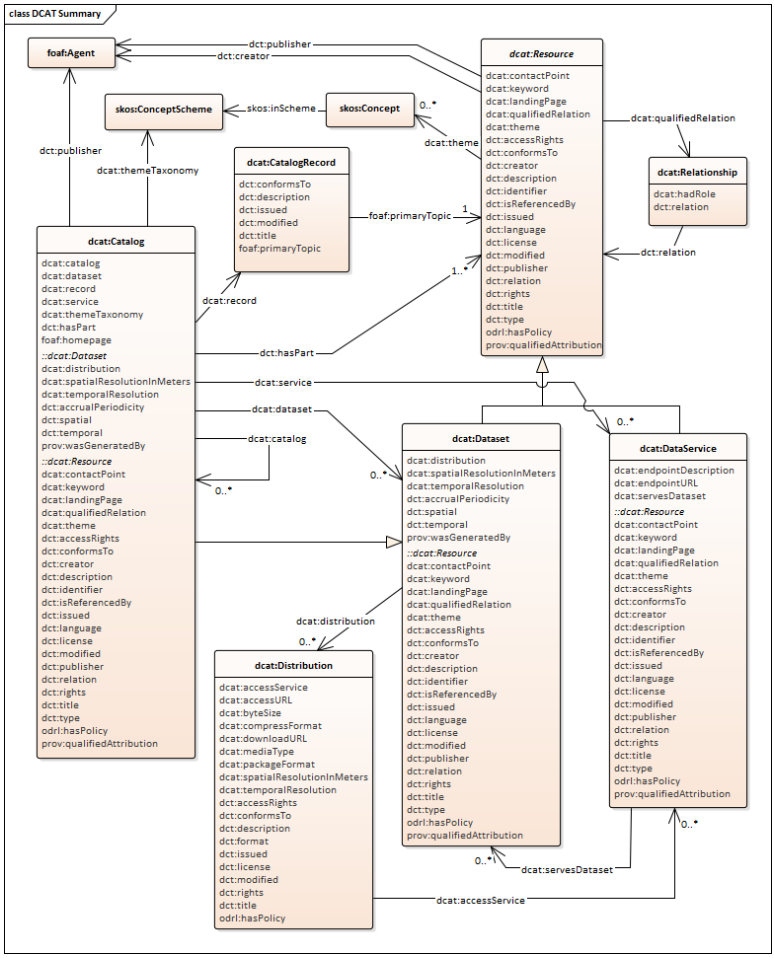

Figure 4.10. Overview of DCAT model, showing the classes of resources that can be members of a Catalogue, and the relationships between them. credit[23].

DCAT-AP provides a set of standard properties for describing datasets and data catalogues, such as the title, description, keywords, publisher, and license of the dataset. In the context of healthcare and hospital management, additional properties can be used to describe the clinical and administrative aspects of the dataset, such as the data format, data standards, data security, and privacy requirements.

DCAT-AP can be used to support a wide range of applications in the context of ODIN, such as:

1. Data discovery and access: DCAT-AP can be used to create data catalogues that provide information about the available datasets and their characteristics. This can help healthcare providers, researchers, (AI) developers, as well as tools to (automatically) discover and access the data they need for their work.
2. Data sharing and collaboration: DCAT-AP can be used to facilitate the sharing of healthcare data between different organizations, researchers and KERs. By providing

[23] https://www.w3.org/TR/vocab-dcat/

standardized metadata about the data, DCAT-AP can help to ensure that the data is used in a responsible and ethical way.

3. Data governance and management: DCAT-AP can be used to support the governance and management of healthcare data. By providing information about the data quality, data standards, and data security requirements, DCAT-AP can help to ensure that the data is managed in a consistent and reliable way.
4. Data analytics and research: DCAT-AP can be used to support data analytics and research in healthcare. By providing information about the available data and its characteristics, DCAT-AP can help researchers to identify datasets that are relevant to their research questions and to analyse the data in a more effective way.

The usage of DCAT-AP for Hospital datasets will have the additional benefit of automatically being able to participate, both as consumer and producer, in initiatives such as the European Health Data Space (EHDS), for secure and privacy-oriented health data exchange based on European regulations.

4.7 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) [41]. Figure 4.11 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.

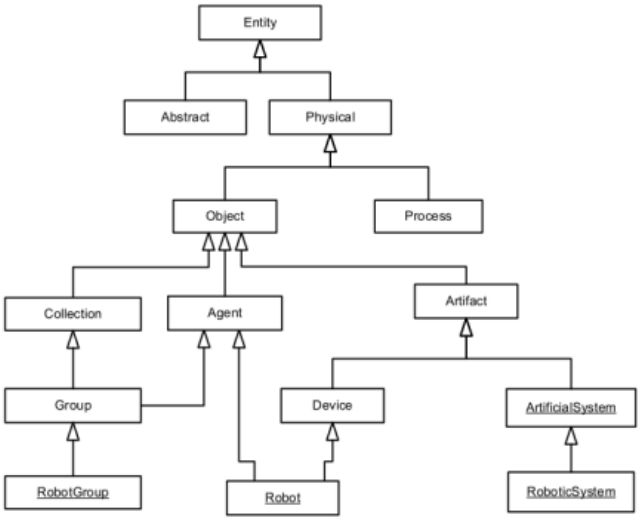

Figure 4.11. CORA Ontology.

4.8 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 4.12, with the one describing AI and its subfields and the second displaying the techniques employed in this subject.

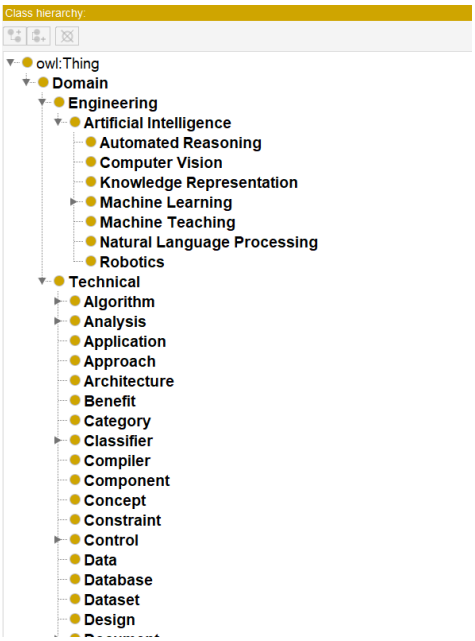

Figure 4.12. AI Ontology classes and subclasses.

4.9 HL7/FHIR

The FHIR ontology is an RDF representation of the FHIR standard, which is a set of resources and data formats for exchanging healthcare information electronically. The FHIR ontology is designed to allow FHIR resources to be expressed using the Semantic Web standards and tools.

The FHIR ontology is based on the W3C RDF and OWL (Web Ontology Language) standards and is defined using the OWL 2 Web Ontology Language. It defines a set of classes and properties for representing FHIR resources and their relationships, as well as mappings between FHIR resources and other healthcare ontologies. Some of these FHIR resources include patients, practitioners, medications, and observations. It also includes properties for representing relationships between resources, such as "subject" and "performer".

In addition, the FHIR ontology includes mappings to other healthcare ontologies, such as SNOMED CT and LOINC, which allow FHIR resources to be integrated with other healthcare data sources.

Overall, the FHIR ontology provides a powerful tool for representing FHIR resources using Semantic Web technologies, which can help improve interoperability and data exchange in the healthcare industry. By employing this ontology ODIN will be able to better interoperate its standard dataflows with FHIR services; FHIR ontology can also provide important models in ODIN context regarding not only hospital services and data, but also organization and management.

4.10SSN – Semantic Sensor Network

The Semantic Sensor Network (SSN) is an ontology for describing sensors, observations, and related phenomena in the Semantic Web. The SSN ontology provides a framework for representing and integrating sensor data from different sources, including environmental monitoring, smart cities, and industrial control systems.

The SSN ontology includes a set of core classes and relationships for describing sensors, observations, and their associated metadata. These include classes for describing physical sensors, virtual sensors, and sensor networks, as well as classes for describing observation data, sensor data streams, and data quality.

One of the key applications of SSN is the environmental monitoring inside the hospital building. SSN can be used to integrate data from different types of sensors, such as temperature sensors, humidity sensors, and air quality sensors, into a single semantic representation. This allows the development of applications for hospital environmental monitoring and management which seamlessly integrate heterogeneous data from IoT sensors.

Another application of SSN is for Smart Cities. SSN can be used to describe the different types of sensors and devices that are used in a smart city, such as traffic sensors, parking sensors, and streetlights. This allows the hospital emergency preparedness solutions to connect to all these heterogeneous data streams and obtain emergency predictions independently of the city they may be operating for.

SSN can also be used to describe the sensors and devices used in control systems, such as energy management systems and building automation systems. This allows operators, managers and even AI components to monitor and control these systems in real time, and to optimize their performance based on sensor data and other information.

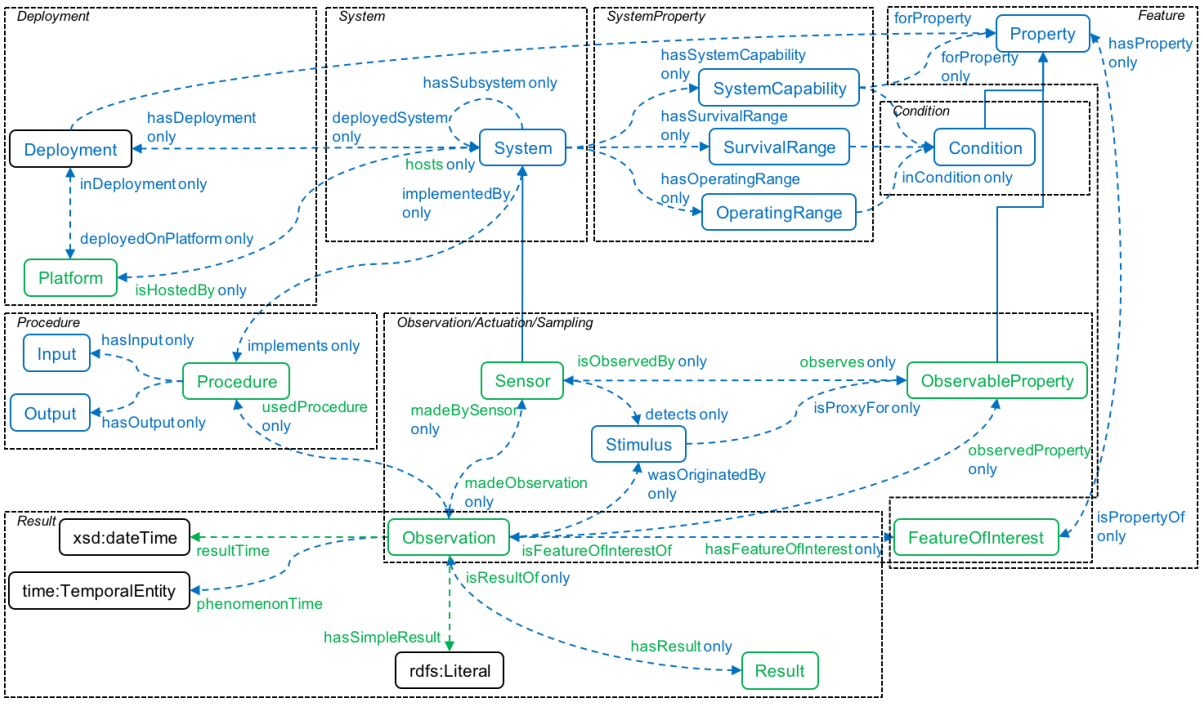

Figure 4.13. Overview of the SSN classes and properties, from an observation perspective. This is the capability to read single datapoints from sensors. SSN also provides actuation perspective, i.e the

capability to change the state of devices, and sampling perspective, this is reading continuous data from devices Credit[24].

The SSN ontology provides a standardized and interoperable way of representing IoT devices, sensor data and related metadata in the Semantic Web. This can help to facilitate the integration and analysis of sensor data from different sources, and to support a wide range of applications inside and outside the hospital environment.

[24] https://www.w3.org/TR/vocab-ssn/

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 axioms, annotations, 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.

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 developed using version 5.6.1. The entities page is the main tab of Protégé, where you can see all 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 [42].

5.1 ODIN Ontology design and realization

The created ontology is the product of the previous section's analyses. It is evident from this that the selected existing ontologies were crucial to the ODIN's realization, Figure 5.1 represents the high-level abstraction of how the different ontologies have been extended, used and mapped, in order to model ODIN domain. Terms defined in the ontology are defined by Internationalized Resource Identifier (IRI) in the namespace https://ont.odin-smarthospitals.eu/odin which will be abbreviated to the prefix **odin:** in the next sections. Section 10 discusses the method by which the ontology is published as well as the CI/CD processes which ensure the published ontology is the same version as the source material.

The result is an ontology with 12 super classes with numerous child classes, KER, Device, Domain, Element, ICD9, Interface, Measure, Occupation, Organization, Sites of Care Delivery, Unit of Measure, System. 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.

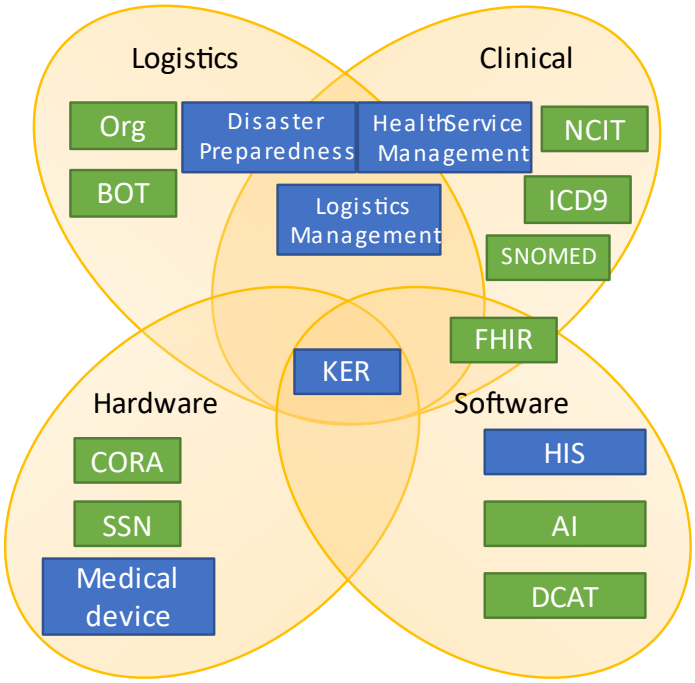

Figure 5.1. ODIN Ontology map, providing an overview of the domains and ontologies imported (green) and models produced (blue).

The classes, properties, and individuals of the ontology are explained in more detail in the next subsections, while in section 8 specific use case examples are provided and section 9 provides an application to some pilot and use cases.

Table 5-1. The main prefixes use related to existing ontologies.

ONTOLOGY	PREFIXES USED			USE
NCIT	thesaurus:	obo:		Professionals: medical and allies
SNOMED	snomed:			Description of hospital structures
CORA	sumo-cora:	pos:	cora-base:	Robots
AI	technical:	ai:		Artificial Intelligence
BOT	bot:	building:		Spaces organization and hospital elements
SSN	ssn:	sosa:		IoT devices: sensors and environment
ICD9	icd9:			Disease/injury and procedures
ORG	org:			Departments and hospital units
ODIN	odin:	odinemdn:		Odin Platform

5.1.1 Hardware KERs

Key Enabling Technologies (KERs) can adopt many forms, because the imported ontologies make a distinction on the physicality of the concept, and because this physicality is relevant to the ODIN domain, we will define first the Hardware KER models, and their properties.

CORA (see section 4.7) has defined the use of the Suggested Upper Merged Ontology (SUMO) as its upper ontology. SUMO provides a general framework for the development of other ontologies. It includes a wide range of concepts and relationships that can be used to describe many different types of entities and their interrelationships. In particular, it has a good representation of physical concepts, in concrete devices and electrical devices. Therefore the core concept for hardware KER will be the sumo-cora:Device, which will be mapped to the rest of the ontology concepts by adding the following classes as subclasses of sumo-cora:Device:

- cora-bare:Robot, which defines any automatic mechanical operator (see section 4.7).
- odin:EMDN Medical Device, which includes all electro-medical devices (see section 5.2).
- ssn:System, which defines any other IoT device with sensor, actuator or sampler capabilities (see section 4.10)

As an important part of hardware devices, it is important to establish measurement procedures, including measurement units. SSN defines the class sosa:Observation which is the abstract form of a measurement. SSN documentation recommends the usage of the Ontology of Units of Measure[25] which includes om:Measure and om:Unit of Measure classes which can be used in conjunction with sosa:Observation to provide observation measures with units. This will be the recommended model for ODIN transactions. The subclasses of om:Unit of Measure, many commonly used units such as meters, meters per second, Kg, etc., have been also selected 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 Ontology of Units of Measure Ontology has been integrated into ODIN.

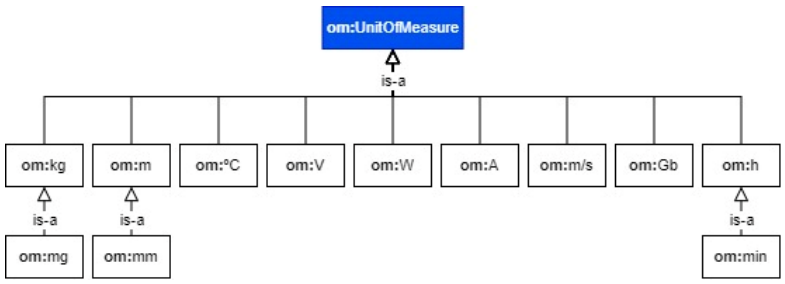

Figure 5.2. Common units of Measure integrated in the ODIN Ontology.

In the context of ODIN there are some missing models, concretely the relationships between certain Classes.

One example of this is the location of physical objects. SUMO provides sumo-cora:located for sumo-cora:Physical objects. This requires some further adaptations in the domain of places (see section 5.1.3). But it also falls short on semantics, thus while keeping sumo-cora:located as the

[25] https://github.com/HajoRijgersberg/OM

current location property, a sub-property of sumo-cora:partially-located, superclass of sumo:located, is created:

- **odin:hasHomeLocation**, indicates where the object is typically at when not in use or on a mission, this will indicate the charging base for a robot, or the spot where medical carts are usually parked so they can be returned to the proper place.

A specific model is needed for robots, the description of its current mission, thus several properties are defined:

- **odin:carries**, describes what a robot is currently carrying.
- **odin:missionFrom**, the origin of the current mission. As a sub-property of sumo-cora:partially-located.
- **odin:missionTo**, the destination of the current mission. As a sub-property of sumo-cora:partially-located

The previous version properties odin:carriesfrom and odin:deliversto are remapped as sub-properties of odin:missionFrom and odin:missionTo respectively. Also kept from previous version are the robot specifications useful for mission selection, these are depicted in Figure 5.3.

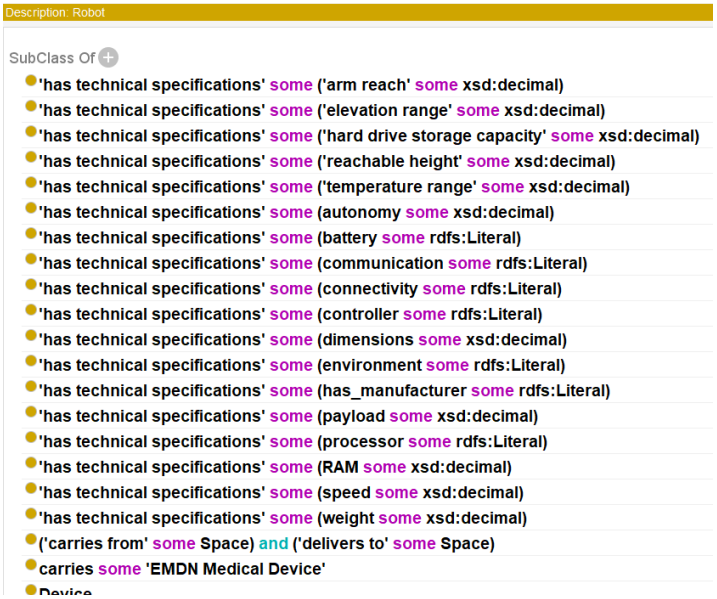

Figure 5.3. Technical specifications of the Robot class.

Finally, in ODIN's context we also need a property to model the operator of the sumo-cora:Device, if any:

- **odin:operatedBy**, describes the current agent operator of the device, this can be either human operators in presence or remote, as well as other AI components.

Table 5-2. Summary of described properties.

PROPERTY	DOMAIN	RANGE
odin:hasHomeLocation	**sumo-cora:**PhysicalObject	**sumo-cora:**Object
odin:carries	**cora-base:**Robot	**sumo-cora:**PhysicalObject
odin:missionFrom	**cora-base:**Robot	**sumo-cora:**Object

odin:missionTo	cora-base:Robot	sumo-cora:Object
odin:deliversTo	sumo-cora:PhysicalObject	sumo-cora:Object
odin:carriesFrom	sumo-cora:PhysicalObject	sumo-cora:Object
odin:operatedBy	sumo-cora:Device	odin:Operator

5.1.2 Software KERs

Non-physical KERs, which according to D3.11 include front-end, back-end applications, data storage, and AI, need to be modelled in the ODIN Ontology. The perfect candidate for all software KERs is the technical:Domain class, which is a very high-level representation of technical concepts capable of including many. However, some subclasses need to be introduced to map across different ontologies:

- ai:Engineering, which will cover most of the AI models.
- technical:Technical, which covers most applications.
- dcat:dataset, which will cover the datasets and their storage systems.
- odin:HosptialInformationSystem, a new class added just to represent HIS as KERs.

In subsequent versions of the ODIN ontology, more define models will be proposed, as specifications become clearer. In the current version, only types of KERs are represented as this is enough for the current identified needs. In the ODIN Process Ontology (see section 5.3), all common models for any KER (hardware or software) will be provided which is as deep the current specification requires to model.

5.1.3 Places

The building is defined by the classes building:Element, bot:Interface and Thesaurus:Sites of Care Delivery, and their subclasses. Thesaurus:Site of Care Delivery specifies any facility that delivers health services, this model may be combined with BOT (see section 4.1) and as a result, a structure of this sort is referred to as a Hospital Facility. Buildings, storeys, and spaces make up the Hospital Facility.

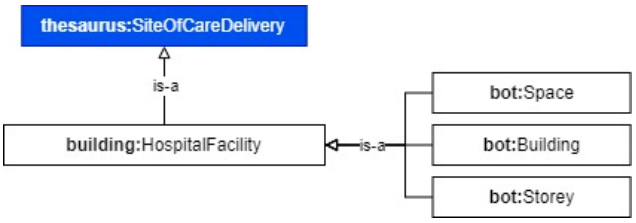

Figure 5.4. ODIN ontology mapping of Site of Care Delivery classes from NCIT, ICD9 and BOT.

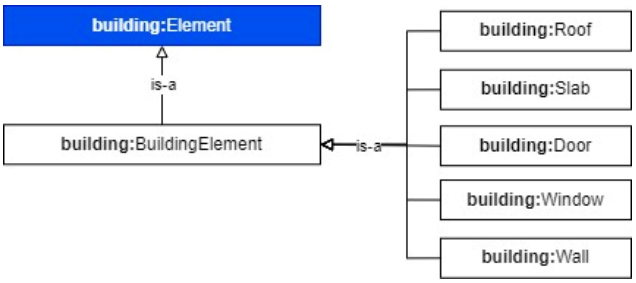

Figure 5.5. building ontology's Element class and how it is mapped to BOT through ODIN ontology.

In order to cover ODIN's use case for logistics some properties related to bot:Space have been added:

- odin:stores, indicates which kind of things can be stored in this space.
- odin:currentStorage, indicates the current storage of the space, for logistical modelling.
- odin:equipedFor, indicates the ICD9:Procedure which can be performed in this space, which will help determine which kind of supplies it needs.

This is a simple model for logistics which may be further developed in future versions of the ontology.

In order for semantic location to work, this is the usage of sumo-cora:partially-located, and its subclasses, to map to bot:zone. bot:zone is made subclass of sumo-cora:Object, which is the original range of the property.

Additionally, relative location must be documented; particularly for the Real Time Location System (RTLS) to report the location of objects (namely robots, persons, or logistical objects). This model is represented in Figure 5.6.

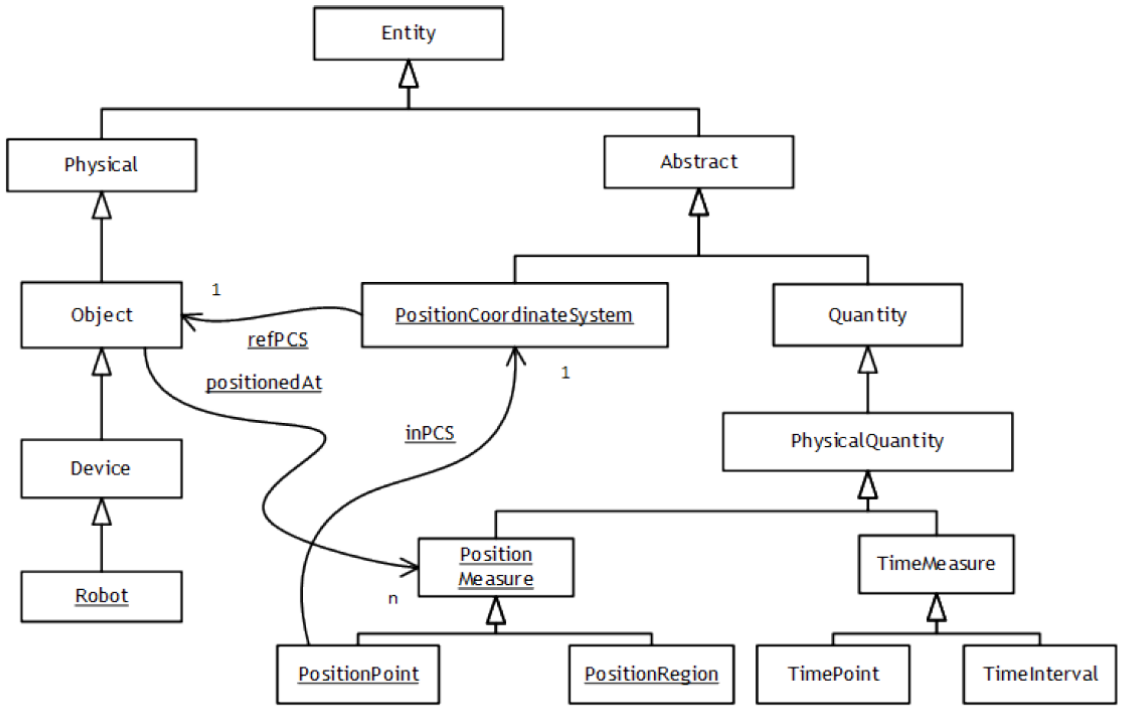

Figure 5.6. Main concepts in POS ontology from CORA regarding Position.

Relative locations are 3 coordinates representing the quantity of space from a particular point, the origin. Thus, if RTLS uses one origin, and a self-locating robot uses another origin, it can be identified that both sets of coordinates are not compatible and will require a service to translate between coordinate systems. CORA identifies the sumo-cora:PositionCoordinateSystem as the class for all coordinate systems, making them distinct from one another, each with an origin, sumo-cora:refPCS. The origin may be any object (real or virtual), which may also be located using another coordinate system.

Relative Coordinates are extended with Quaternions, this adds the capability to describe orientation, which is a meaningful component for robotic platforms as well fleet management, and navigation.

Table 5-3. Summary of described properties.

PROPERTY	DOMAIN	RANGE
odin:stores	**bot:**Zone	**sumo-cora:**PhysicalObject
odin:currentStorage	**bot:**Zone	**rdf:**List
odin:equipedFor	**bot:**Zone	**ICD9:**Procedures

5.1.4 Clinical Operations

The odin:Icd9 class defines the documentation linked to the ICD9CM ontology, whose primary classes are ICD9:DISEASES AND INJURIES and ICD9:PROCEDURES, with their subclasses, Figure 5.7. It was considered appropriate to import the whole ontology because it will be helpful in future expansions.

ICD9 has been selected as it is the most stable version of the vocabulary, there are new versions ICD10 and ICD11. These vocabularies can be used, ideally only if the piloting hospital operations are already compatible with these versions.

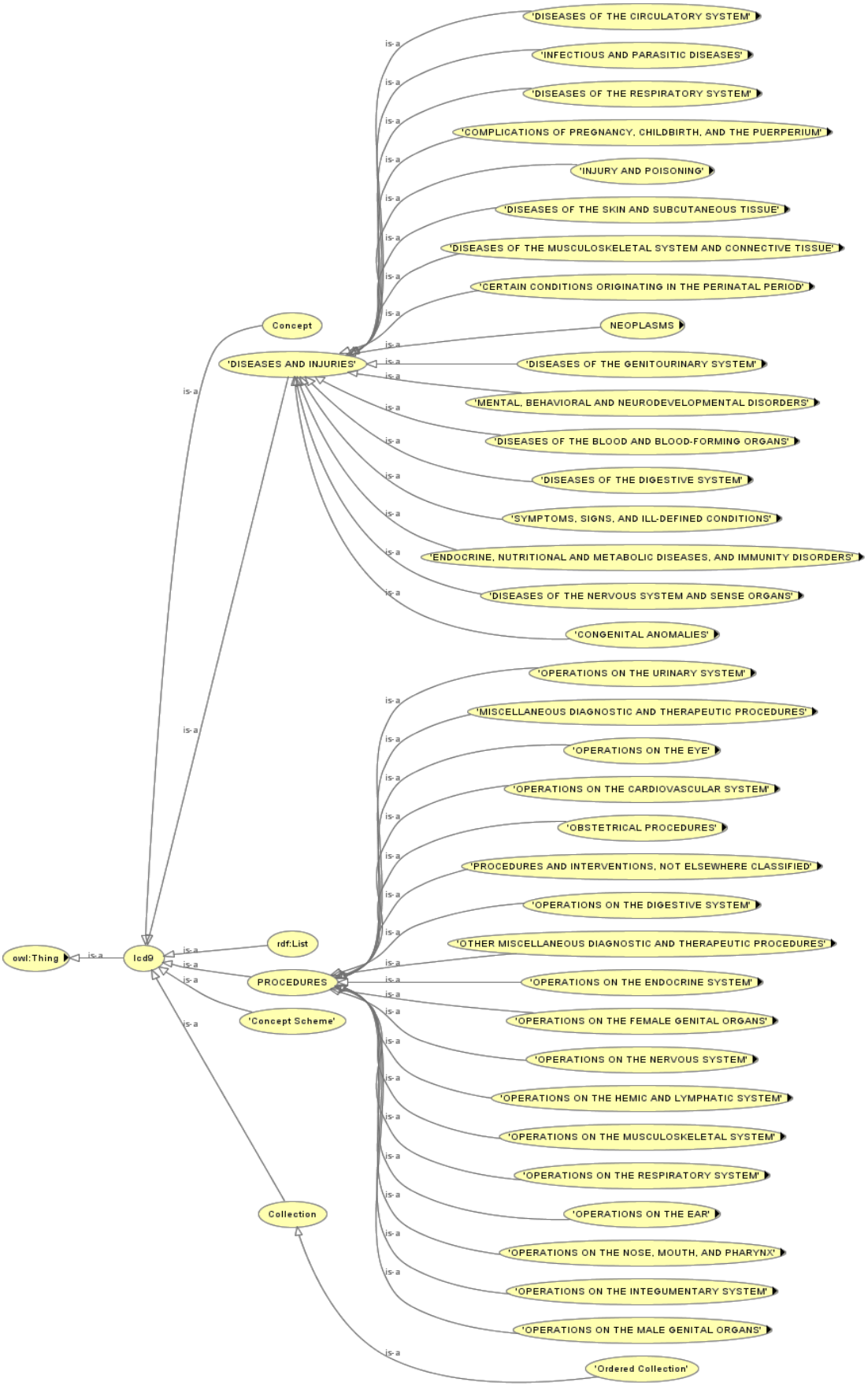

Figure 5.7. ICD9 class of ODIN Ontology.

5.1.5 Hospital Organization and Logistics

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 5.8 depicts what has been expressed. In future versions of the ODIN ontology, other of non-clinical staff like administrative, technical or legal may be modelled if needed.

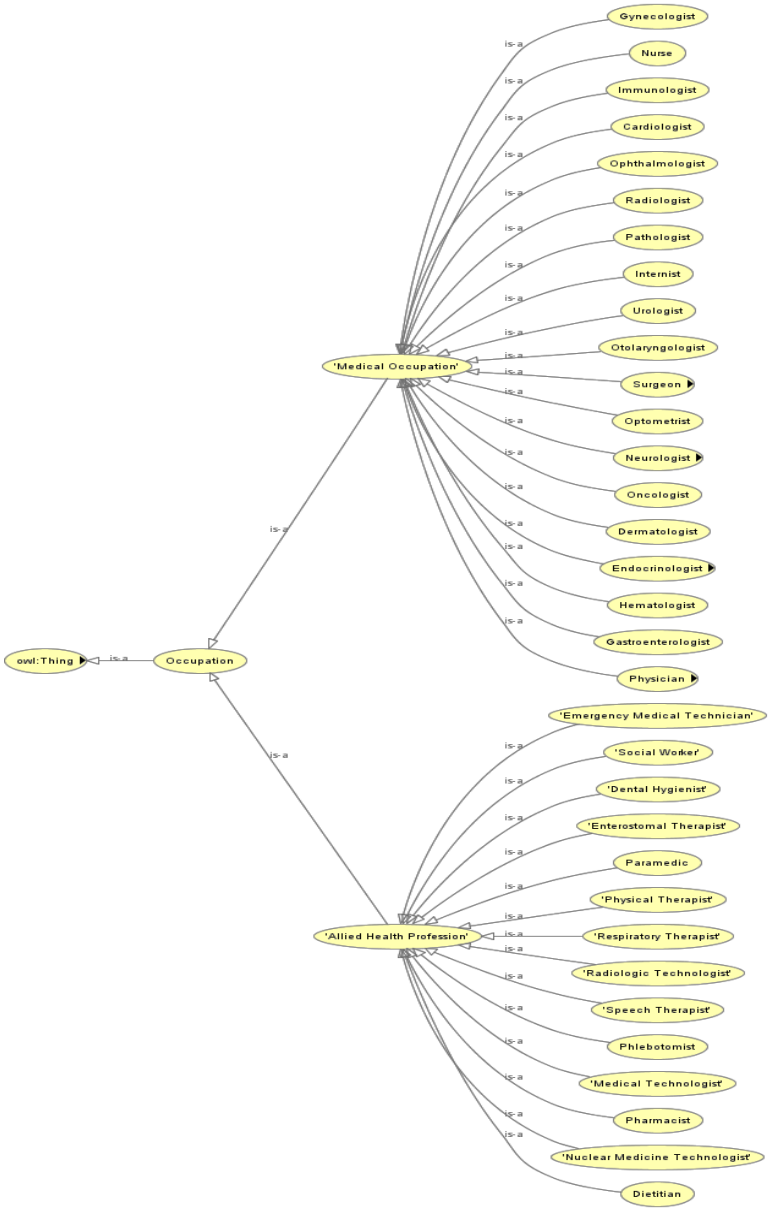

Figure 5.8. Occupation class of ODIN.

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 5.9.

Figure 5.9. Organization class.

Additional properties to map the organization to places have been added:

- **odin:operatesIn**, indicates where an org:Organization Unit may be found.
- **odin:isOperatedBy**, indicates for a bot:zone, which org:Organization Unit manages it.

Table 5-4. Summary of described properties.

PROPERTY	DOMAIN	RANGE
odin:operatesIn	**org:**OrganizationUnit	**bot:**Zone
odin:isOperatedBy	**bot:**Zone	**org:**OrganizationUnit

5.1.6 ODIN Platform internal model

In Deliverable D4.3 the Resource Descriptor is specified, this is the KER metamodel which should be a super class for both hardware KERs (sumo-cora:Device) and software KERs (technical:Domain). As such odin:KER should contain the following properties:

- **rdf:label**, the human readable name of the KER.
- **rdf:comment**, the human readable description of the KER, it is recommended to provide multilingual descriptions.
- **odin:channels**, the communication channel(s) use by this resource(see below).
- **odin:health**, describes how to check for the KERs health, in this version, it is expected to contain a http URL whose response will indicate if the KER is working (http error code class 200) or not (any other http error). In the future, this might contain other classes which enrich how the health is checked.
- **odin:graphical user interface**, this property describes how users access the KER's GUI. In this version this property is expected to contain an http URL which can be loaded by a web browser
- **odin:administration interface**, a sub property of odin:graphical user interface, this property precises the interface for administrators to use instead of normal users (if this is the case for the KER).
- **odin:documentation**, a sub property of odin:graphical user interface, this property precises the available documentation for the KER. See D3.8 on different strategies about documentation authoring, and deployment.

As explained in D3.11 KERs might use different channels of communication. The expected channel of communication are topics in the Kafka message broker. However, there can be other channels such as http services, or even real-time audiovisual streams, also allowing for any other Remote Process Communication (RPC) protocol in the future. To accommodate for this the odin:Channel class will be an abstract class, which will be extended (i.e. subclasses added) for each RPC protocol used in ODIN. As described inf D4.3, the current model for Kafka topic description will be based on AsyncAPI Specification, for REST services OpenAPI may be used as the model for the channel, for FHIR services the FHIR specification may be used.

Table 5-5. Summary of Described Properties.

PROPERTY	DOMAIN	RANGE
rdf:label	**rdf:**Thing	**String**
rdf:comment	**rdf:**Thing	**String**
odin:channels	**odin:**KER	**odin:**Channel
odin:health	**odin:**KER	**URL**
odin:graphical user interface	**odin:**KER	**URL**
odin:administration interface	**odin:**KER	**URL**
odin:documentation	**odin:**KER	**URL**

5.2 European Medical Device Nomenclature Semantic Ontology

Medical devices are the final essential component for the ODIN ontology. It is important 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 to be utilised in the future.

The European Medical Device Nomenclature (EMDN) [43] is a system of coding medical devices in the European Union (EU). It provides a standardized way of identifying and categorizing medical devices, which helps to ensure consistency and accuracy in their classification and regulation. The EMDN is based on the Global Medical Device Nomenclature (GMDN), which is used internationally. The EMDN was developed by the European Commission and is used by regulatory authorities, such as the European Medicines Agency (EMA) and regulations like medical devices regulation (MDR) and in vitro diagnostic medical devices (IVDR), to classify medical devices and ensure their safety and efficacy. The EMDN is organized into a hierarchical structure, with each level representing a more specific category of medical device. The codes assigned to each device in the EMDN provide information about its characteristics, such as its intended use, mode of action, and materials.

The alphanumeric structure of the EMDN, Figure 5.10, which is formed in a seven-level hierarchical tree, distinguishes it. It divides medical devices into three hierarchical levels [44]:

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

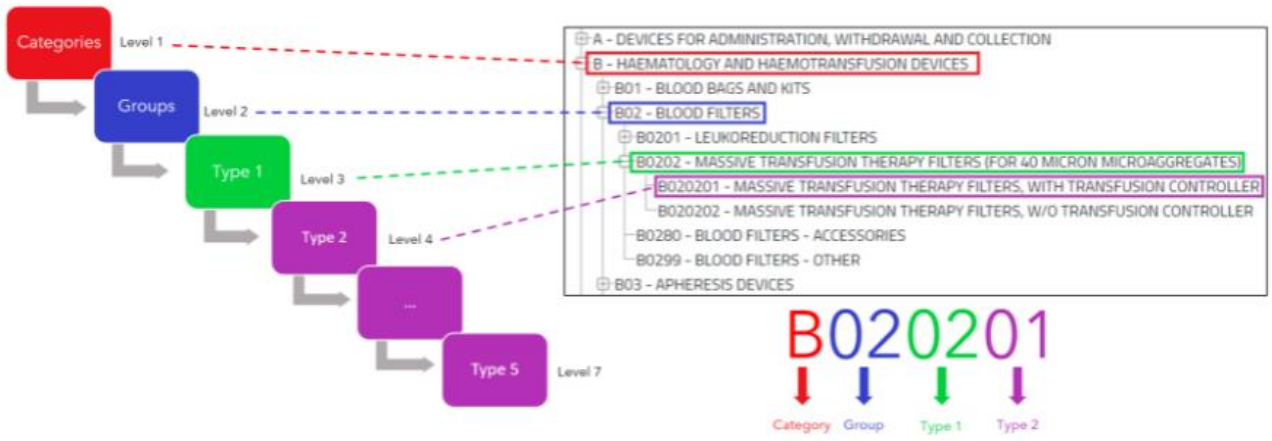

Figure 5.10. EMDN medical devices alphanumeric structure.

5.2.1 ODINEMDN Ontology Development

As the nomenclature is not formally represented in any semantic format, and in order to use this nomenclature in the ODIN project context, we decided to codify it in OWL format. As a first step in this development, the EMDN version 1.1 is downloaded as an excel file[26], which contains the whole list of medical devices that have been appropriately adjusted to reach the final ontology.

Using the CellFile plugin[27], Figure 5.11, for Protégé which allows users to transform 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 [45]. The rules are very easy to understand, these examples should be enough to build the required rules:

- 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 5.12.

[26] https://webgate.ec.europa.eu/dyna2/emdn/build/2021-09-29_%20EMDN_EUDAMED%20+%20DYNA_v1.1.xlsx

[27] https://github.com/protegeproject/cellfie-plugin

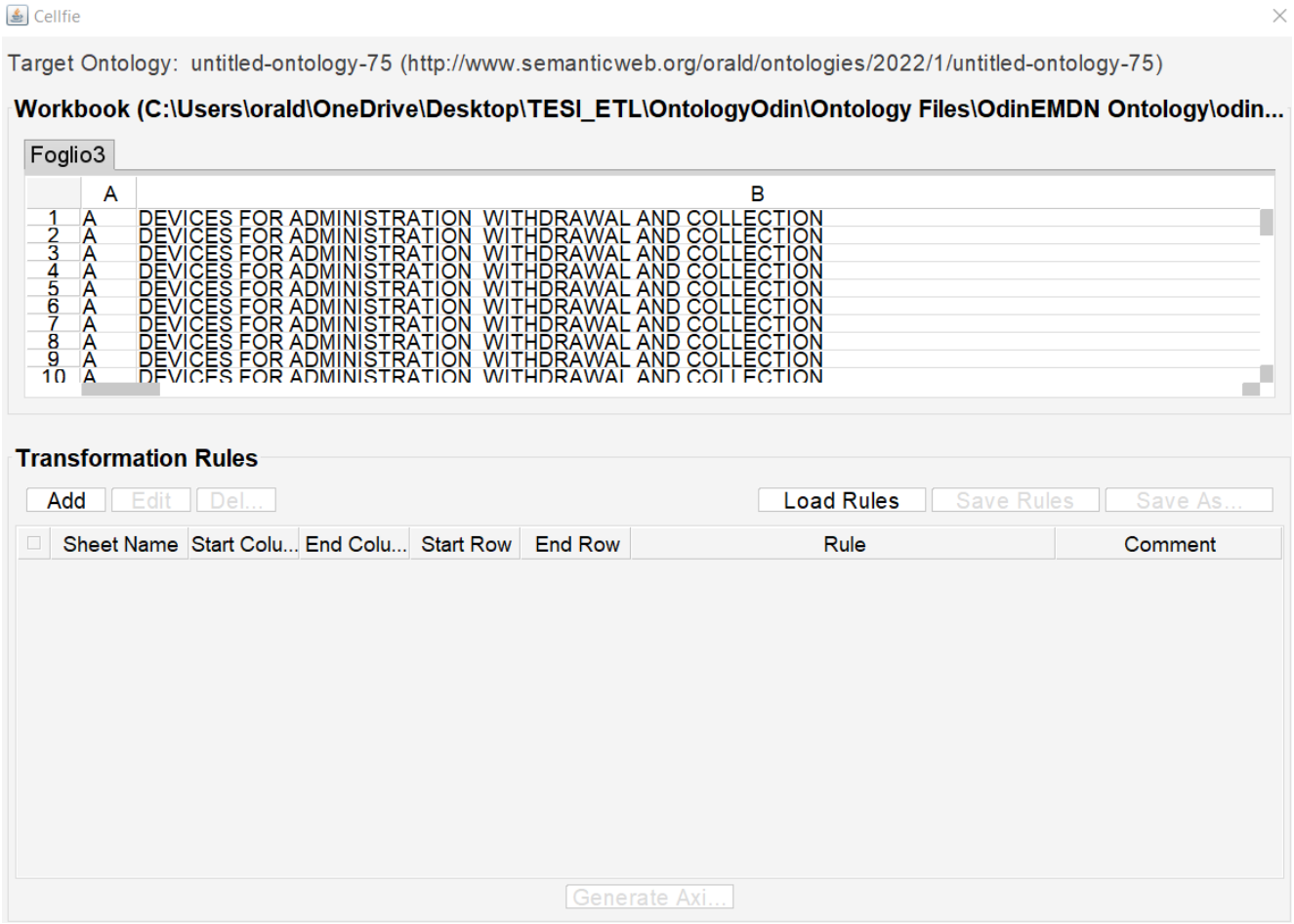

Figure 5.11. Cellfile plugin in Protégé with EMDN excel file loaded.

Add | Edit | Del... | Load Rules | Save Rules | Save As...

Sheet Name	Start Colu...	End Colu...	Start Row	End Row	Rule	Comment
Foglio3	D	D	1	+	Class: @D* Annotations: rdfs:label @F* Annotations: rdfs:comment @W*	
Foglio3	J	J	1	+	Class: @J* Annotations: rdfs:label @L* Annotations: rdfs:comment @W*	
Foglio3	A	A	1	+	Class: @A* Annotations: rdfs:label @C* Annotations: rdfs:comment @W*	
Foglio3	M	M	1	+	Class: @M* Annotations: rdfs:label @O* Annotations: rdfs:comment @W*	
Foglio3	J	J	1	+	Class: @J* SubClassOf: @G*	
Foglio3	D	D	1	+	Class: @D* SubClassOf: @A*	
Foglio3	G	G	1	+	Class: @G* SubClassOf: @D*	
Foglio3	P	P	1	+	Class: @P* Annotations: rdfs:label @R* Annotations: rdfs:comment @W*	
Foglio3	P	P	1	+	Class: @P* SubClassOf: @M*	
Foglio3	M	M	1	+	Class: @M* SubClassOf: @J*	
Foglio3	S	S	1	+	Class: @S* SubClassOf: @P*	
Foglio3	S	S	1	+	Class: @S* Annotations: rdfs:label @U* Annotations: rdfs:comment @W*	
Foglio3	G	G	1	+	Class: @G* Annotations: rdfs:label @I* Annotations: rdfs:comment @W*	

Generate Axi...

Figure 5.12. Cellfile plugin in Protégé showing the transformation rules for the EMDN excel file.

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 5.13.

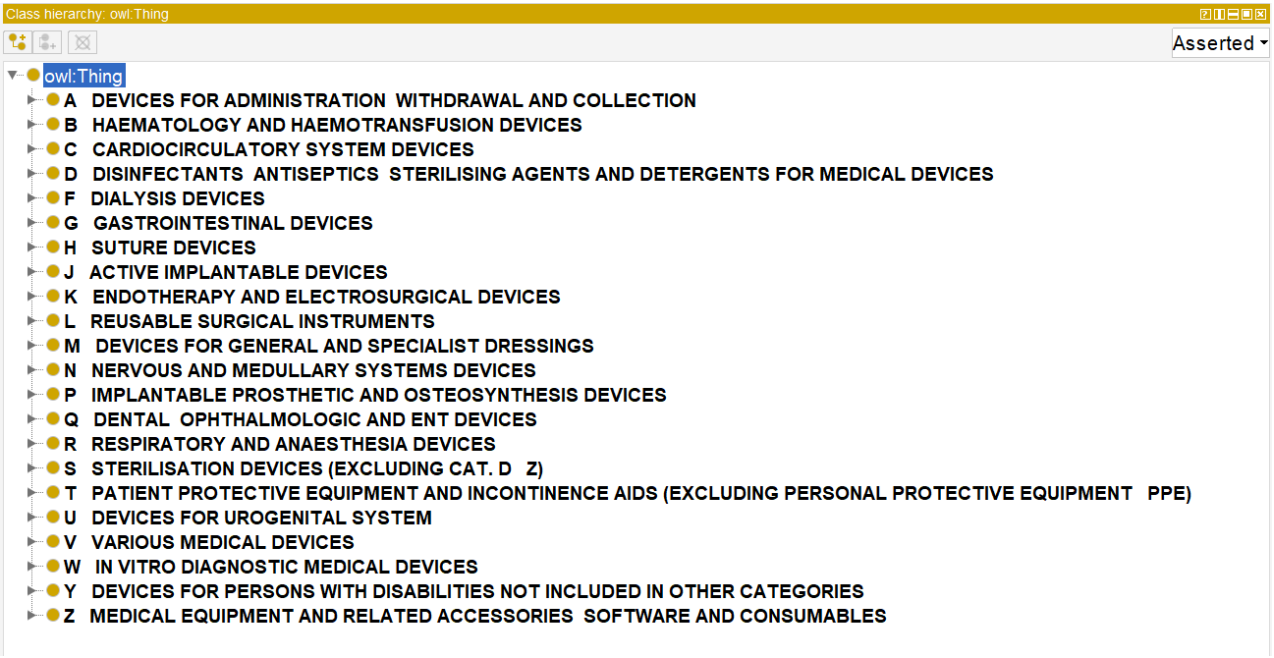

Figure 5.13. ODIN EMDN Ontology classes.

Following the definition of the classes, the ontology is completed with the object property **odinemdn: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. The odin:hasHomeLocation could be seen as a more general reverse property of this one.

5.3 ODIN Processes Ontology

From M12 to M24 ODIN's pilots, through the work performed in WP7 has advanced in interesting definitions of Reference Use Cases (RUCs) and KPIs. In this context D7.1 and other WP7 related material has been used to provide models for these concepts.

5.3.1 RUC A – Health Services Management

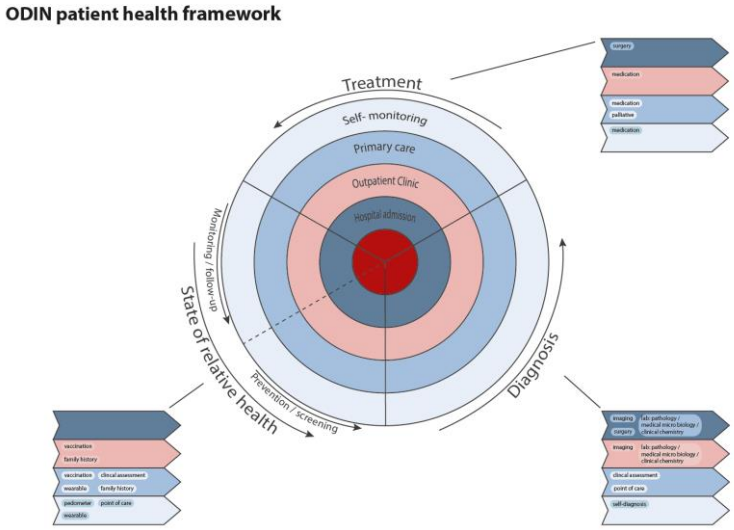

Figure 5.14. The ODIN patient health framework.

This model from D7.1 contains 3 phases (Diagnosis, Treatment, and State of relative health) with 4 Layers:

- Self-monitoring
- Primary care layer
- Outpatient clinic layer
- Hospital admission layer
- Death, not explicitly mentioned in D7.1, but it is what the red circle in Figure 5.14 represents.

The combination of the 3 phases and 5 layers gives a state machine representation of 13 states (3 phases x 4 layers + Death state). Each of these states can be associated with at least one procedure from ICD9 (see section 0). With this representation each patient journey can be easily represented as a sorted list of actions, each with a phase, layer and procedures applied at that phase. This is a very useful model to analyse individual patient journeys in the different UCs or perform aggregate analysis for optimization of hospital procedures. The model is open for future expansion, like the Disease or Injury diagnosed/treated.

5.3.2 RUC B – Devices, Goods, Facilities Management

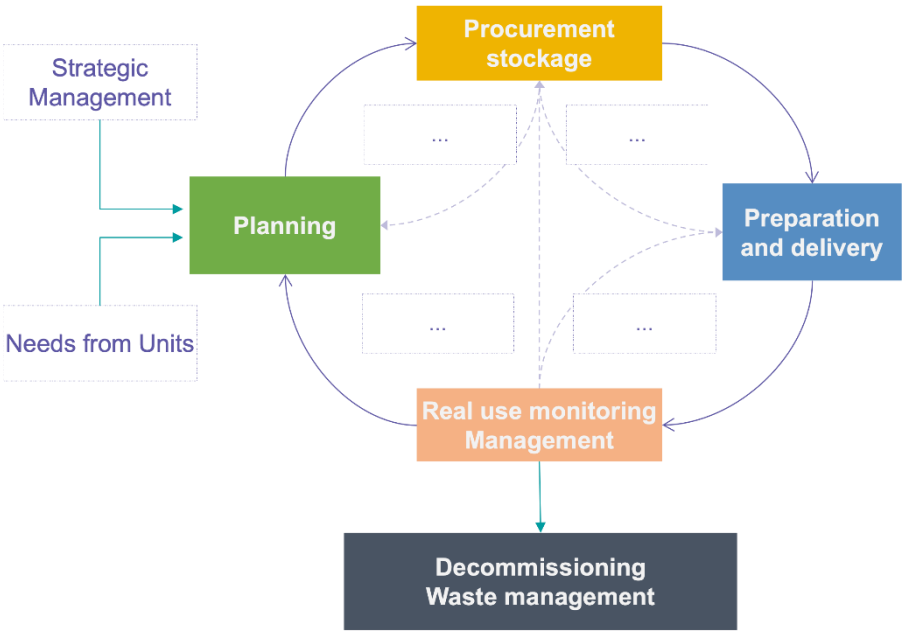

Figure 5.15. RUC B Phases workflow.

RUC B focuses on the logistics of the hospital. D7.1 provides a workflow model with different phases, called ODIN Framework for industry 4.0 hospital logistic management. We provide the model support this workflow.

5.3.3 RUC C – Disaster Preparedness

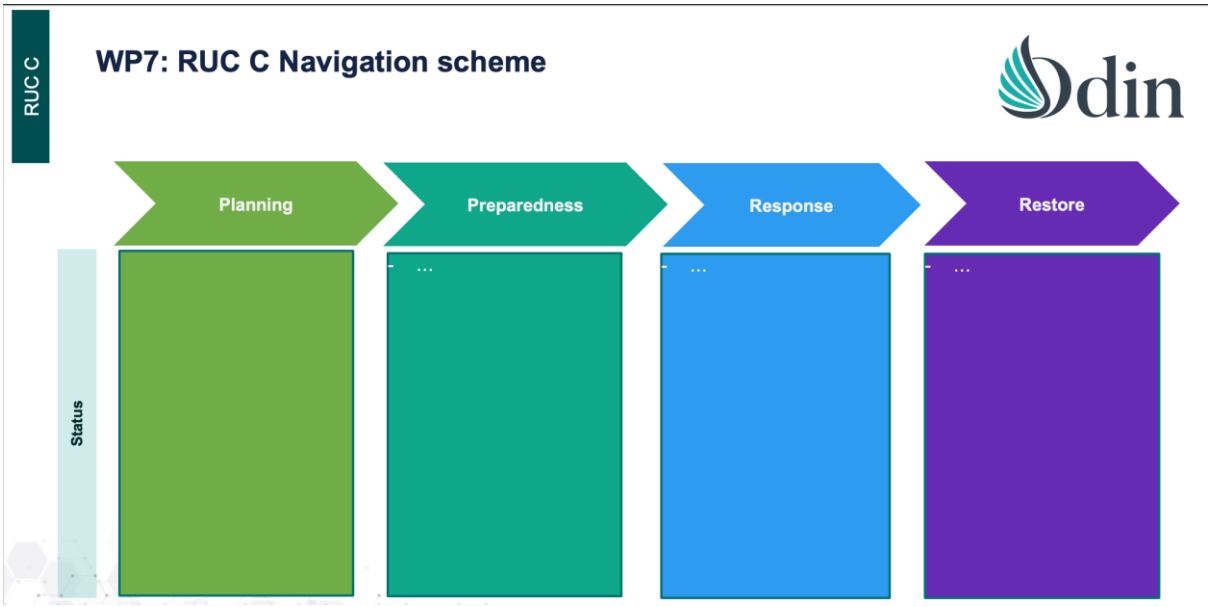

Figure 5.16. States of disaster preparedness.

WP 7 is still working on a unified model for RUC C, Figure 5.16 depicts the draft proposal for workflow model. Again, RUC C can be model as a state machine for different aspects of disaster preparedness.

5.3.4 KPIs

WP7 and T4.6 are working on a categorization of KPIs which will be modelled into the ontology as soon as the pertinent discussions are complete. The model will try to find existing generic KPI models for abstract models and complete with classes for the different KPI categories or hierarchy, as well as with concrete instances for identified KPIs.

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 6.1. Dataset collection ETL methodology.

The ETL (Extract, Transform, Load) methodology (see Figure 6.1) 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 section7), each phase will be explained in the following sections.

6.1 Data identification

The initial phase involves identifying the data that is accessible. Although we may already have knowledge of existing datasets, it is often surprising to discover data that is readily available and may have been overlooked as assumed facts. The goal of this phase is to thoroughly analyse the accessible data and generate a list of potential data that can 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 XLSX.
- 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 6-1 shows some examples by types.
- API endpoints, modern applications have APIs for interoperability, these APIs may be REST (Representional 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 type of data is labelled as unstructured data.
- Semantic formats (RDF, JSON-LD), in this case the data is already semantic.

Table 6-1. 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. For example, 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 is it 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 an 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 section2.5), 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 aide 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 and 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 Section6.1).

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 7.1.

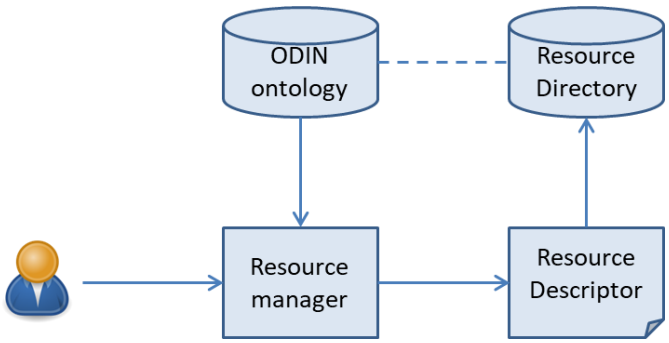

Figure 7.1. Resource registration process.

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 7.2. 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.

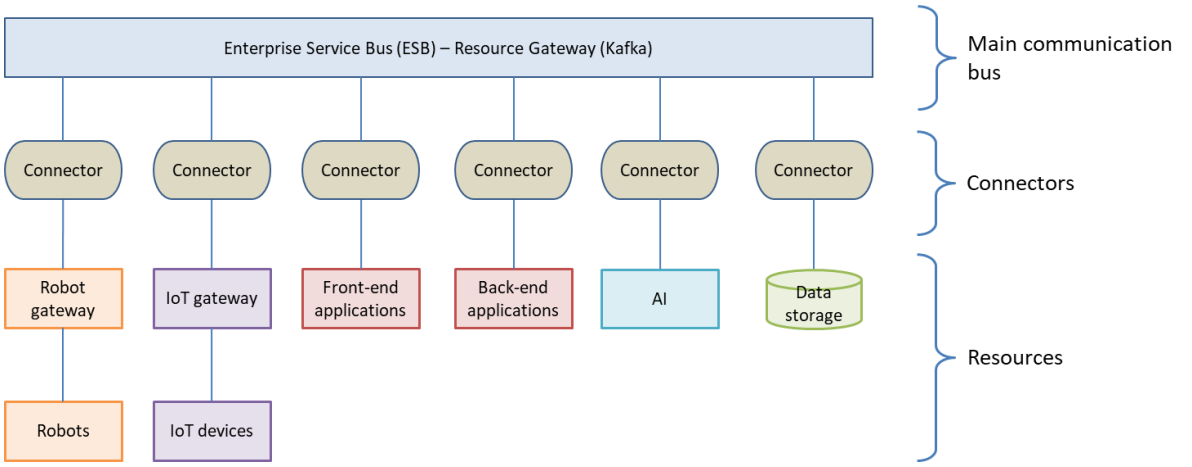

Figure 7.2. Resource communication within the ODIN architecture.

Each connector performs two levels of operations, as depicted in Figure 7.3: 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).

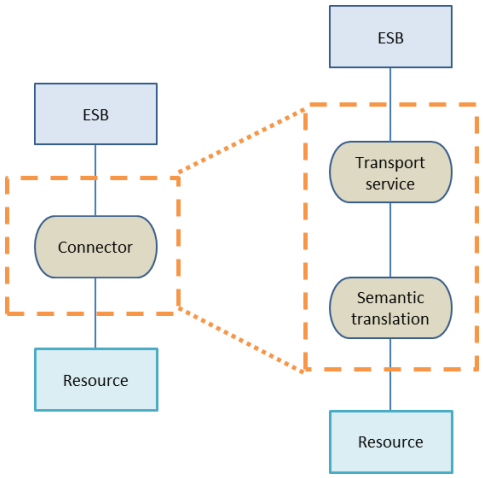

Figure 7.3. Each connector performs semantic translation and transport to the ESB message format.

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 and model, described in Section 2.4 and 4.9, 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 7.3 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 Section2.5). The tool then automatically transforms the input data to the output harmonized data model, according to the instructions, as shown in Figure 7.4.

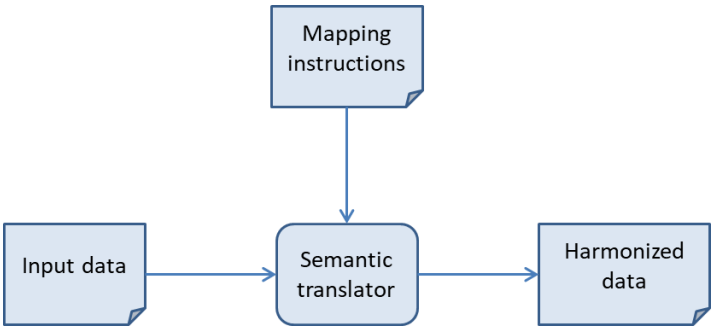

Figure 7.4. Automatic semantic translation and data harmonization process.

Table 7-1 lists a number of existing tools that perform automatic semantic translation, following the process of Figure 7.4. 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 7-1. Existing tools for automatic semantic translation.

Name	Description	Input formats	Output formats
RMLMapper[28]	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

[28] RMLMapper, https://github.com/rmlio/rmlmapper-java

RMLStreamer[29]	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
CARML[30]	CARML is a Java library that transforms structured sources to RDF based on RML mapping.	Database sources, XML files	RML rules
Morph-KGC[31]	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

[29] RMLStreamer, https://github.com/RMLio/RMLStreamer

[30] CARML, https://github.com/carml/carml

[31] Morph-KGC, https://github.com/oeg-upm/Morph-KGC

SDM-RDFizer[32]	SDM-RDFizer interprets RML mapping rules to transform (un)structured data to RDF graphs.	CSV, JSON, RDB, XML	RML (TriplesMap)
RocketRML[33]	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 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).

[32] SDM-RDFizer, https://github.com/SDM-TIB/SDM-RDFizer

[33] 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

This section will provide a set of examples which will illustrate the different uses of the ontology in the context of ODIN. The ODIN Ontology has different purposes, as a data model its main purpose is to offer a common structure for which the ODIN platform, its components and KERs to be able to express different objects. The objective is that both the structure, and the objects that will be expressed are simple enough for every entity involved to create and interpret with precision the data that is exchanged, essentially the objective is Interoperability.

Because RDF objects can be expressed in different serializations, and because JSON is one of the most wide-spread serializations, all of the expressions will be using JSON-LD 1.1 as serialization standard.

8.1 Platform

The ODIN platform will be the primary employer of the ODIN Ontology, as this will ensure interoperability between all of its components as well as with the connected KERs. There are some of the usecases where the ODIN platform components (and KERs) will employ the ontology:

- Resource Descriptor, When KERs describe themselves to the platform or when they are inspecting other KERs, the data will be expressed using JSON-LD and the odin:KER as base class. Because there are different subclasses, depending on the KER, each will define the set of standardized properties for said class.
- Semantic resource discovery, The Resource Directory component will use the Previous Resource Descriptor and register it as an accessible KER by adding the RDF object to the triple store. In turn this allows for the Resource Directory, or any other component of the platform (or KER) to query using SPARQL, effectively enabling semantic resource discovery.
- Resource communication, core EBS channels will comply with ontology, this means all of the core messages of the platform, as well as most of the KER intercommunications, will be standardised, through the ontology; which will also enable semantic operations and reasoning over these communications. Additionally, because the ontology enables extension, non-core channels can be created where the data model is based on the ontology, and its extension. Additionally the Ontology has the capability of describing non-semantic channels, ensuring the KERs connecting to any channel are able to predict and interact with said channels.
- Resource Federation, Determining which KERs need to participate in federated operations, and in which way will require semantic reasoning. Together with the Resource Directory, the DLT federation component will be able to query, filter, and connect the different KERs according the stablished rules and access.
- Resource Choreographer, The Choreographer, through the querying capabilities of the Resource Directory will be able to semantically discover the potential KERs to operate with; additionally, the Choreographer will be able to determine which messages, and how to connect, in order to achieve a desired effect.

8.2 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

8.2.1 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 temperature 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. Fragmentation can be addressed through the employment of standardized formats, which are vendor independent, precise on the data delivery and self-explainable for uninitiated. The SSN ontology provides the framework to communicate perception in this standardised way and allow to then be processed un a uniform way.

Table 8-1. SSN perception example. This is an official sensor read example[34] of an IoT Observation, translated to JSON-LD, converted to use units of measure ontology and compressed to the minimal components.

```
{
  "@context": {
    "@base": "http://example.org/data/",
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "sosa": "http://www.w3.org/ns/sosa/",
    "time": "http://www.w3.org/2006/time#",
    "om": " http://www.ontology-of-units-of-measure.org/resource/om-2/"
  },
  "@id": "http://example.org/data/Observation/346345",
  "rdf:type": {
    "@id": "sosa:Observation"
  },
  "sosa:observedProperty": {
    "@id":"http://example.org/844818-0/BMP282/atmosphericPressure"
  },
  "sosa:madeBySensor": {
    "@id": "http://example.org/844818-0/BMP282"
  },
  "sosa:hasResult": {
    "rdf:type": {
      "@id": "om:Measure"
    },
    "om:hasNumericValue": 101936,
    "om:hasUnit": {
      "@id": "om:Pascal"
    }
  },
  "sosa:resultTime": {
    "rdf:type": {
      "@id": "time:Instant"
    },
    "time:inXSDDateTimeStamp": {
      "@value": "2017-06-06T12:36:13+00:00",
      "@type": "xsd:dateTimeStamp"
    }
  }
}
```

Table 8-1 depicts an example (which is derived from official SSN example[34]), an Observation is represented, which is the main concept behind the process of perception (measuring anything and producing data). This observation contains all the details about the measurement, from what it is about (sosa:observedProperty, in this case it is just an identification to another object), to the actual measurement (sosa:hasResult, in this case it is an om:Measure, but sosa:hasSimpleResult could be used to just report a number). Additional metadata and important contextual information is added such as when was the observation made (sosa:resultTime), and which sensor was used

[34] https://www.w3.org/TR/vocab-ssn/#iphone_barometer-sosa

to make the observation (sosa:madeBySensor, in this case it is just a reference to a sensor id, for which the full description could be stored in the Resource Directory of ODIN).

This example could be used as an example of message used in the ESB of ODIN. This message contains lots of information which could be used by ESB subscribers to parse, determine if they are interested in this particular observation, and then interpret it if so.

8.2.2 Automation

With a common ontology, and creating the right mechanisms, ODIN can minimize those problems. Through the use of SSN, 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).

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, performing actuations 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.

8.2.3 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 8.1, 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 they are not working, and the failure has not been detected.

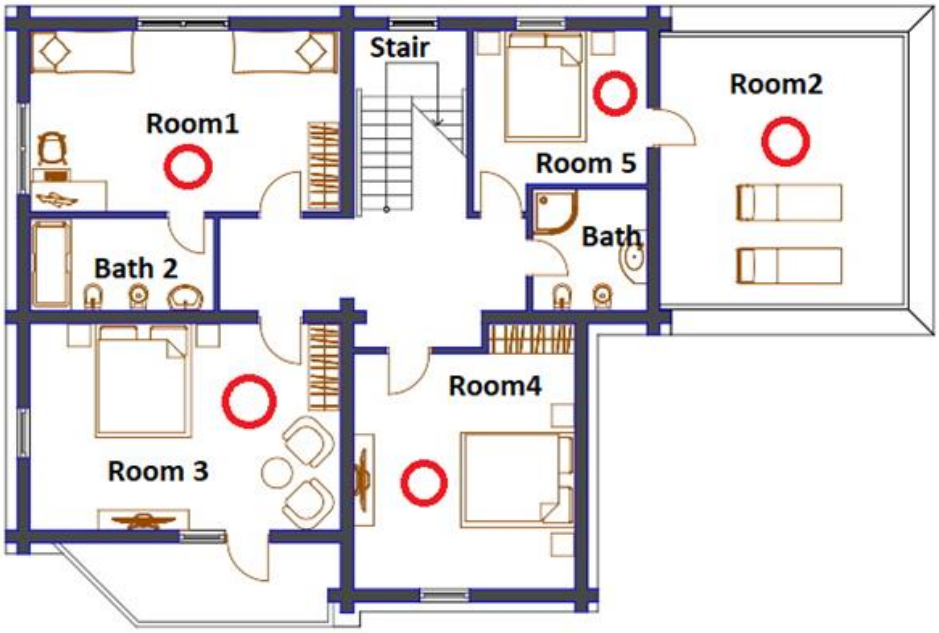

Figure 8.1. First floor of the house of a monitored patient.

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 wristbands 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 ODIN's format (see Table 8-2).

Table 8-2. JSON-LD example of RTLS relative position to the RTLS instance coordinate system.

```
{
  "@context": {
    "cora": "http://purl.org/ieee1872-owl/cora-bare#
    "xsd": http://www.w3.org/2001/XMLSchema#,
    "@vocab ": "http://purl.org/ieee1872-owl/pos#"
  },
  "@type": "cora:Robot",
  "@id": https://example.com/Robots/rob1
  "positionedAt”: {
      "@type": "PositionPoint",
      "inPCS": "https://example.com/Sensor/RTLS/installation1",
      "x": 354.2,
      "y": -24.7,
      "z": 0.0,

    }

}
```

The relative position 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, as well as the information that the tag is being carried by a person and the calculated position from the RTLS tag, 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 (after the RTLS reports the position of the tag and it is reasoned that the tag is carried by 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 (e.g. constructing a SPARQL query to retrieve the building Zone definitions with any additional condition required by the robot). 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, especially if there are several robots. This is also where quaternions, which represent orientation, may also be useful, as two robots at the same distance from the task may not be oriented in a way that their time to perform the task are equal (i.e. one has to turn before addressing the task the other does not).

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.

8.3 Robots

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 (dynamic knowledge, the Resource Descriptor component of the architecture) in order to provide information of the environment and related to both logistic aspect and environmental monitoring to dynamically update the context knowledge; 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.

An example of the first case, in particular to context reconnaissance is very similar to the IoT representation using SSN to represent the sensors of the robot platform and readings they report. The robot may however provide status report on their location, current activities, battery levels an other intrinsic data through the employment of the cora ontology.

Table 8-3 Simple CORA Robot description example.

```
{
  "@context": {
    "@vocab": "https://w3id.org/cora#",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "@type": "Robot",
  "@id" : https://example.com/Robots/rob1
  "name": "RoboBot",
  "hasComponent": [
    {
      "@type": "Sensor",
      "name": "Camera",
      "type": "Visual",
      "hasResolution": "1920x1080",
      "hasFieldOfView": {
        "@type": "FieldOfView",
        "horizontal": {
          "@value": "90",
          "@type": "xsd:float"
        },
        "vertical": {
          "@value": "60",
          "@type": "xsd:float"
        }
      }
    },
    {
      "@type": "Actuator",
      "name": "Gripper",
      "type": "EndEffector",
      "hasGrippingForce": {
        "@value": "20",
        "@type": "xsd:float"
      }
    }
  ],
  "hasCapability": [
    "ObjectRecognition",
    "Navigation"
  ]
}
```

An example of the downward case (from the knowledge base to the robot), could be the case where the robot needs to know the building planning in order to calculate the path to take to their objective.

Table 8-4. Simple SPARQL query to get all space adjacency, enough to build a navigation graph of the whole site.

```
PREFIX bot: <https://w3id.org/bot#>

SELECT ?space ?adjacentPlace
WHERE {
  ?space a bot:Space .
  ?space bot:adjacentZone ?adjacentPlace .
}
```

Table 8-5. Example output of space adjancency query.

space	adjacentPace
bt:Space1	bt:Space2
bt:Space1	bt:Space3
bt:Space2	bt:Space1
bt:Space2	bt:Space4
bt:Space3	bt:Space1
bt:Space3	bt:Space4
bt:Space4	bt:Space2
bt:Space4	bt:Space3

KERs may, through the platform communicate with the robots, to provide them with instructions, missions, or requests. A statement can be built to express these requests, which can then be interpreted by the robots, or the robot specific intelligence (such as the Fleet Management System) which will then enact the necessary actions. Mission statements can be identified by comparing the issued message with the current status, if they match, then this is a notification of the current status, if they don't (and it is issued by another, authorised, entity) then the message can be interpreted as the desired status, the request.

Because all the robotic technologies developed within WP5 uses two communication protocols, the robots will interact with the main ESB through 2 sets of connectors (see section 7.2), based on the transport service used:

i) Robotic Operating System (ROS) framework

ii) Message Queuing Telemetry Transport (MQTT – standard ISO protocol).

Both bridges will need to handle bi-directional communication, these connectors are developed in WP4. The second stage of the bridge, the semantic translation, is where the ontology plays a critical role, since for ODIN platform it is always preferrable for the kafka topics to deliver semantic statements in JSON-LD, which will employ the ontology to build the context for the JSON-LD based message.

This concept decouples the robots, and allows for future expansion with other transport protocols, as well as integration of non-ROS nor MQTT compliant robotic platforms. In addition it also allows the usage of other platform components such as the choreographer to coordinate robot operations in both data directions.

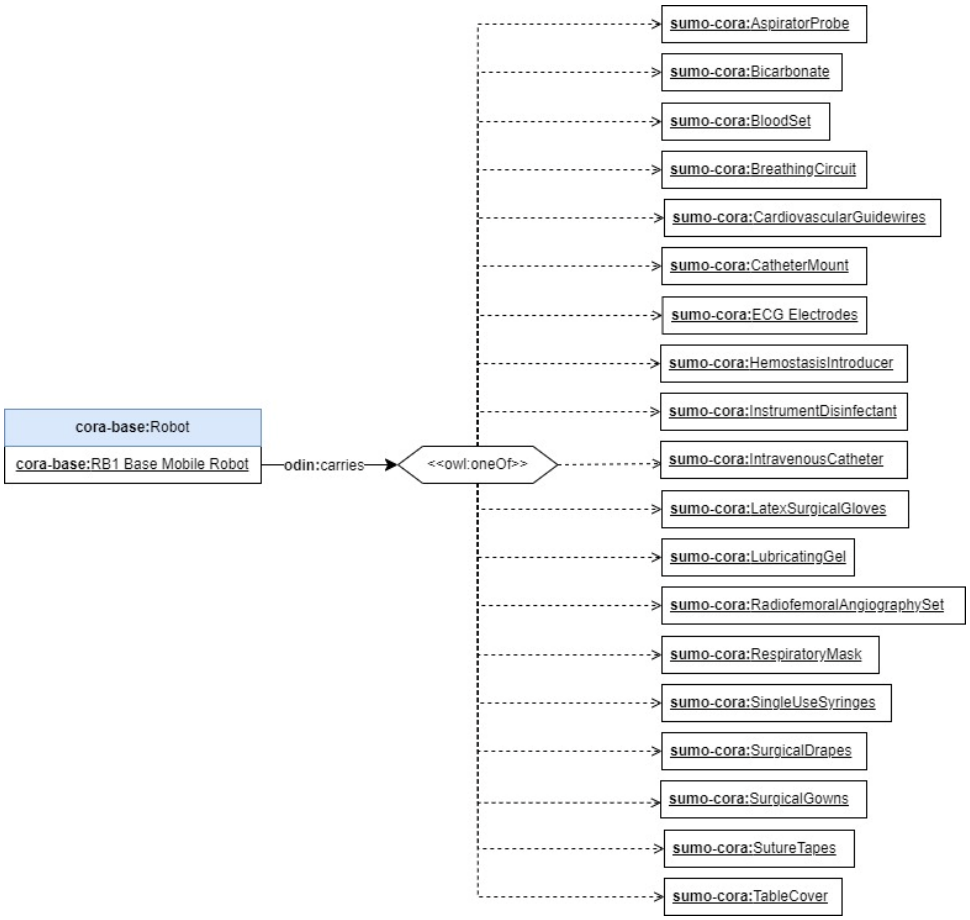

Figure 8.2. Example of RB1 robot being able to carry different things.

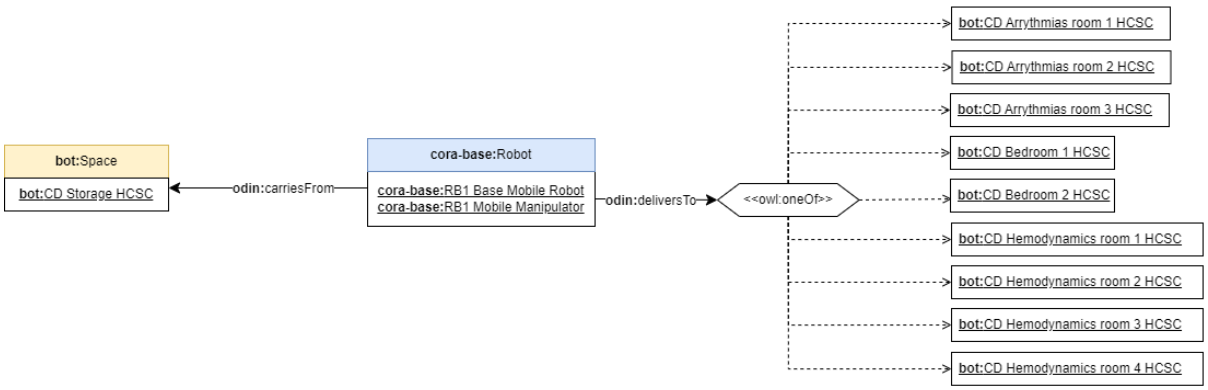

Figure 8.3. Example of simple mission definition.

8.4 AI

In the realm of Artificial Intelligence (AI), ontologies have emerged as a powerful tool for enhancing the development and effectiveness of AI solutions. Ontologies provide a structured representation of domain knowledge, capturing the concepts, relationships, and rules within a specific field. By incorporating ontologies into AI solutions, we unlock the potential to augment AI systems with a deep understanding of domain semantics, enabling them to reason, learn, and make informed decisions based on rich contextual knowledge. The use of ontologies facilitates knowledge sharing, interoperability, and data integration, paving the way for more accurate, interpretable,

and context-aware AI models. Leveraging ontologies in AI solutions not only improves the accuracy and efficiency of AI algorithms but also promotes explainability, transparency, and collaboration among different stakeholders. As a result, the synergy between ontologies and AI offers exciting opportunities to unlock the full potential of AI in various domains, including healthcare, logistics, emergencies, and beyond.

8.4.1 Training AI models

Ontologies can be used for training AI models in several ways:

1. Knowledge Representation: Ontologies provide a formal and structured representation of domain knowledge, capturing concepts, relationships, and rules. This knowledge can be used to train AI models by providing a foundation of domain-specific information. AI models can learn from the ontology to understand the semantics and relationships between different entities and concepts in the domain.
2. Feature Extraction: Ontologies can define important features or attributes of entities within a domain. AI models can utilize these features as inputs during the training process. By extracting relevant features from the ontology, AI models can learn to recognize patterns, correlations, and dependencies between different features, enabling more accurate predictions and classifications.
3. Training Data Generation: Ontologies can assist in generating training data for AI models. By leveraging the structured information in the ontology, synthetic or augmented training datasets can be generated to supplement existing datasets. This can help address issues like data scarcity, data imbalance, or the need for diverse data samples, enhancing the model's training process.
4. Labelling and Annotation: Ontologies can provide a standardized framework for labelling and annotating data. By aligning the data with the concepts and relationships defined in the ontology, AI models can be trained using labelled data that adheres to a consistent and predefined schema. This improves the quality and consistency of training data, leading to more robust and accurate models.
5. Transfer Learning: Ontologies can provide a transferable knowledge structure that can be shared across different AI models or tasks. Pre-trained ontologies can serve as a knowledge base or a starting point for training new AI models. The pre-existing knowledge in the ontology can be transferred to the new model, enabling faster learning and reducing the need for extensive training on large datasets.

By incorporating ontologies into the training process, AI models can leverage the structured knowledge and semantic representations to enhance their learning capabilities, improve data quality, facilitate data integration, and enable knowledge transfer. This can lead to more accurate, interpretable, and domain-aware AI models.

8.4.2 Data Integration and Fusion

Ontologies can serve as a common framework for integrating and fusing data from various sources, such as electronic health records, wearable devices, and medical imaging. AI models can leverage the ontology-based data integration to gain a holistic view of patients' health status and derive valuable insights.

Thanks to the semantic description of the different KERs, AI modules could query the Resource directory to find compatible datasets. Compatible datasets are those whose structure, content and purpose allow the AI to operate, in this way the AI models, supported by the platform could be easily used, i.e. plug and play.

One of the most interesting datasets which could be used to enrich AI models is the historic component, this component stores all the messages that the ESB has handled. With this information AI models could fusion with their training set in order to provide further context to it. This fusion will help AI models to be more precise, as well as providing better insights and expandability of the results.

8.4.3 Context-aware Decision Making

Ontologies can capture contextual information about patients, healthcare providers, and other relevant entities. AI models can leverage this contextual knowledge to make more informed and personalized decisions, such as treatment recommendations, diagnosis support, and patient monitoring.

In the case of ODIN AI, AI context awareness could be used to perform self-assertion. This is the capability of the AI software to recognize the available data, sources and even services, to self-configure, connect, feed and train, to provide fully automatic AI operation. This would be performed thanks to the semantic information that the ontology provides as well as the semantic description of all the resources registered in the platform which provides all the additional knowledge to perform this kind of decision making.

8.4.4 Supporting Explainable AI

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.

8.4.5 Semantic Interoperability

Ontologies can provide a shared understanding of concepts and relationships within a domain. This enables interoperability between different AI models and systems, allowing them to exchange and integrate data and knowledge seamlessly.

Specifically, the ODIN ESB, when using the ontology for data models, provides intrinsic semantic interoperability for AI models in two ways. With interoperable messages, AI will be able to receive information and automatically interpret it, when any KER produces information that is relevant to the AI algorithm, the AI may receive it and automatically execution

8.4.6 Ontology enrichment with 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.

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

8.4.7 Other health oriented semantic AI Use cases

ODIN already provides AI tools and models for ODIN pilots to operate. Furthermore, the ODIN platform enables and promotes the creation and connection of AI models and algorithms, some of which may be more semantically oriented. The following are some examples of such systems which may be interesting in the context of ODIN where the ODIN ontology may play a fundamental role in their operation.

Clinical Decision Support Systems: Ontologies can be used to represent medical guidelines, best practices, and clinical knowledge. AI models can utilize these ontologies to develop intelligent decision support systems that assist healthcare professionals in making accurate diagnoses, suggesting appropriate treatments, and avoiding medical errors.

Patient Risk Assessment: By utilizing ontologies, AI models can assess and predict patient risk factors for various conditions and diseases. The ontology can capture relevant information such as patient demographics, medical history, genetic factors, and lifestyle choices, enabling AI models to identify individuals at higher risk and recommend preventive measures.

Clinical Trials and Research: Ontologies can facilitate the design and execution of clinical trials by providing standardized representation of trial protocols, eligibility criteria, and outcome measures. AI models can utilize ontologies to identify suitable patient cohorts for clinical trials, support trial data analysis, and accelerate the discovery of new medical treatments.

8.5 Dataset

Ontologies can play a crucial role in describing, registering, cataloguing, and discovering datasets held in hospitals.

Ontologies provide a structured framework to describe and define the properties, characteristics, and metadata associated with datasets. By using ontologies, hospitals can create standardized descriptions of their datasets, including information such as dataset name, creator, creation date, format, size, variables, and any relevant domain-specific attributes. This enables consistent and comprehensive documentation of datasets, making it easier to understand and interpret their content. The DCAT ontology can be used as basis for providing this structured and interoperable metainformation about data sets, see Table 8-6 as an example of a this expression.

Table 8-6. JSON-LD example of a DCAT Descriptor for a Dataset, which can be mapped to a KER.

```
{
  "@context": {
    "@vocab": "http://www.w3.org/ns/dcat#",
    "dct": "http://purl.org/dc/terms/",
    "foaf": http://xmlns.com/foaf/0.1/,
    "vcard": "http://www.w3.org/2006/vcard/ns#"
  },
  "@type": "Dataset",
  "@id": "http://example.org/datasets/xyz-hospital-records",
  "dct:title": "Hospital Patient Records",
  "dct:description": "Anonymized patient records from XYZ Hospital.",
  "dct:issued": "2023-05-01",
  "dct:modified": "2023-05-10",
  "theme": {
    "@id": "http://example.org/themes/healthcare"
  },
  "keyword": [
    "patient records",
    "healthcare",
    "anonymized data"
  ],
  "distribution": [
    {
      "@type": "Distribution",
      "dct:title": "CSV Format",
      "mediaType": "text/csv",
      "downloadURL": "http://example.org/datasets/xyz-hospital-records/csv"
    },
    {
      "@type": "Distribution",
      "dct:title": "RDF Format",
      "mediaType": "application/rdf+xml",
      "accessURL": "http://example.org/datasets/xyz-hospital-records/rdf"
    }
  ],
  "dcat:publisher": {
    "@type": "Organization",
    "foaf:name": "XYZ Hospital"
  },
  "dcat:contactPoint": {
    "@type": "vcard:Contact",
    "vcard:email": "contact@example.org"
  }
}
```

Additionally, the registration process for datasets held in hospitals can be supported. By defining ontology-based registration forms, hospitals can capture important information about the dataset use, such as its source, access restrictions, associated projects, ethical considerations, and data usage policies. The ontology can enforce data quality standards and ensure that all necessary information is provided during the registration process as a KER, promoting transparency and compliance with data governance guidelines. To register a data set, the dataset description should be used complemented with the KER mandatory fields, in the case of Table 8-6, it could be used as is, as the ontology already includes the mapping of rdf:label, rdf:comment, and odin:channels to dct:tilte, dct:description and dcat:distribution respectively.

Table 8-7. A SPARQL example which lists the datasets containting anonymized in its keywoards and are distributed using csv.

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset
WHERE {
  ?dataset a dcat:Dataset ;
           dcat:keyword ?keyword ;
           dcat:distribution ?distribution .

  ?distribution a dcat:Distribution ;
                dct:title ?title ;
                dcat:mediaType "text/csv" .

  FILTER (CONTAINS(LCASE(?keyword), "anonymized"))
}
```

Ontologies can be used to create catalogues or repositories that organize and index the available datasets within a hospital. The ontological representation allows for classifying datasets based on various criteria, such as data type, medical specialty, patient demographics, data source, or research purpose. This facilitates efficient dataset management, retrieval, and exploration, making it easier for researchers, clinicians and even AI to identify relevant datasets for their specific needs. Catalogues also enable discovery, by providing a semantic search and matching mechanism. Researchers, clinicians and even automatic processes can query the dataset catalogue, which is stored in the Resource directory, using domain-specific terms and relationships defined in the ontology, through a SPARQL query (see Table 8-7 as an example). This allows them to find datasets relevant to their research or clinical needs based on the semantic meaning of the data. Ontology-based dataset discovery can enhance the efficiency of data reuse, foster collaboration, and encourage interdisciplinary research within hospitals.

Ontologies facilitate interoperability and integration of datasets across different hospital systems. By aligning the ontological representation of datasets with existing healthcare standards and ontologies (e.g., HL7, FHIR), hospitals can achieve seamless data exchange and integration between different healthcare applications and research platforms. This promotes data sharing, secondary use of data, and enables comprehensive analysis and decision-making based on integrated datasets.

9 CASE STUDIES

To effectively utilize the ODIN ontology, the process involved mapping the imported ontologies and the ontology's new classes with the specific Use Cases (UCs) outlined in each pilot of the project. This approach enabled the identification of the ontologies and classes that play a role in implementing each UC within the pilots.

Table 9-1 shows the ontologies uses in each UC implemented in the project's pilots. It is evident that the UCBM pilot employs the same set of ontologies across all its use cases. Similarly, the UCs within the UMCU pilot also rely on identical ontologies. In contrast, the remaining pilots exhibit greater diversity in terms of the ontologies utilized within each use case.

Finally, the imported ontologies and classes integrated into the ODIN ontology effectively address the requirements of the UCs in terms of data and management aspects.

Table 9-1. Use of ODIN ontology in pilots UCs.

	ORG	BOT	NCIT	ICD9	SNOMED	FHIR	AI	DCAT	CORA	SSN
SERMAS										
Stents Administration										
Transportation of stents										
Disasters preparedness										
UCBM										
Undernutrition										
Mobility loses										
Oxygen monitoring										
Food logistic										
CUB										
Sleep scoring										
Sleep monitoring										
Patients monitoring and evacuation										
UMCU										
Personalized treatment CVD										
CVD patient identification										
Postoperative telemonitoring										
Pathogens overview for IPD										
MUL										
Blood sample transportation										
Indoor location of medical devices										

9.1 HOSPITAL CLINICO SAN CARLOS (SERMAS)

SERMAS - Servicio Madrileno 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 Clinico San Carlos (HCSC) in the ODIN project. The San Carlos Clinical Hospital consists of seven institutes:

- "Josè 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 [46].

9.1.1 DATASETS

The pilot conducted at San Carlos Clinical Hospital involved several datasets, which are listed below along with their relevant information:

- MSCOCO dataset: for object detection.
- Stents' purchase and consumption information: information about the stents that were used.
- Patient hospitalization, emergency and outpatient episode information: hospitalization, emergency and outpatient episode information.
- Variables relating the stents, patients and patient episodes: provides information about patients and stents.

Table 9-2 presents information about the datasets involved in the SERMAS pilot in terms of data format and volume, the physical location of the datasets and their access conditions, and whether the dataset is GDPR compliant and contains sensitive medical content.

Table 9-2. SERMAS datasets.

Datasets	Format	Volume	Location	Access conditions	GDPR compliant	Sensitive content
MSCOCO	Image	19 Gb	SERMAS Laboratory server	Not accessible	Yes	No
Stent’s purchase	xlsx	100 Kb	Shared file with SERMAS	On request	Yes	Yes
Patient hospitalization	xlsx	60 Kb	Shared file with SERMAS	On request	Yes	Yes

Variables relating the stents	xlsx	60 Kb	Shared file with SERMAS	On request	Yes	Yes

9.1.2 FLOOR PLANS

The Haemodynamic Unit and Emergency Room maps have been supplied. The floor plans of the Haemodynamic Unit include a description of each room based on its use (Figure 9.1). The Emergency rooms plans have been numbered to make them easier to find, as seen in Figure 9.2 and Figure 9.3.

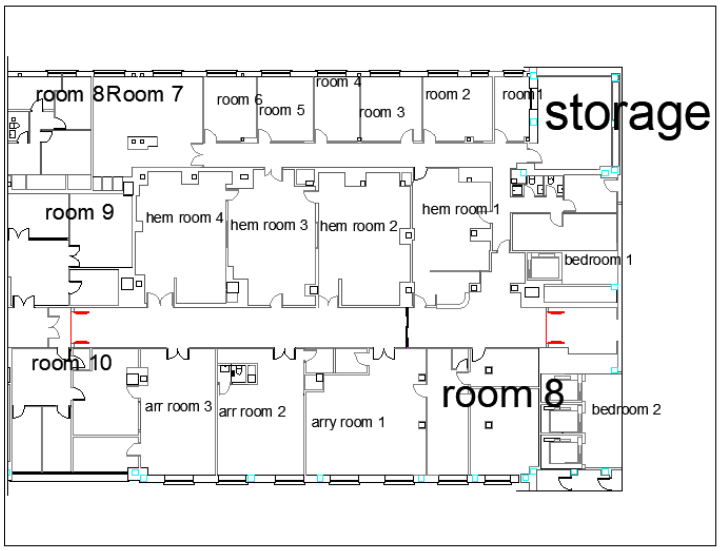

Figure 9.1. Haemodynamic Map.

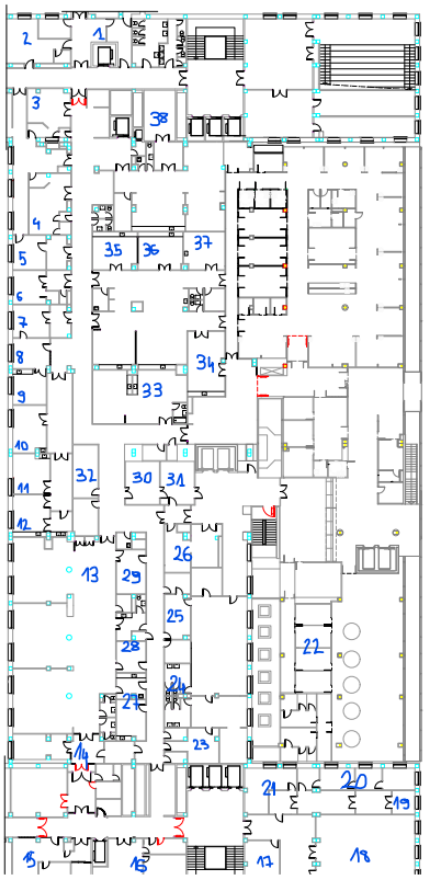

Figure 9.2. Emergency Map 1.

Figure 9.3. Emergency Map 2.

9.1.3 ORGANIZATION CHART

The organization chart for San Carlos Clinical Hospital was inferred from the official HCSC website. The intern organization (with the Heads of each unit) of the Cardiology and Emergency department is shown below. As a result, the heads of the departments and units involved were added to the ODIN ontology.

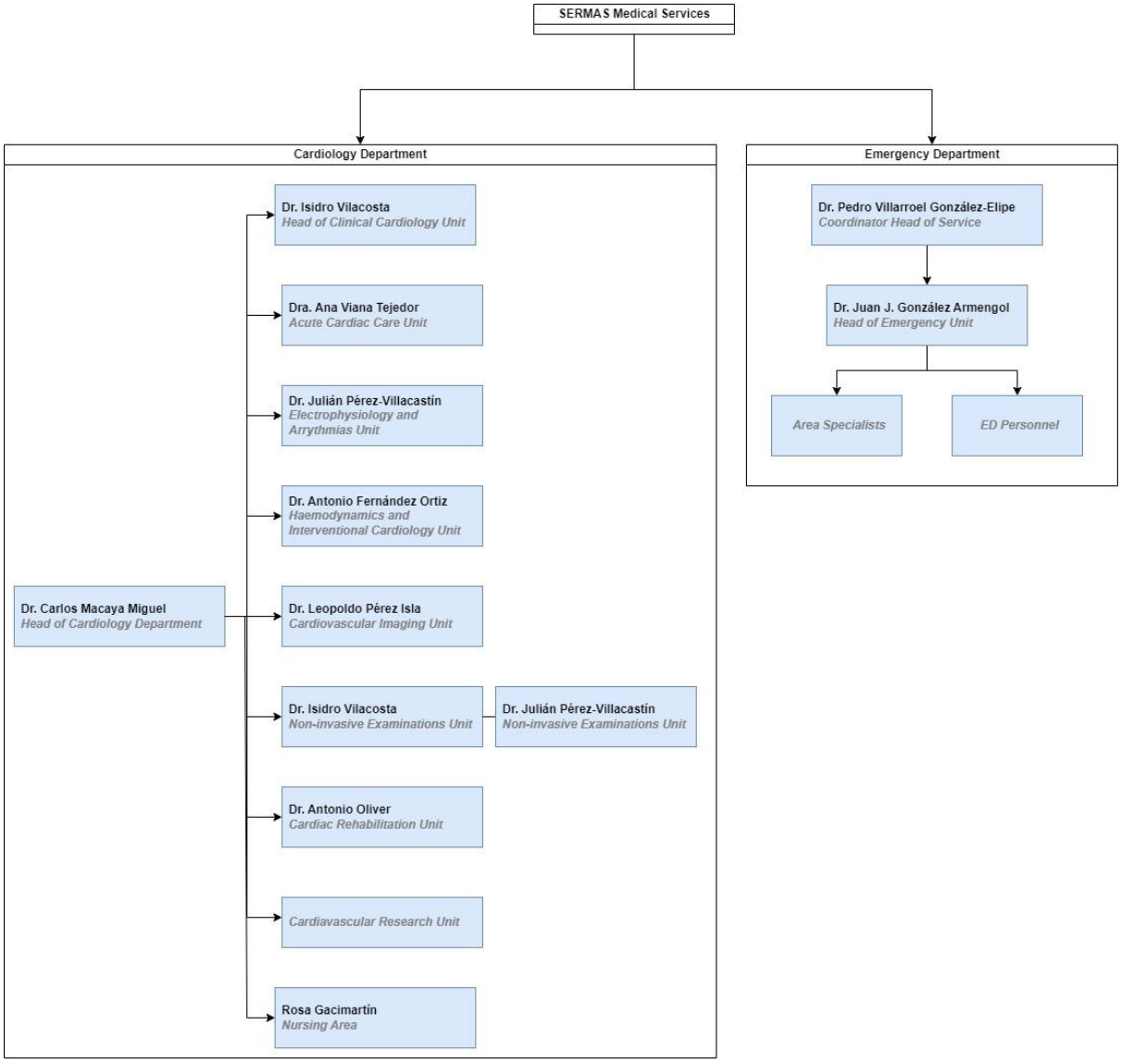

Figure 9.4. SERMAS Cardiology and Emergency Departments.

9.2 CHARITÉ UNIVERSITY HOSPITAL (CUB)

Charité University Hospital is a large hospital in Berlin with an Interdisciplinary Sleep Medicine Centre, providing care for all sleep disorders. It has an outpatient clinic, an inpatient sleep laboratory, and an outpatient sleep laboratory for patients with sleep-related breathing disorders. The staff includes specialists and nurses. Charité aims to combine patient care, medical research, and education. It uses experiments to test and implement new technology to improve patient care, focusing on improving organizational, environmental, and economic factors. For example, it is testing new devices as an alternative to the gold standard polysomnography to reduce the waiting time for patients. Charité is also exploring the use of machine learning algorithms to automate sleep scoring and improve disaster preparedness services.

9.2.1 DATASETS

The datasets involved in the UCs carried out in the Charité pilot mainly include data related to polysomnography (PSG), also known as a sleep study.

- Berlin ESADA data: provide raw signals (PSG) and sleep parameters (PSG).
- Insomnia Dataset: provide raw signals (PSG) and sleep parameters (PSG).
- Finger ring Dataset: provide raw signals (PSG) and sleep parameters (PSG and Pulse Oximeter).

Table 9-3 includes detailed information about the Charité datasets related to the data format and volume, the physical location of the datasets and their access conditions, and whether the dataset is GDPR compliant and contains sensitive medical content.

Table 9-3. Charité datasets.

Datasets	Format	Volume	Location	Access conditions	GDPR compliant	Sensitive content
Berlin ESADA	EDF / EDF+	65GB	Charité sleep lab	Access to members	Yes	Yes
Insomnia	EDF / EDF+	15GB	Charité sleep lab	Access to members	Yes	Yes
Finger ring	EDF / EDF+	15GB	Charité sleep lab	Access to members	Yes	Yes

9.2.2 FLOOR PLANS

Figure 9.5 depicts the floor plan of the sleep labs where wards are available for overnight studies. The floor plan includes a legend to identify each part of the floor (e.g. hallway, room, etc.). This information is necessary for the robot guidance that will be used for this use case.

Figure 9.5. A floor plan of the sleep laboratory at the Charité Virchow Klinikum.

9.2.3 ORGANIZATION CHART

The organization chart (Figure 9.6) for the Interdisciplinary Sleep Medicine Center at Charité-Universitätsmedizin Berlin was inferred from the official CUB website. Since the Sleep Medicine Center is the only department involved in ODIN Project, it will be the only analysed. The team is composed by a Management Team, an Administrative Team, Physicians and Researchers, Technical Assistants and Patient Management Team as shown below.

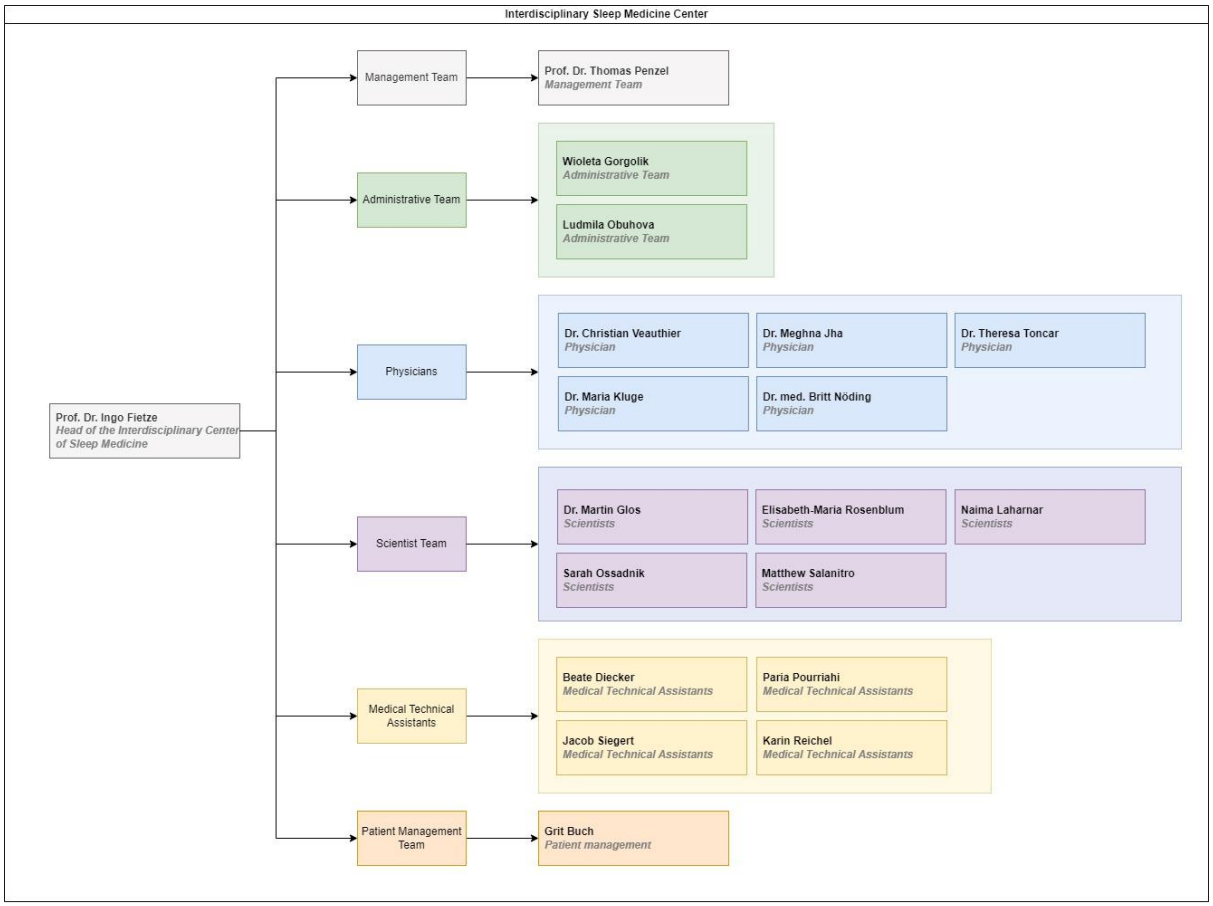

Figure 9.6. CUB Sleep Medicine Center Organigram.

9.3 UNIVERSITY MEDICAL CENTER UTRECHT (UMCU)

The University Medical Center Utrecht (UMCU) is a leading medical center in the Netherlands, known for providing specialized healthcare services, conducting cutting-edge research, and offering education to healthcare professionals and students. As part of the ODIN project, UMCU's Division Laboratories, Pharmacy and Biomedical Genetics collaborates with the Center for Circulatory Health to improve patient care for cardiovascular diseases. UMCU's translational subunit, ARCADIA, houses the Utrecht Patient Oriented Database (UPOD), providing access to electronic health records of all UMCU patients since the 1990s. UPOD aims to improve clinical diagnostics using routine care data and is part of the UMCU's transition into a learning healthcare system. The ODIN project focuses on developing clinical decision support systems within this system, using UMCU's expertise in pre- and clinical research and collaborations with companies and institutions worldwide.

9.3.1 DATASETS

To facilitate the implementation of the use cases (UCs) outlined in the UMCU pilot, various components of the UPOD database will be leveraged. This will involve gathering and utilizing data and information from diverse fields within the database, such as Agenda, Laboratory, Medication, Measurement, Diagnosis and Operation information, and Financial/Diagnosis billing code.

9.3.2 FLOOR PLANS

UMCU's experiments do not involve the use of robots, therefore, floor plans of the hospital areas involved are not necessary for the experiments.

9.3.3 ORGANIZATION CHART

ODIN Project is part of the Circulatory Health theme, one of the six strategic themes at UMC Utrecht research.

The project is enclosed at the Central Diagnostic Laboratory (CDL) as part of the "Laboratories, Pharmacy, and Biomedical Genetics" division, one of ten divisions that constitutes Healthcare at UMCU. The translational subunit ARCADIA (Academic Research for Clinical Applications of Diagnostics) hosts the Utrecht Patient-Oriented Database (UPOD). The described organization can be shown in Figure 9.7.

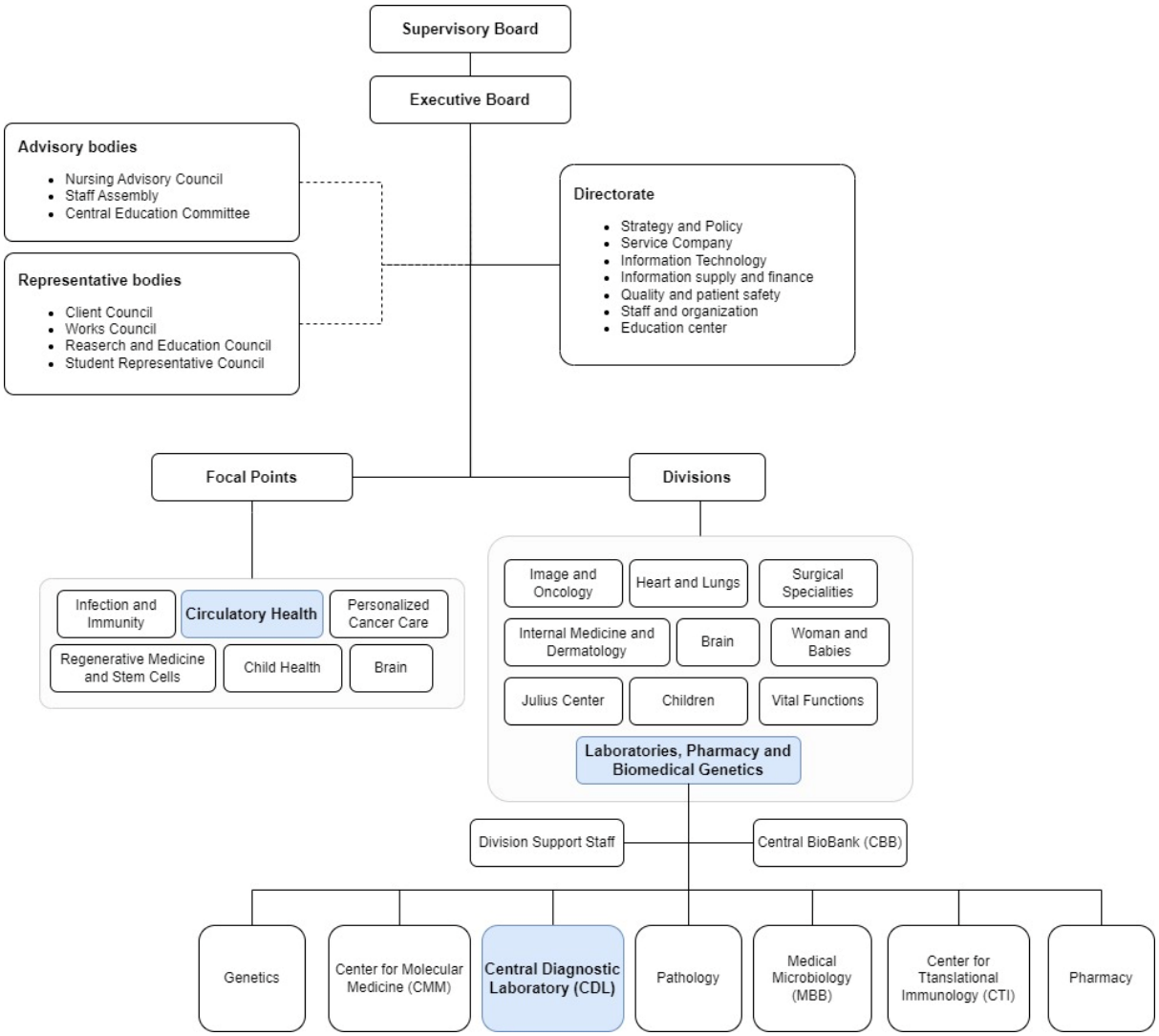

Figure 9.7. UMCU Laboratory Department Organigram.

9.4 UNIVERSITÀ CAMPUS BIO-MEDICO DI ROMA (UCBM)

The Università Campus Bio-Medico di Roma (UCBM) is a private academic institution focused on undergraduate and postgraduate education, advanced research, and third mission. UCBM has 51 multidisciplinary research units and hosts various PhD courses in different fields, including Bioengineering, Applied Sciences, Intelligent Systems, Integrated Biomedical Sciences, Bioethics, and Robotics. It also has an outstanding network of national and international key scientific and educational partners. UCBM has devolved the Policlinico Universitario Campus Bio-Medico business unit to the Fondazione Policlinico Universitario Campus Bio-Medico (FPUCBM), a not-for-profit institution that focuses on healthcare, training, scientific research, and innovation in the biomedical and health fields. FPUCBM is hosting 60 Research Operative Units and 10+ laboratories. The Geriatrics Unit within FPUCBM conducts research activities in different areas, including the evaluation of elderly patient health conditions, respiratory functions, the development and application of remote telemonitoring systems for patients with chronic diseases, and pharmacoepidemiologic and epidemiologic geriatric research.

9.4.1 DATASETS

The datasets used in the UCs conducted in the UCBM pilot mainly consist of data pertaining to food and nutrition, rehabilitation, and oxygen monitoring. Table 9-4 contains detailed information about the datasets, including data format and volume, physical location, access conditions, GDPR compliance, and whether the dataset contains sensitive medical content.

The and the type of data they contain are listed below:

- Oxygen mask positioning: Images for detecting the correct positioning of the oxygen mask.
- Rehabilitation videos: Videos of UCBM members executing prescribed rehabilitation exercises.
- Motion data during rehabilitation exercises: Quaternion orientation, angular velocity and linear acceleration information.
- Mediterranean Greek Food (MedGRFood): Nutritional information.

Table 9-4. UCBM datasets.

Datasets	Format	Volume	Location	Access conditions	GDPR compliant	Sensitive content
Oxygen mask positioning	JPG	40 GB	Forth server	Not accessible	Yes	No
Rehabilitation videos	MP4	14.5 MB	Shared file with UCBM	Access on request	Yes	No
Motion data	CSV	880 KB	Shared file with UCBM	Access on request	Yes	No
MedGRFood	JPG + JSON	5 GB	Forth server	Access on request	Yes	No

9.4.2 FLOOR PLANS

UCBM's pilot floor plans are currently being developed and will be implemented once finalized. These blueprints play a crucial role in ensuring that the robots involved in the pilot function properly.

9.4.3 ORGANIZATION CHART

The organization chart for Campus Bio-Medico University Hospital was obtained from the official website. The intern organization is displayed below.

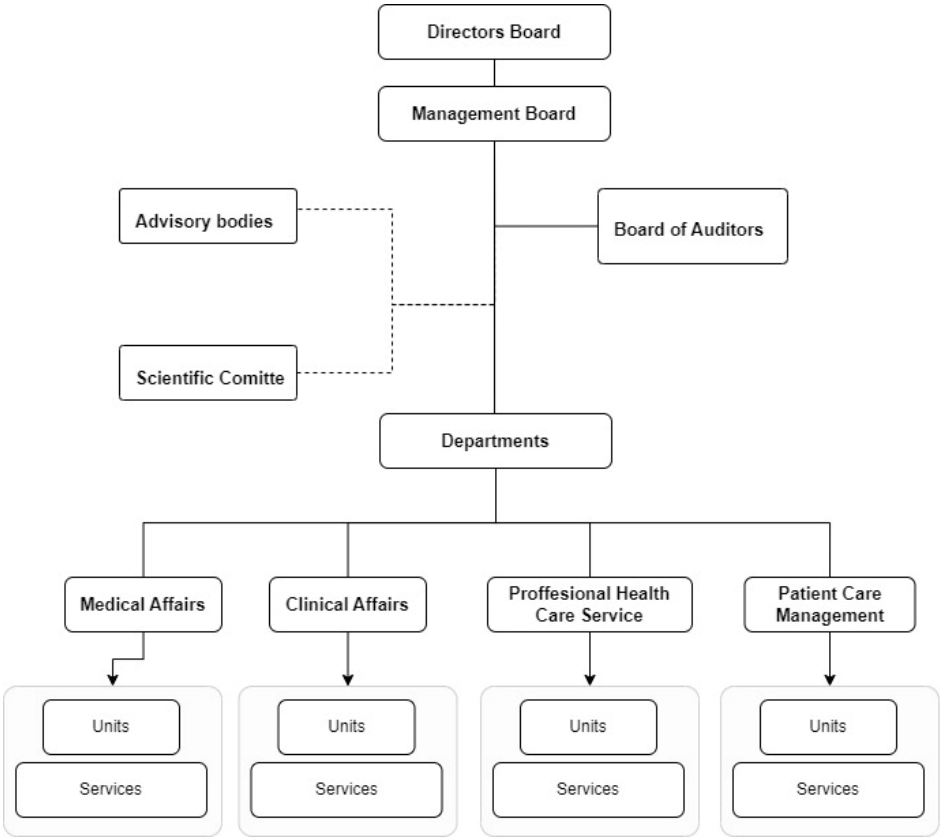

Figure 9.8. UCBM Hospital Organigram

10 Ontology Governance model

Because this is the last deliverable version reporting on the ODIN ontologies, and because these ontologies will need to continue to evolve and adapt to the evolving needs of ODIN platform, its associated KERs as well as the use cases they are applied to, there is the need to have in place a robust sustainability strategy. Ontology governance is the process of managing the development, maintenance, and use of ontologies in the community. It involves defining policies, procedures, and roles for ensuring the quality, consistency, and interoperability of ontologies over time.

The most common governance models for ontologies are:

1. *Centralized governance model*: In this model, a central authority is responsible for developing, maintaining, and managing the ontology. This can provide a high degree of control and consistency, but may limit participation and collaboration from other stakeholders.
2. *Federated governance model*: In this model, multiple entities or groups share responsibility for developing, maintaining, and managing the ontology. This can promote collaboration and participation, but may require more coordination and communication.
3. *Community-based governance model*: In this model, the ontology is developed and maintained by a community of users and stakeholders. This can promote collaboration, participation, and innovation, but may require more coordination and consensus-building.
4. *Hybrid governance model*: In this model, elements of centralized, federated, and community-based governance are combined to suit the needs of the organization or community. For example, a central authority may oversee the overall development and quality assurance process, while allowing for community-based contributions and feedback.

The selection of the governance model which best suits the ODIN ontology is heavily dependent on the sustainability and exploitation strategies, as well as the success of the ODIN platform as a whole. Therefore, the governance model will also need to change and adapt to the structures which support it. Initially, the governance will be centralised through the work of Task 3.2 throughout the project; when the project ends, and depending on the transference, the governance model will shift to a community driven approach. Either way it is important to communicate properly the governance model and procedures so that all stakeholders are aware of the latest operative, especially when changes are required.

The governance framework includes the policies, procedures, and guidelines for managing the ontology. The main processes of the framework are the development process, quality assurance process, version control, and documentation. All of these will be tackled as any other software product, therefore the governance framework will be based on the DevOps processes for the platform, as described in D3.1. The ontology source, the OWL files containing all the axioms describing it, as well as additional documentation, examples, templates, tests, and any other supplementary material will be uploaded to a git repository, hosted in the Gitlab server of ODIN platform, if at the end of the project the repositories are moved to a public hosting platform such as GitHub, then the ontology will move too. In either case, the repository manager already includes relevant tools for developing (git), documenting (wiki), contributing (pull request), reporting (issues), planning and even debating about the ontology.

As part of the DevOps for the Ontology, a pipeline will be developed to automatically validate the ontology (using tools such as HermiT or oops[35]); and once it has been verified the pipeline can also generate the documentation (using the Widoco tool[36]) and finally publish both ontology and documentation to be publicly available under the appropriate URL.

In this way, the documentation is always accessible online in the same way the syntax and schemas are accessible too, following this principle it must be guaranteed that the latest version of the data model is online and ready to be used. This procedure of data model files online hosting is a good practice in W3C semantic web communities in order to guarantee the sharing characteristic of the data models globally.

One of the key aspects the ODIN ontology V2 is being updated is the application of a simple versioning method that will report improvements and extensions, thus the versioning system will reflect any update and/or modifications.

10.1 Ontology Update Process

The ODIN Ontology maintenance procedure follows the design idea of data modelling updates and corrective activities continuously as well as covering the range of support services centralised through the issue management tool. This activity was identified as crucial after several interactions with the development team and with the pilots' technical experts where it is expected to have not only a support/help to clarify doubts concerning the data model but also to provide an efficient management tool for controlling updates and proposing possible changes in the model.

The ontology update process is a point of contact for stakeholders (users, pilot managers, Open Callers, Developers, community at large) to request support in the form of issue, formally request changes, or even contribute changes on the form of Pull Requests (see Figure 10.1). The issue management system notifies the ODIN experts team (Week1) who can track and organize issues by topic and priority (using tags) assigning them to individual team members, for faster resolution. If more information is required two side communication is stablished (Week 2 – Week 3). The issue management allows agents to mark resolved issues (by closing them), split issues, if necessary (by generating new ones), or even to escalate transforming issue into formal change requests (by adding the corresponding tag).

Change requests and contributions will be evaluated against ODIN ontology criteria (specified in this document as well as in the documentation in the repository), as well as validated with participating hospitals (Week 4-7). Along this line, the ontology update process's main idea is that a system can track and understand how ODIN experts are managing and interacting with the different requests about Ontology updates and changes and perform a correct documentation (Week 8). So, when a Change is rejected, it can properly be explained why.

The main task of documenting correctly the ontology (Week 8) and facilitate the Ontology editing process by having all the updates and changes clearly identified (Week 9 - Week 11). Finally, the process finishes when it is considered that the Ontology update is ready and then the new version

[35] https://oops.linkeddata.es/

[36] https://github.com/dgarijo/Widoco

of the ontology including schemas, files, examples, diagrams and documentation is released (Week 12) if a final automatic evaluation is accepted.

The whole process takes time, at least 2 weeks for simple support question and up to 7 for complex support (if no new issues or changes are required). Changes to the ontology can take anywhere from 6 weeks to 12 weeks, depending on the level of change required. Updates can be parallelized, meaning the team can work in multiple change requests at once releasing all of them in the same version. This of course requires coordination and synchronization between issues and teams, paying special attention to those requests which may have dependencies. Thus, the overall time between releases could be extended beyond the 12 weeks.

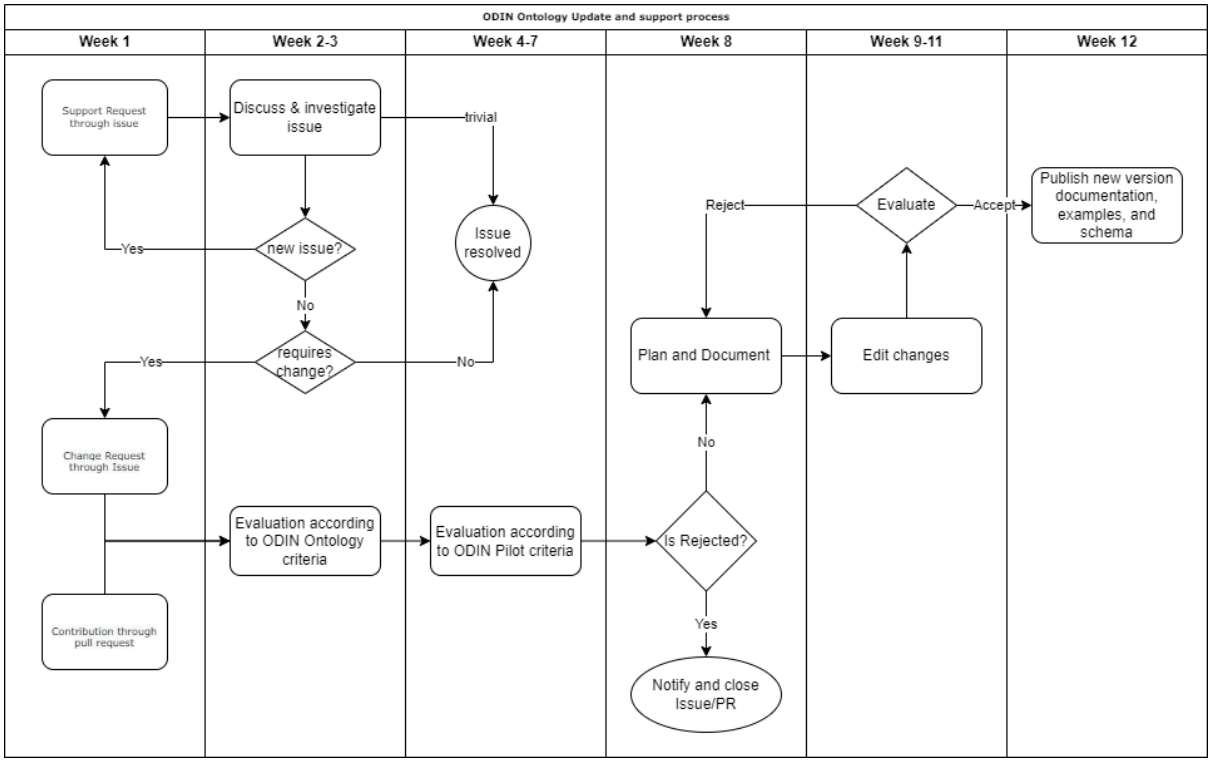

Figure 10.1. ODIN Ontology Update and support process.

If the ontology finds that it is useful for hospital management, not just within the ODIN project scope, then the best practice is to seek standardization effort. At this stage is difficult to commit on this endeavour, as most of the ontology is reusing models from already established ontologies (see Section 4), the material contribution is limited to the domain is restricted to mapping these ontologies for ODIN use cases, as well as internal models of the platform. The ODINEMDN ontology is formalising existing categories and may have more acceptance in the community as this initial effort is already performed.

11 CONCLUSIONS

This deliverable illustrates the second period of activity in Task 3.2, which started at M3, and ends on M36.

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 (D3.2), the same process has been iterated with the new requirements from the second period to develop ODIN ontology v2. After a study conducted on pilots' needs has identified a lack of ontologies covering KPIs, supply chain management, emergency situations and human-modelling.

KPIs allow the analysis and evaluation of performance factors, making them utile in Business Analytics. The use of KPIOWL[37] [47] or BIMERR-KPI[38] [48] ontologies can be employed to define the information for each indicator. These ontologies conceptualise important elements of indicators such as performance, results, measures, and target value.

For RUC C, disaster preparedness, further analysis of empathi[39] [49] and SEMA4A ontology [50] may be conducted in future versions. Empathi offers a conceptual framework for crisis management, through the identification and mapping of potential hazard situations. Similarly, SEMA4A facilitates the classification of alerts, enabling the modelling of evacuation routes.

As for logistics management, a search of ontologies must be done. As mentioned in section 3.1.1.3, the search should include the words “supply chain”, “logistics” followed by “semantic ontology”. In a preliminary review, the EAGLET ontology [51] was found. This ontology integrates three different perspectives: the physical flow, the chain of custody, and the chain of ownership, allowing a traceability of the item.

These will be analysed in more detail in the next version, and their inclusion in the platform will be discussed.

A new ontology has been created for the representation of medical devices, as no satisfying ontology could be found, based on the European Medical Devices Nomenclature (EMDN), recently released as a standard nomenclature from the European Commission. Another Ontology has been created the last period to explicitly model and support piloting operations in the ODIN project, modelling the 3 Reference Use Cases, and the defined KPIs, which are generic enough to be reused beyond the ODIN project.

Several 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 as well as the internal representation of ODIN platform have also been mapped in the ODIN ontology. The ontology has

[37] https://github.com/KhaosResearch/KPIOWL

[38] https://bimerr.iot.linkeddata.es/def/key-performance-indicator/

[39] https://shekarpour.github.io/empathi.io/

been applied to some pilots as case studies for static models, as well as with practical examples of the ontology in use.

Because this is the last version of the Ontology deliverable, an Ontology governance model has been proposed in order to keep the ontology up to date with the needs of ODIN project and platform, which may also be applied beyond the life of the project. The governance model stipulates how the ontology will be included in the DevOps for automatization of the validation, verification, and publication processes.

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

- Interoperability, defines 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.
- Replacement of the NCIT ontology by the Medical Subject Headings [51] (MeSH) ontology. NCIT is used in the ODIN platform for the definition of medical professionals, being an ontology for cardiological diseases, and as seen in Table 9-1 it does not have core value to ODIN use cases. MeSH, on the other hand, is a thesaurus of medical terms that also defines the different professions and has a broader scope.

12 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] «FIHR 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] S. H. W. V. H. R. P. M. M. S. D. W. M. L. G. B. B. C. J. R. K. E. H. Lauren B Becnel, «BRIDG: a domain information model for translational and clinical protocol-driven research,» *Journal of the American Medical Informatics Association,* vol. 24, n. 5, p. 882–890, 2017.

[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] «ICD9 for Health Ontology Mapper,» [Online]. Available: https://bioportal.bioontology.org/ontologies/HOM-ICD9.

[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. Gl¨ockner and A. Ludwig, «Lose ODP - an ontology design pattern for logistics services,» *Advances in Ontology Design and Patterns,* pp. 131-143, 2017.

[22] V. Mascardi, V. Cordì and P. Rosso, "A Comparison of Upper Ontologies," *Woa,* vol. 2007, pp. 55-64, 2007.

[23] 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, Annotatio and Analysis of Biomaterials Data,» *Advanced Functional Materials,* vol. 30, n. 16, 2020.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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 SCAIVew,» *CEUR Workshop Proceedings,* vol. 2087, n. 2020.

[29] 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.

[30] 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.

[31] «ITEMAS ontology for healthcare technology innovation,» *Health Research Policy and Systems,* vol. 17, n. 1, pp. 1-10, 2019.

[32] 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.

[33] H. R. a. M. I. Hussain, «A light-weight dynamic ontology for Internet of Things using machine learning technique,» *ICT Express,* 2020.

[34] 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 Amercian journal of Human Genetics,* vol. 83, n. 5, pp. 610-615, 2008.

[35] P. N. R. a. S. Mundlos, «The Human Phenotype Ontology,» *Clinical Genetics,* vol. 77, n. 6, pp. 525-534, 2010.

[36] 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.

[37] 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.

[38] «Organization ontology,» January 2014. [Online]. Available: https://www.w3.org/TR/vocab-org/.

[39] «Nci thesaurus,» August 2021. [Online]. Available: " https://ncithesaurus.nci.nih.gov/ncitbrowser/.

[40] «SNOMED 5 steps briefing,» [Online]. Available: https://www.snomed.org/snomed-ct/five-step-briefing.

[41] 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.

[42] «Protégé documentation,» [Online]. Available: http://protegeproject.github.io/protege/.

[43] «EMDN European Medical Device Nomenclature,» [Online]. Available: https://webgate.ec.europa.eu/dyna2/emdn/.

[44] «EUDAMED Device Nomenclature,» [Online]. Available: https://ec.europa.eu/tools/eudamed/#/screen/nomenclatures.

[45] «Cellfile plugin,» [Online]. Available: https://github.com/protegeproject/cellfie-plugin.

[46] «San Carlos Clinical Hospital,» [Online]. Available: https://www.comunidad.madrid/hospital/clinicosancarlos/nosotros/oferta-asistencial.

[47] M. d. M. Roldán-García, J. García-Nieto, A. Maté, J. Trujillo e J. F. Aldana-Montes, «Ontology-driven approach for KPI meta-modelling, selection and reasoning,» *International Journal of Information Management,* vol. 58, June 2021.

[48] M. Poveda Villalón, F. Radulovic e R. García Castro, «Developing ontologies for representing data about key performance indicators.,» 2014.

[49] M. Gaur, S. Shekarpour, A. Gyrard e A. Sheth, «Empathi: An Ontology for Emergency Managing and Planning About Hazard Crisis,» *IEEE 13th International Conference on Semantic Computing (ICSC),* pp. 396-403, 2019.

[50] A. Malizia, T. Onorati, P. M. Diaz, I. Aedo e F. Astorga-Paliza, «SEMA4A: An ontology for emergency notification systems accessibility,» *Expert Systems with Applications,* vol. 37, n. 4, pp. 3380-3391, 2010.

[51] G. L. Geerts e D. E. O'Leary, «A supply chain of things: The EAGLET ontology for highly visible supply chains,» *Decision Support Systems,* vol. 63, pp. 3-22, 2014.

[52] «Medical Subject Headings,» BioPortal, [Online]. Available: https://bioportal.bioontology.org/ontologies/MESH/?p=summary.

[53] 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.