FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF MASTER'S THESIS

Title:	Comparing Data Annotations using Deep and Shallow Semantics
Student:	Bc. Lukáš Bicek
Supervisor:	Ing. Robert Pergl, Ph.D.
Study Programme:	Informatics
Study Branch:	Web and Software Engineering
Department:	Department of Software Engineering
Validity:	Until the end of summer semester 2019/20

Instructions

The emergence of the FAIR data initiative brought attention to the need to provide proper semantics to digital resources in order to improve machine-based interoperability. However, in the current scenario, the majority of the so-called "ontologies" are dictionaries, vocabularies or taxonomies, providing shallow semantics not providing enough properties, relations and constraints to allow computer-based systems to infer their meaning and, therefore, present more intelligent behavior.

1. Acquaint yourself with FAIR data initiative.
2. Acquaint yourself with shallow and deep semantics ways of ontological modelling and perform their comparison.
3. Apply both ways for modelling meta-data of selected data sets provided by GO FAIR.
4. Compare the results and their possibilities with respect to requirements set for rich meta-data and formulate conclusions.

References

https://www.go-fair.org
https://ontouml.org
https://www.elixir-czech.cz/platforms/interoperability

Ing. Michal Valenta, Ph.D.
Head of Department

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague February 15, 2019

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Comparing Data Annotations using Deep and Shallow Semantics

Bc. Lukáš Bicek

Department of Software Engineering
Supervisor: Ing. Robert Pergl, Ph.D.

May 8, 2019

Acknowledgements

My deepest gratitude goes to my supervisor, Ing. Robert Pergl, Ph.D., because without his help, energy, support and insights I wouldn't be able to finish this thesis. I'd also like to thank Mark Thompson for taking on the role of domain expert with such drive and passion. His arguments, discussions and insights provided the needed push for the creation of the model. I'd also like to thank Luiz Olavo Bonino for his mentoring from the side of the domain supervisor; his insights proved to be the push in the right direction.

My thanks also go to my family, who provided me with the opportunity to study the school I wanted to and also for all the support and encouragement they provided me with during my studies and while writing this thesis.

Last but not least, I'd like to thank all my friends who supported me during my academic journey and provided me with distraction when it was needed and encouragement when the motivation was at its lowest and I was thinking about quitting.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 8, 2019 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Bicek, Lukáš. *Comparing Data Annotations using Deep and Shallow Semantics*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

Tato práce se zaměřuje na porovnání datových anotací za pomocí mělkých a hlubokých sémantik. Na začátku práce jsou položené teoretické podklady pro tuto práci. Praktická část práce se zaměřuje na porovnání těchto datových anotací za pomoci praktického případu. Diskuze je provedena nad transformací modelu v mělké sémantice (přesněji OWL) na model v hluboké sémantice, konkrétně v jazyku OntoUML.

Tato transformace podpořila hypotézy stanovené v práci, když donutila autora porozumět lépe doméně. Hotový model je více explicitní a pokládá základy pro více jednotnou interpretaci modelu.

Klíčová slova hluboké sémantiky, mělké sémantiky, ontologie, OntoUML, OWL, porovnání, sémantiky

Abstract

This thesis focuses on the comparison of data annotation using shallow and deep semantics. At the beginning of the thesis, all the necessary theoretical foundations are introduced. The practical comparison is made on an example of transforming a compared data model in shallow semantics, OWL to be specific, into data model using deep semantics in OntoUML language.

This transformation proved to be useful because it forced the author to understand the modelled domain better. The finished model proved to be a more explicit and more homogenous interpretation of the model was available.

Keywords comparison, deep semantics, ontologies, OntoUML, OWL, semantics, shallow semantics

Contents

List of Figures

List of Tables

Introduction

These days, the amount of data generated by research and other means is enormous, which drives the need for models for such data - an order is needed to conquer the chaos. These models are giving the possibility to other researches, for example focusing on tumor research, that may use the data in their research. Sadly all the models are opportunistic and domain-specific, without using any standardised ontologies, which makes the integration more difficult, time and effort demanding. The integration is then more human based, meaning that human interaction is required. Under human interaction is meant that the human has to review the data, consider the model, whether the model - if provided - is acceptable for the research and then write the transformation algorithms to integrate the data.

All this could be made easier when the data has rich metadata, that machines when provided with proper algorithms, interpret and integrate on their own, without almost any help from a human. Under human help, one can understand the fine-tuning the algorithms, setting the parameters for the transformation or scaling the incoming values.

Structure of the thesis

This thesis is structured into two main chapters: Theoretical foundation and a practical example of model transformation.

In the theoretical foundation, the reader is introduced to the Go-FAIR organisation and its concepts and aims. Afterwards, the thesis focuses on the Web Ontology Language (OWL in short) and all its subtopics. Following the OWL introduction, the focus is then laid on ontologies and ontological concepts. Second, to last in the theoretical foundation, the emphasis is laid on modelling, the logic, whether it is modal or description logic, on which both approaches are founded are introduced, followed by the Unified Foundational Ontology, an ontology introduced by Giancarlo Guizzardi with its modelling

language, OntoUML. Last but not least, the thesis compares the modelling tools, that can be used to create OntoUML models.

In the practical part, the thesis follows a process of transforming a model in the OWL language to a model in the OntoUML language. The thesis discusses all the issues that were encountered during the transformation. At the end of the practical chapter, a discussion of the feasibility of this approach takes place.

Aim of the thesis

This thesis aims to provide argumentation for the need of data annotation using deep annotations, OntoUML to be specific. It should discuss all the benefits of this approach but not forget the downsides of this approach.

Chapter 1

Theoretical foundation

In this chapter, the reader will go through all the needed knowledge for this thesis. It all starts with an introduction in the GO–FAIR initiative, which provides the demand for this thesis, then reader moves on to the introduction to the shallow semantics, explaining the OWL language with all necessary subtopics. Next the text looks into some basic ontologies and ontological concepts, and at last, we'll talk about OntoUML and modelling tools for OntoUML.

1.1 GO-FAIR

In this section, the thesis introduces the GO FAIR initiative, what it represents and what it aims to achieve. Entire section is based on [1] and its sub-pages.

GO, standing for Global Open, FAIR is an initiative trying to make the available data, mostly research data, that are fragmented and unlinked, FAIR, which means Findable, Accessible, Interoperable and therefore Reusable. The principles of GO FAIR initiative, which will be listed and explained later in section 1.1.1, are not just for handling the vast amount of generated research data, but also to provide the trusted environment of open data, where companies, researchers, citizens and innovators can create and re–use each other's data, for different purposes.

"There is a rapidly growing, world-wide consensus among scientists, science funders, and policy makers that the transition to truly data-driven Open Science can only be achieved when we collectively build a globally interoperable research infrastructure." [2]

1.1.1 The Principles

The principles were first stated in 2016 in [3] with the aim of establishing guidelines for findability, accessibility, interoperability, and reuse of digital assets. The emphasis is put on the machine readability (meaning the ma-

chines can find, access, interoperate and reuse data with none or low human interaction. This section will go through all four steps.

1.1.1.1 Findable

If we want to be able to reuse data, we first need to find them; this means that metadata and data itself should be easily findable for humans and machines as well. This machine-findability is essential for the ability to discover datasets and services, thus being a necessary component of the FAIR process. This section describes all the subprinciples of the Findable principle.

1.1.1.1.1 (Meta)data are assigned a globally unique and persistent identifier

This principle is, without any doubt, the most important one, because without any unique and persistent identifiers, it'll be nearly impossible to achieve other FAIR aspects. These globally unique and persistent IDs remove the ambiguity in the meaning of the published data by creating and assigning a unique and persistent identifier to every metadata element and every concept and measurement that was taken in the given dataset. Various data repositories resolve this by generating and assigning globally unique and persistent identifiers to deposited datasets. The unique identifiers aim to help people understand what the author of the dataset was pursuing and meaning by his/her research, and it also allows computers to interpret the data in a meaningful way and integrate them correctly. Identifiers are crucial for the interaction between human and a machine and also help others to cite correctly and reuse published data. This principle sets two conditions for the unique identifiers:

1. Global uniqueness obtained from a registry service using an algorithm to guarantee the uniqueness.

2. Persistency guaranteed by a registry that resolves the links to some degree.

Here are some examples of identifiers fulfilling these conditions:

- ORCID[1] – global and unique identifier for academic and research personnel

- DOI[2] – registration authority for the ISO standard 26324, used mainly to identify academic, professional, and government information (like articles, papers, datasets, etc.)

[1]For further information plese visit `https://orcid.org/`
[2]More information under `https://www.doi.org/`

1.1.1.1.2 Data are described with rich metadata

The second findable principle puts the emphasis the metadata that should be rich in describing information like context, quality and condition, or characteristics of the data. These rich metadata allow computers to automatically find, sort, prioritise and accomplish routine tasks, that are currently really time demanding for the researchers. The main idea behind this principle is for the research community to be able to find datasets and/or any other relevant information based just on the provided metadata without the need to know the identifier. Therefore, the compliance with this principle helps others to find any data and properly reuse or cite them. In layman terms, there's no such thing as useless metadata; anyone should be generous and provide any metadata that comes up in his or her mind.

For example, it includes both intrinsic metadata captured, while the data was created by machines as well as systematic metadata like protocols, algorithms etc. that were used. The measurement devices also should be included in the metadata.

1.1.1.1.3 Metadata clearly and explicitly include the identifier of the data they describe

Principle F3 is simple, straight forward and obvious, but thought this simplicity; it's critical for FAIR. The metadata file should be associated explicitly by stating the datasets global and unique identifier in the metadata file. As mentioned in 1.1.1.1.1, this is often done by dataset repositories upon uploading or creation.

1.1.1.1.4 (Meta)data are registered or indexed in a searchable resource

Having rich metadata and unique and persistent identifiers isn't what ensures the findability of given objects or data on the internet. Moreover, if any perfectly suitable data would fit someone's research just perfectly, without being listed in some sorts of registry, database or even being listed somewhere, it'd be entirely in vain. One example of ensuring that data will be found is by indexing it. The first three *Findability* principles provide the basis for this principle and indexing the data in some of the current repositories. In this section, the reader becomes familiar with all the subprinciples of the Accessible principle.

1.1.1.2 Accessible

When the data is found, the user also needs to be able to access them, possibly with some sort of authentification and authorisation.

1.1.1.2.1 (Meta)data are retrievable by their identifier using a standardised communications protocol

A link, that's how most of the users of the internet reach and retrieve their data. A link provides the interface to the low-level TCP protocol loading data to everyone's browser. This principle states that all data should be retrievable without having to use special tools or methods, which results in an explicit specification of who can access the data and how. In example, the most used protocols will be HTTP(s) and FTP.

1.1.1.2.1.1 The protocol is open, free, and universally implementable

To maximalize the reusability of the data the used protocol should be free of charge (no-cost) and opensource and therefore easy to implement on a global scale. Everyone, who owns a computer with internet access, should be able to access at least the metadata. And this will always impact the choice of the repository chosen for the data publication. Some examples are HTTP (more under [4]), FTP (more under [5]), SMTP (more under [6]) for the universal and for example Microsoft Exchange Server protcol for the proprietary one.

1.1.1.2.1.2 The protocol allows for an authentication and authorisation procedure, where necessary

This principle is vital but often misunderstood one element of the FAIR. Accessible doesn't mean open or free, but it rather implies that there should be precise and exact conditions under which the data is available. Therefore even a heavily protected and private data can be FAIR. Which means that ideally accessibility should be specified in such a way, that all machines can automatically understand the requirements and then execute them, or alert the user to take a closer look at them. Often this leads to the creation of a user account for the given repository, allowing for the identification of the owner or contributor of a given repository and specify the access rights. Hence, this will also be a huge aspect for the choice of repository.

1.1.1.2.2 Metadata are accessible, even when the data are no longer available

Over time, repositories tend to degrade or disappear because of the cost of maintenance of the online presence. As a result of this disappearing acts, users often waste time following invalid links leading nowhere. Metadata storage is much cheaper and more comfortable. This principle states that the persistence of metadata should be ensured even if the data is no longer available. This ties to the principle described in 1.1.1.1.4 referring to registration and indexing issues.

1.1.1.3 Interoperable

This principle aims at data being connected and being integrated with other data. Furthermore, the data needs to be interoperable with applications, al-

gorithms and workflows for analysis, storage and processing. This section provides fundamental insight into the subprinciples of the Interoperable principle

1.1.1.3.1 (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.

As well as humans should understand and be able to interpret the data, which means that no dead languages should not be used, machines and computers should be able to do the same without the need for any specialised and old ad-hoc algorithms, translators or mappings. Interoperability means in this point that computers should understand or at least have knowledge of other computer's data exchange formats or interface. This requires one of following:

1. Use of commonly used vocabularies and ontologies
2. A good, well-formed and well-defined data model

Some examples:

- RDF - will be described later in chapter 1.2.2
- OWL - will also be described later in chapter 1.2
- JSON LD [3] - more under [7]

1.1.1.3.2 (Meta)data use vocabularies that follow FAIR principles

The vocabulary that is used to describe the datasets needs to be well-documented and resolvable using persistent and unique identifiers. Both need to be also easily findable and accessible for everyone, who wants to use given dataset.

1.1.1.3.3 (Meta)data include qualified references to other (meta)data

Under qualified reference, we understand a cross-reference that explains its intent. *X is a regulator of Y* is, for example, a much more qualified reference than *X is associated with Y*. So the aim of this principle is, therefore, creating as many meaningful links as possible between the (meta-)data to make sure that the contextual knowledge is enriched and balanced in making a well-formed data model.

More precisely, it should be specified, whether a given dataset is being based on another dataset, or if it requires some other datasets to complete the data, or if any complementary information is stored in a different dataset. This means that all scientific links between datasets need to be described and all datasets need to be well cited.

[3] JavaScript Object Notation for Linked Data

1.1.1.3.4 Importance of interoperability

This principle is essential because if someone has already published datasets, you don't want to write special programs or parsers for each dataset, or configure all of the settings manually, without any machine assistance. This principle makes it easy for sharing the datasets without the need for having to customise all settings manually without any machine interpretation, etc.

1.1.1.4 Reusable

The goal of FAIR principles is the reusability of data and the optimisation of this reuse. For this to be possible, the data and meta–data needs to be well-described, so that anyone can use, replicate and combine it in different settings. In this section, the subprinciples of the final FAIR principle, Reusable principle, will be introduced.

1.1.1.4.1 Meta(data) are richly described with a plurality of accurate and relevant attributes

The findability and reusability will be much easier if the data contains many labels describing it. This principle is related to 1.1.1.1.2 but focuses more on the user, whether it's a human or a machine, to be able to decide whether the data is useful in given particular context. For this, the author of the dataset should provide not only meta-data making the discovery possible but also meta-data richly describing the context under which the dataset was created or generated. Under which one can understand experiment protocols, manufacturer and/or brand of the machine or sensor that created the data, etc.

The importance of this principle is that the author should not attempt to predict the user of the data, his/her identity or the needs or the reason for the usage of the data. It's important to provide as many metadata possible, to be generous, even including information that seems irrelevant at first sight. Some examples to take into consideration (but not all of them):

- Scope description
- Limitation or particularities about the data
- Specification of the date of generation, conditions, parameters and settings
- Raw or processed data
- Etc.

1.1.1.4.1.1 (Meta)data are released with a clear and accessible data usage license

The chapter 1.1.1.3, Interoperable, focuses on the technical interoperability. However, this principle focuses on the legal interoperability, meaning that every author of every dataset should think about the usage rights attached to a given dataset. The usage rights should be clearly described, any loose or ambiguities meaning could gravely limit the usage of a given dataset by anyone (e.g. organisations) struggling to comply with the provided license. Thus it's essential to provide a clear licencing status of any dataset, and it should be clear to anyone (humans and machines). Commonly used licences are MIT[4] and Creative Commons[5].

1.1.1.4.1.2 (Meta)data are associated with detailed provenance

It's important to include the description of the origin of the data, who to cite and/or how the author of the data wants to be acknowledged. It's recommended to include a description of the workflow how the data was created including, but not limited to:

- Who generated and/or collected the data
- Processing of the data
- Publishing occasions
- Etc.

Also, it's optimal, that this workflow is described in machine-readable format.

1.1.1.4.1.3 (Meta)data meet domain-relevant community standards

What makes reusability easier? Similarities like the same datatype, organisation in a standardised way, sustainable file-formats, etc. Also if there are any standards and/or best practices, they should be followed. An example of such standards is MIAME[6]. Data complying with the FAIR principles should meet at least these standards.

Different communities might have different standards, that might be less formal, but it's nevertheless essential to publish metadata in the manner that complies with the standards to increase the usability for the given community because this is the primary objective of FAIRness. All of this should be addressed in the metadata describing any dataset.

[4]More under `https://en.wikipedia.org/wiki/MIT_License`
[5]To learn more visit `https://en.wikipedia.org/wiki/Creative_Commons`
[6]More information under `https://www.ncbi.nlm.nih.gov/geo/info/MIAME.html`

1.1.1.4.2 Importance of reusability

The reusability is essential because it increases the value of already created data and datasets. Someone others could benefit from previously published data, using them for comparison, as an example or even as an experiment potentially gone wrong using the dataset as the point of origin for further improvements.

1.1.1.5 Conclusion

The principles talk about three entities: data, meta–data and infrastructure, data standing for any digital object and meta–data holding any information about this given object. The most important principles for this thesis are interoperability and reusability because these principles are profiting heavily from deep semantic.

1.1.2 Benefits

It is evident that in the current age, the sheer amount of data, that is being produced by research and other institutions, the need for some order in this chaos is needed. And on the other side of this production problem are the consumers, also researchers, schools and other institution, that might need this stream of data in some format so that the data can be integrated into their work.

And it is here, where the FAIR principles come into play. Currently, everyone needs to write their integration tools, or tweak the existing ones, to get the data in a shape that they can work with them. With data complying with the FAIR principles, there can theoretically be only one integration tool, that will tweak itself, after some basic configuration of the format, that the data output needs to be in, according to the rich metadata, that comes with the data. Integration would then be easy, and researchers could then focus on the core of their work, the research itself, and not waste effort and time on polishing the data for them to be integrated into the research.

1.1.3 Conclusion of GO FAIR

The GO FAIR initiative points out the importance of rich metadata. In all four of its principles, it states and points out, from different points of view, the importance of attaching rich metadata to published datasets. These principles ensure that all the datasets following the GO FAIR are easily findable, accessible, interoperable and reusable. This section described all of the principles and subprinciples and points, why each out of them contributes to the importance of rich metadata.

The key takeaway from this section is, that even when a piece of information seems irrelevant, it's better to include it in the metadata than leave it out because no one never knows, when it might come in handy. It provides

arguments for the importance of deep semantics over the shallow semantics because deep semantics provide more metadata by themselves without needing to include and interpret any ontology used by shallow semantics.

1.2 Web Ontology Language

Web Ontology Language, OWL, in short, stands for a family of languages for representing and authoring ontologies (more on ontologies in section 1.3). OWL languages are represented by formal semantics, build upon the World Wide Web Consortium's (W3C) XML standard for objects with the name Resource Description Framework (RDF, more on this topic in section 1.2.2). These standards are attractive to and used by academic, medical and also commercial subjects. OWL is built upon description logic, which provides the logical formalism for the ontologies and semantic web[7]. More about description logic will be in 1.4.1.1.

1.2.1 Semantics

Semantics are used in computer science to describe the definition of any semantic model, the relation between different models, etc. This subsection is based on [8].

There are several classes to formal semantics, in this thesis, three major will be listed:

Denotational semantics, also known as Mathematical is an approach to formalise meaning of programming languages using mathematical objects (or by constructing them), called denotation, hence the name of the method, to describe the meanings of expressions. [9]

Vastly speaking, this denotational semantics is focused on finding mathematical objects, often called domains, representing the program's doing. It is essential that denotational semantics should be compositional, built out of denotations of subphrases. [10]

Operational semantics is a category focusing on verification of specific program properties such as correctness, security or safety, based on the construction of proofs from logical statements rather than by associating mathematical meanings to the terms of the program (as in denotational semantics). There are two separate categories of operational semantics: structural (or small-step) and natural (or big-step).

Small-step (or structured, shortened SOS) operational semantics is a logical mean to describe operational semantics. The basic idea is the definition of a program's behaviour by specifying the behaviour of its

[7] more under `https://en.wikipedia.org/wiki/Semantic_Web`

parts, therefore providing a structural view on the operational semantics. [11, 12]

Big-step (also known as natural semantics) semantics can be viewed in general as a definition of relations to interpret a language's construct in the given domain, thus making it a popular choice, even if it's not suitable for some situations. [13]

Axiomatic semantics approach is based on mathematical logic to prove the correctness of any computer program, being closely related to Hoare logic (more on Hoare logic under [14]). The objective is to define the meaning of a command by using and describing the effect on assertions about the program state.

This is not the full enumeration of semantical approaches. There are more. These are the three most popular or most used ones. Semantics were first introduced in [13], by Robert W. Floyd.

1.2.2 Resource Description Framework

Resource Description Framework (often abbreviated as RDF), was first introduced in 1997. In 1999 it was adopted by W3C as a recommendation. Currently used specification is RDF 1.1, published in 2014. RDF was initially designed as a metadata data model, but since that, it became a general method for many other purposes like conceptual description, information modelling implemented in web resources or knowledge management applications. This section will be based on [15] and [16].

There are many similarities between the RDF data model and classical modelling approaches (e.g. Entity-Relationship (ER) diagrams). The foundation of RDF is making statements about resources in the form of *subject-predicate-object* (also known as triples, which will be explained in section 1.2.3). The subject is referencing the resource, predicate stands for the aspects of the resource and also represents the relation between subject and object.

The structure (often referred to as vocabulary) of RDF will be touched only briefly, because of the complexity and massiveness of the topic. The vocabulary structure consists of classes (e.g. `rdfs:Class`, `rdfs:Literal`), properties (e.g. `rdf:type`, `rdfs:label`) and other vocabularies like containers, collections. An example of usage can be found in listing 1 using the RDF/XML Syntax (more about this topic in [17]). Further information on this topic is available under [18]. RDF/XML is one of several serialisation formats. Other commonly used formats are (a few examples):

Turtle : compact format, human-friendly, more under [19], an example is given in listing 2

N-Triples : more about them in section 1.2.3

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:ex="http://example.org/stuff/1.0/">

  <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar"
             dc:title="RDF1.1 XML Syntax">
    <ex:editor>
      <rdf:Description ex:fullName="Dave Beckett">
        <ex:homePage rdf:resource="http://purl.org/net/dajobe/" />
      </rdf:Description>
    </ex:editor>
  </rdf:Description>

</rdf:RDF>
```

Listing 1: Example an RDF structure in RDF/XML notation from [17]

RDF/JSON : an alternative to XML, triples representation using JSON notation, more under [20], example in listing 3

```
@base <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rel: <http://www.perceive.net/schemas/relationship/> .

<#green-goblin>
    rel:enemyOf <#spiderman> ;
    a foaf:Person ;    # in the context of the Marvel universe
    foaf:name "Green Goblin" .

<#spiderman>
    rel:enemyOf <#green-goblin> ;
    a foaf:Person ;
    foaf:name "Spiderman" .
```

Listing 2: Example an RDF structure in Turtle notation from [19]

```
{
    "http://example.org/about" : {
        "http://purl.org/dc/terms/title" :
            [ {
                "value" : "Anna's Homepage",
                "type" : "literal",
                "lang" : "en"
            } ]
    }
}
```

Listing 3: Example an RDF structure in RDF/JSON notation from [20]

1.2.3 N-Triples

N-Triples is a concrete syntax for the RDF. It is a subset of Turtle (can be seen in the Listing 2) and is a line based and easy to parse. N-Triples were initially intended to be used for the description of test cases but evolved into popular exchange format for RDF data or documents.

The natural ability to parse comes from simple lines and no parsing directives. N-Triples are line-based, which means, each line represents a statement. Therefore no wrapping of lines for better legibility is allowed because it would end the statement. N-Triples triples are representing a subject, predicate and object as a sequence of RDF terms.

N-Triples triples are in the same time Turtle simple triples, but Turtle includes more and other representation of RDF. This means that whether N-Triples is parsed by Turtle parser or N-Triples parser, the same triples come out every time. An example of N-Triples representation can be found in Listing 4[8]. [21]

```
<http://one.example/subject1>
↪  <http://one.example/predicate1><http://one.example/object1>
↪  . # comments here
# or on a line by themselves
_:subject1 <http://an.example/predicate1> "object1" .
_:subject2 <http://an.example/predicate2> "object2" .
```

Listing 4: Example of an RDF structure in N-Triples notation from [21]

[8] Linebreaks were inserted into the listing using the `[breaklines=true]` parameter for the *Minted* package, because N-Triples standard does not allow linebreaks for better legibility, as mentioned in 1.2.3

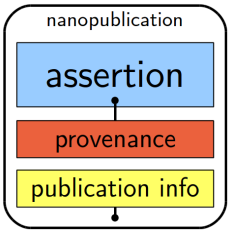


Figure 1.1: Nanopublication Scheme from [22]

1.2.4 Nanopublications

Nanopublications are a community-driven approach to represent structured data, with its provenance into a single, easily publishable and citable, entity. It is the smallest unit of publishable information: any assertion about anything, that can be uniquely identified and attached to its author.

Nanopublications are essential because they make it easy to spread individual data independently. That means that they don't have to be accompanied by the research article, but they can. Also, the ability for nanopublications to be cited and attributed, it makes it easy for researchers to make their data available in standard, machine-readable formats to drive the data accessibility and interoperability.

In Figure 1.1 is the visual description of a nanopublication, which consists of three fundamental elements (often called named graphs), each of which is represented by N-Triples (more about them in 1.2.3):

Assertion: is a proposition that can be tested, whether it is true or false. It is also the minimal unit of thought. An example can be in Listing 5.

Provenance: provides information about the assertion. Provenance graph must be linked to assertion graph via identifiers. The primary purpose of provenance element is, in layman terms, how the assertion came to be. It can include statements about how and when the assertion was generated, who generated it, where it was obtained from and any other similar information. An example of provenance is visible in Listing 6.

Publication Information: includes metadata about the nanopublication itself. It must include the nanopublication URI and should contain the attribution to the author and the timestamp (as seen in Listing 7).

As seen in the listings providing the examples of each element of nanopublication, nanopublications are represented with and may be queried using RDF or OWL (Semantic Web technologies). They also offer a mechanism that ensures the integrity and trustworthiness of the nanopublication. They also include a mechanism that ensures, that authors and institutions are acknowledged for the contribution to the global knowledge graph. The purpose of

```
:assertion {
 :BRCA1-gene :is-involved-in :breast-cancer .
 :BRCA1-gene :encodes :BRCA1-protein .
 :BRCA1-protein :is-expressed-in, :breast .
}
```

Listing 5: Example of an assertion element of nanopublication from [23]

```
:provenance {
    :assertion prov:generatedAtTime
    ↪  "2012-02-03T14:38:00Z"^^xsd:dateTime  .
    :assertion prov:wasDerivedFrom :experiment .
    :assertion prov:wasAttributedTo :experimentScientist .
}
```

Listing 6: Example of a provenance element of nanopublication from [23]

```
:pubInfo {
 :nanopubEx prov:wasAttributedTo :paul .
 :nanopubEx prov:generatedAtTime
 ↪  "2012-10-26T12:45:00Z"^^xsd:dateTime .
}
```

Listing 7: Example of a publication information element of nanopublication from [23]

nanopublications is to expose all the quantitative and qualitative data, such as hypotheses, claims, results (even negative, which often go unpublished) as individual and independent publication without the accompanying research article. [22, 23] In Listing 8 is a full example of such nanopublication, with all its elements.

1.2.5 Conclusion of Web Ontology Language

This section provided basic insight into the OWL, with its notations, and representations, foundation. Therms like RDF or N-Triples are introduced and explained.

The key takeaway from this section is, that OWL is the go-to language for representation of shallow ontologies.

```
@prefix : <http://www.example.org/pubs#> .
@prefix np:  <http://www.nanopub.org/nschema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .

:nanopubEx {
     :nanopubEx a np:Nanopublication .
     :nanopubEx np:hasAssertion :assertion .
     :nanopubEx np:hasProvenance :provenance .
     :nanopubEx np:hasPublicationInfo :pubInfo .
}

:assertion {
    :trastuzumab :is-indicated-for :breast-cancer .
    :assertion a np:Assertion .
}

:provenance {
    :assertion prov:generatedAtTime
    ↪  "2012-02-03T14:38:00Z"^^xsd:dateTime  .
    :assertion prov:wasDerivedFrom :experiment .
    :assertion prov:wasAttributedTo :experimentScientist .
    :provenance a np:Provenance .
}

:pubInfo {
 :nanopubEx prov:wasAttributedTo :paul .
 :nanopubEx prov:generatedAtTime
 ↪  "2012-10-26T12:45:00Z"^^xsd:dateTime .
 :pubInfo a np:PublicationInfo .
}
```

Listing 8: Example of a well-formed Nanopublication from [23]

1.3 Ontology and ontological concepts

In this section, the term *ontology* will be explained, in both the philosophical meaning and the meaning in information science. Furthermore, there will be listed some basic ontologies, and the difference between shallow and deep ontologies will be pointed out.

1.3.1 Ontology

The word ontology is a compound word consisting of two words originating in ancient Greece. Onto (from greek ὄντος, ontos) means "being" or "that which is" and logia (from greek λογία, logia) having the meaning "logical discourse". These two compounds give the word ontology the meaning of "study of being". The first occurrence in English is in [24] by Gideon Harvey.

He used it in its Latin form used by philosophers based on Latin roots, which themselves are based on the Greek. Gottfried Wilhelm Leibniz, one of the great philosophers of the 17th century, is the only of them, who has used the term ontology.

Philosophers from around the word and schools provide different answers to the fundamental questions of ontology, but they do have one approach in common – the division of subjects and predicates into groups that are called categories. This philosophical study is listed under the major branch of philosophy focusing on the fundamental nature of reality, the *metaphysics*. Ontology and its question often deal with the existence of entities and their grouping, with relation to hierarchies, similarities and differences. Classification of ontologies (based on philosophers) happens based on different criteria, such as abstraction or field of application. Some types of ontologies are:

1. Upper ontology (supporting the development of ontologies, so-called meta-ontology)
2. Domain ontology (concepts of given ontology are relevant only to a specific domain (e.g. computer language))

This listing is not complete; these categories are also used in the computer science meaning of ontology. Some of the known, prominent ontologists are (for example): Aristotle, Plato, Friedrich Nietzsche, René Descartes, above mentioned Gottfried Leibniz and many more. [25, 26]

The similarity between philosophy and computer science is the attempt to classify and represent all entities, ideas and events, with all the relations, properties according to a system of categories. Current work in ontology focuses more on establishing controlled vocabularies for narrow domains than building the foundation like principles, fixed essences or what is more ontologically important objects or processes. Tom Gruber, who is also a co-founder of Siri, presents an interesting way of thinking about ontology in his article that he published on the website of Knowledge systems, AI laboratory of Stanford University [27]:

"In the context of knowledge sharing, I use the term ontology to mean a specification of a conceptualization. That is, an ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents. This definition is consistent with the usage of ontology as set-of-concept-definitions, but more general. And it is certainly a different sense of the word than its use in philosophy."

And then he adds an example of his usage of ontology (in the same article):

"What is important is what an ontology is for. My colleagues and I have been designing ontologies for the purpose of enabling knowledge sharing and reuse. In that context, an ontology is a specification used for making ontological commitments. The formal definition of ontological commitment is given below. For pragmetic reasons, we choose to write an ontology as a set of definitions of formal vocabulary. Although this isn't the only way to specify a conceptualization, it has some nice properties for knowledge sharing among AI software (e.g., semantics independent of reader and context). Practically, an ontological commitment is an agreement to use a vocabulary (i.e., ask queries and make assertions) in a way that is consistent (but not complete) with respect to the theory specified by an ontology. We build agents that commit to ontologies. We design ontologies so we can share knowledge with and among these agents."

His article on the website is based on these two articles: [28] and [29]. Even thou Gruber is active in the AI field of computer science, his observation of ontology is valid for the broader use.

All the different ontologies share a mutual part, regardless of the expression of the ontology – the components. The following lines will provide some insight into these components:

Individuals: or instances are the fundamental "ground level" elements of an ontology. They represent concrete objects like people, animals, cars, buildings, planes, etc. Any ontology doesn't need to, strictly speaking, include any individuals. The general purpose of ontologies is to provide means to classify any individuals, even the ones, that are not part of an ontology.

Classes: often called type, sort or category. They often have two definitions: an extensional and an intensional. The extensional definition describes classes as abstract groups, sets or collections of objects. Intension describes groups as abstract objects, defined by values of aspects and constraints that are laid on the members of one group. An example of a class is person, vehicle or car. Whether a class can contain other classes, whether there's a superclass containing other classes is often put up for discussion. There are also restrictions put in place to avoid well-known paradoxes. From practical usage of any object-oriented programming language, it's known that there is the possibility of polymorphism, or that classes can contain other classes or a class can be part of another class like mammals are a subclass of animals.

Attributes: are often considered as descriptors of any objects, but they also can be independent. The kind of attribute and also the kind of the object defines the relationship between them. For example, the *Skoda Octavia* object has attributes **name** with value *Skoda Octavia*, **type** with value

sedan, and so on. The value of an attribute can be a **primitive** data type such as *integer* or *string*, but also a **complex** data type like the *type of engine* (can only have values from a list of subtypes) in the above-given example.

Relations: describe ways in which classes and individuals interact with each other and are related to each other. All relations can be directional, like for example an ontology containing concepts of *Barack Obama* and *George W. Bush* that might be related by a relation with the type *is the successor of* (Barack Obama is the successor of George W. Bush). Much of the power of ontologies comes from the ability to describe relations. A set of relations can describe the semantics of any domain.

Other components are *Functional terms*, which are complex structures that are build form certain relations, *Restrictions*, describing what needs to be true to be accepted as input. Furthermore, there are also *Rules* in the way of if-else statements describing logical inferences, *Events*, that change attributes or relations and *Axioms* that aids, with the help of assertions and rules, to comprise the overall theory that ontology describes in the given domain.

Another common thing between philosophy and information science is the classification of ontologies into different types. Similar to philosophy, in information science, there are also domain ontologies and upper ontologies.

Domain ontologies, as the name suggests, are domain specific ontologies. That means that they belong to a particular part of the world, e.g. politics or biology. This leads to every domain having a domain-specific definition of terms. Let's look at the term full-house. In the poker ontology, the term full-house represents a card combination, whereas, in the entertainment ontology, the term full-house represents a sold out venue. Different domain-specific ontologies mean that there are also many different authors of these ontologies, who have, each of them, unique way to represent the concepts. This leads to difficulties when merging ontologies even within the same project. The merge requires loads of effort, especially hand-tuning each entity, using some software merging tools and hand-tuning, which, especially hand-tuning, proves to be time demanding and expensive. This effect is partly mitigated when the ontologies are based on common upper ontology. There is ongoing research being done at the University of Edinburg (more about his research can be found in [30] for generalized techniques for merging of ontologies.

Upper ontologies, their main aim is to provide semantic interoperability across multiple domains. Upper-level ontologies provide structure for the above-mentioned domain ontologies. Upper ontologies provide a "top-down" approach to modelling, the basic structure of categories that don't need to be reinvented again and also the mentioned interoperability. The downside of them is to understand them, which can take a lot of effort and also for some they may be too abstract. There are several upper ontologies; this thesis will

go into detail, in its later chapters (1.4.2), about Unified Foundation Ontology (UFO), hence its the foundation for OntoUML, but now thesis will talk about other "well-known" ontologies.

First of them is the Basic Formal Ontology (BFO) [31, 32], developed by Berry Smith. BFO main purpose is to promote the interoperability among domain ontologies. It also incorporates the traditional three-dimensional enduring objects and also the four-dimensional (time-related and time-reflecting) objects into one framework. BFO is popular because it has more than 200 extension ontologies, that are built on it and shape it to its needs. Relevant to this thesis are the ontologies of the Open Biomedical Ontologies Foundry (OBO), because, in the practical part, the diagram provided for the transformation from OWL into OntoUML contains elements form OBO (more about OBO later in section 1.3.2).

Another ontology is General Formal Ontology (GFO) [33], developed by Heinrich Herre and the research group Onto-Med in Leipzig. GFO is a realistic ontology integrating processes and objects. Its specialities, among others, are its account of persistence and its time model, to which GFO introduces a special category, a persistent. Instances of the persistent category "remain identical" over time.

This section was based on following sources (among others mentioned in the section text): [25, 26, 27, 28, 29, 34, 35, 36, 37].

1.3.2 Ontological concepts

This subsection will list two concepts or vocabularies. First will be commonly used and well-known Friend-Of-A-Friend ontology (also known as foaf) and then OBO Foundry, which is used in the practical part of this thesis.

Friend of a Friend (FOAF) [38] is an ontology focusing on describing persons, their activities and their relations to other people and objects. It is also machine-readable. The FOAF project, which maintains the FOAF vocabulary, was started in 2000 by Libby Miller and Dan Brickley. FOAF was intended to be used as a description medium for people to describe themselves on the internet. It is a reasonably small vocabulary with only 13 classes and 62 properties, but the value is immense because over 300 other ontologies or vocabularies are using elements from FOAF.

Other ontology, discussed in this section, will be OBO and the OBO Foundry. OBO Foundry [39] is an initiative with the belief that the value of data is greatly enhanced when it exists in a form that allows it to be integrated with other data. At the homepage of the OBO Foundry, there is a listing of all ontologies that are used within the OBO ontology, and that are based on it. Among those ontologies, there is also BFO listed as an upper-level ontology, on which OBO ontologies are built. [40]

The model, that has been provided for the practical part of this thesis, includes following ontologies, that are members of the OBO Foundry:

Human Disease Ontology (DO): An ontology that has been deloped to be a standard for modelling human diseases. It aims to provide the biomedical community with a consistent, reusable and sustainable vocabulary of descriptions for human disease terms, phenotype characteristics and related medical vocabulary. [41, 42]

Genotype Ontology (GENO): Main focus of this ontology is to represent the levels of genetic variation specified in genotypes. The core of the ontology is built around a graph, that decomposes the genotype into smaller components, that provide all the necessary information about the genome. [43]

eagle-i Research Resource Ontology (ERO): An ontology upon which is a software of the same name built. The aim of this ontology is to model biomedical research resources such as instruments, Core Facilities, protocols, etc. Nowadays it is fully integrated into the VIVO-IFS Ontology. [44]

Information Artifact Ontology (IAO): New ontology based on the work done by the OBI-ontology. It focuses on information entities. [45]

Apart from ontologies that are part of the OBO Foundry, the model also includes the following ontologies:

Semanticscience Integrated Ontology (SIO): A simple and integrated ontology of types and relations for rich description of objects, processes and their attributes. [46]

Dublin Core (DC): An organisation/work-group that focuses on developing specifications on other topics of relevance to metadata, such as encoding syntaxes, usage guidelines, and metadata models. In this particular instance is the key point of the Dublin Core the DCMI Metadata Terms. [47]

1.3.3 Shallow and Deep ontologies

Before 2005, everything was called ontology; there was no need for the distinction between deep and shallow ontologies because all existing ontologies were shallow ontologies (according to today's naming conventions). The term deep ontologies had to be introduced after Giancarlo Guizzardi created the UFO and OntoUML.

The wording shallow and deep comes from programming languages - shallow and deep copies. Shallow copy is a reference to the original object, similar to this concept; shallow ontologies need vocabularies they can reference the modelled object to. Without them, the models are just graphs with some labels but zero.

Whereas a deep copy copies the complete contents of the given object and thus creating a new object, deep ontologies provide standalone structure, without any need for vocabularies.

1.3.3.1 Conclusion of Ontology and ontological concepts

The focus of the section was on the definition of the word Ontology from both perspectives, the philosophical perspective and the perspective of computer science. Next the types of ontologies are explained, and finally, the different ontology concepts are introduced, mostly the ones, that are present in the OWL model provided for the practical task of the thesis. Last but not least this section takes on the distinction between the shallow and deep ontologies.

1.4 Modelling

This section puts the emphasis on modelling basics for this thesis. We start with introduction to the UML modelling language and then will proceed to the more advanced OntoUML, which takes ontological concepts in account and implements them into UML, hence the name OntoUML.

1.4.1 Description logic vs. modal logic

In this section, the two foundations for each ontology will be introduced: description logic being the foundation for OWL and Modal logic for UFO.

1.4.1.1 Description logic

Description logic, often abbreviated as DL, is a family of knowledge representation languages. DLs are more expressive than propositional logic[9], but also less expressive than first-order logic (FOL)[10]. Main reasoning problems that DL deals with are in most of the cases decidable and there exist and have been designed decision procedures, that are efficient. Each DL features a balance between DL itself and expressivity and reasoning complexity by supporting different sets of mathematical constructors, an example being fuzzy descriptions logic.

Importance of DL lies in its application. Artificial intelligence needs DL to describe and reason the relevant concepts. It is also of key importance for shallow ontologies and Semantic Web[11] because of its foundation, the OWL (as defined in section 1.2) and its profile is based on DL. Most of the use of DLs

[9] More about propositional under `https://en.wikipedia.org/wiki/Propositional_calculus`

[10] Also known as Predicate Logic. More about this topic under `https://en.wikipedia.org/wiki/First-order_logic`

[11] More on Semantic Web under `https://en.wikipedia.org/wiki/Semantic_Web`

and OWL makes biomedical informatics, where it assists in the codification of the knowledge.

The terms that are modelled by DL are concepts, roles and individuals. The fundamental modelling concept is an axiom, a statement relating roles and/or concepts.

1.4.1.1.1 Nomeclature and naming convention

Although OWL is based on DL, it has slightly different naming of the concepts, as it can be seen in table 1.1.

Table 1.1: Synonyms table between OWL and DL

DL	**OWL**
individual	individual
concept	class
role	property

There are many varieties of DL. Therefore was put together a naming convention roughly describing allowed operators. The naming convention consists of the label for a logical language it is based on:

Attributive language, with the abbreviation $\mathcal{AL}$, being the base language allowing following concepts: atomic negations, concept intersections, universal restriction and limited existential quantification.

Frame-based description language, abbreviation $\mathcal{FL}$, allows the following: concept intersections, universal restriction, limited existential quantification and role restriction.

Existential language, with the abbreviation being, following the same trend as above, $\mathcal{EL}$, allowing concept intersection and existential restriction.

Followed by any of the elements listed in the table 1.2. There are, as could be expected some exceptions in the naming of some canonical DLs. $\mathcal{S}$ stands for $\mathcal{ALC}$ with transitive roles. $\mathcal{FL}$ has two sublanguages. One having disallowed role restrictions and being the same as $\mathcal{AL}$ without atomic negations with the label $\mathcal{FL}^-$. The other having disallowed limited existential qualifications and the label $\mathcal{FL}_0$. There is also an abbreviation for $\mathcal{ELRO}$ – $\mathcal{EL}^{++}$.

1.4.1.1.2 History

The term Description logic was given its name in the 1890s. Previously it was called terminological systems or concept languages. The history of DLs can be split into two main developments: Knowledge representation and the Semantic web.

Table 1.2: Elements used for labeling DLs

Label	**Label description**
$\mathcal{F}$	Functional properties
$\mathcal{E}$	Full existential qualification
$\mathcal{U}$	Union of concepts
$\mathcal{C}$	Complex concept negation
$\mathcal{H}$	Role hierarchy
$\mathcal{R}$	Role disjointness
$\mathcal{O}$	Nominals
$\mathcal{I}$	Inverse properties
$\mathcal{N}$	Cardinality restrictions
$\mathcal{Q}$	Qualified cardinality restrictions (available in OWL 2)

DL was introduced into knowledge representation (KR) to overcome the lack of logic-based semantics. Firs KR system that was based on DL was KL-ONE by Brachman and Schmolze [48], which started an entire family of systems. The 80s were prosperous for DL-based KR systems, which were developed using the structural subsumption algorithms. Some of these are KRYPTON (1983) or LOOM (1987). These systems were using DLs with limited expressiveness but relatively efficient (meaning polynomial time) reasoning. The 90s were turbulent times for the DL-based KR. The early 90s introduced a new paradigm, tableau-based algorithm, that allowed more expression for DLs. Systems based on these algorithms showed promising and acceptable reasoning performance on typical problems; however, the complexity was no longer polynomial in the worst case scenarios. The Mid to late 90s brought high expression DL-based systems with excellent performance, despite the high worst-case complexity.

DARPA Agent Markup Language (DAML) and Ontology Inference Language (OIL) can be viewed as syntactic variants of DL, on which the semantic web was built. OIL uses the $\mathcal{SHIQ}$ DL. The development of the combination of those two languages – DAML+OIL, formed the starting point of the W3C web ontology working group, which is responsible for the creation of OWL, which was issued in 2004 and replaced the DAML+OIL recommendation. OWL is based on the $\mathcal{SH}$ family of DL. After the finishing OWL, the recommendation for OWL was issued in 2009m which is based on the $\mathcal{SROIQ}^{(\mathcal{D})}$ DL language, where $(\mathcal{D})$ stands for the use of datatype properties.

1.4.1.1.3 Modelling and syntax using DL

In DL, there is a distinction drawn between a terminological box (so-called TBox) and assertional box (so-called ABox). TBox consists of sentences describing hierarchies of concepts, like relations between them. Meanwhile ABox consists of sentences about where the hierarchy individuals belong like rela-

tions between individuals and concepts. An example being: Every student is a person; belongs to the TBox, meanwhile, changing the sentence up a bit and making it more concrete as Lukas is a student; belongs to ABox. The distinction between those two is not significant, but it is vital for the modeller's perspective because it makes sense to distinguish between the contention of a term in the real world (TBox) and their particular manifestation (ABox). Also, important is the distinction for the DL reasoners may process both boxes separately. For one, certain inference problems might occur that are tied to one but not the other one (like classification and instance checking – the first belonging to the TBox and the latter being ABox.

Syntax, similar to FOL, defines a collection of symbols, that are legal expressions and semantics determine the meaning. Some of the constructors that are being used in DL are similar to the FOL, like insertions, conjunctions, unios and disjunctions. In the table 1.3, the conventional notation of DL is listed, X and Y being concepts, x and y being individuals and R being a role.

Table 1.3: Conventional notation for DL

Symbol	**Description**	**Example**	**Read**
$\bot$	empty concept	$\bot$	bottom
$\top$	every individual is an instance	$\top$	top
$\sqcup$	union/disjunction of concepts	$X \sqcup Y$	X or Y
$\sqcap$	intersection/conjunction of concepts	$X \sqcap Y$	X and Y
$\sqsubseteq$	Concept inclusion	$X \sqsubseteq Y$	all X are Y
$\equiv$	Concept equivalence	$X \equiv Y$	X is equivalent to Y
$\exists$	existential restriction	$\exists R.X$	a R-successor exists in X
$\forall$	universal restriction	$\forall R.X$	all R-successors are in X
$\neg$	negation/complement of concepts	$\neg X$	not X
$\doteq$	Concept definition	$X \doteq Y$	X is defined to be equal to Y
:	Role assertion	$(x, y) : R$	x is R-related to y
:	Concept assertion	$x : X$	x is a X

Interpreting concepts as different sets of individuals and roles as different pairs of individuals define the semantics of DL. This will be shown at an

example of $\mathcal{ALC}$. It follows the definitions set in [49]. $\mathcal{I} = \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}$, where $\Delta^{\mathcal{I}}$ is called *domain* and $\cdot^{\mathcal{I}}$ is an *interpretation function* mapping: every individual x to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, then every concept is mapped to a subset of $\Delta^{\mathcal{I}}$ and finally every role name maps to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ is called *terminological interpretation* with the interpretation of the notations being:

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
- $\bot^{\mathcal{I}} = \emptyset$
- $(X \sqcup Y)^{\mathcal{I}} = X^{\mathcal{I}} \cup Y^{\mathcal{I}}$ (union equals disjunction)
- $(X \sqcap Y)^{\mathcal{I}} = X^{\mathcal{I}} \cap Y^{\mathcal{I}}$ (intersection equals conjunction)
- $(\neg X)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus X^{\mathcal{I}}$ (complement equals negation)
- $(\forall R.X)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \text{for every } y, (x, y) \in R^{\mathcal{I}} \text{ implies } y \in X^{\mathcal{I}}\}$
- $(\exists R.X)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \text{there exists } y, (x, y) \in R^{\mathcal{I}} \text{ and } y \in X^{\mathcal{I}}\}$

This section is based on following sources: [48, 49, 50, 51, 52, 53].

1.4.1.2 Modal logic

Modal logic had its initial development in the 1960s, extending the existing propositional and predicate logic by introducing the operators to express modality. A statement needs to be qualified by a word that expresses modality – a modal. An example, let Robert is supportive be a statement, needs to be qualified by making it Robert is always supportive, where always has the function of the modal. Modalities can include, but not limited to: possibility, necessity, impossibility, time modalities (as seen in the example above), obligations and permissions, knowledge and belief.

Modal logic (or ML) uses two basic unary operators. First of them is a diamond ($\Diamond$), representing possibility and the second one being a square ($\Box$), representing necessary. Let P be a statement like *a thesis will be submitted.* Then $\Diamond P$ means: *It is possible that a thesis will be submitted*, and $\Box P$ has the meaning: *It is necessary that a thesis will be submitted.* As in any other logic, each term can be expressed by the negation of the other. $\Diamond P \leftrightarrow \neg\Box\neg P$; with the meaning: *It is possible that a thesis will be submitted, if and only if it is not necessary that a thesis will not be submitted* and vice versa: $\Box P \leftrightarrow \neg\Diamond\neg P$ with the meaning: *It is necessary that a thesis will be submitted, if and only if it is not possible that a thesis will not be submitted.*

1.4.1.2.1 Development of modal logic

Aristotle did the first development of modal logic or syllogistic in his work Prior Analytics. Afterwards, his successor Theophrastus tried to improve his work. Further Greek logicians worked on the modal logic. Notably, Diodorus

Cronus and Chrysippus formed modal systems with accounting for possibility and necessity. Furthermore, Avicenna developed the first modal logic system.

As the founder of the modern modal logic is considered Clarence Irving Lewis with his Ph.D. thesis submitted in 1910 on Harvard [54]. His work culminated in a collaboration with Langford with the book [55]. Another developer of ML is Ruth Barcan, who worked on axiomatic systems. It seems as Harward university was one of the critical places for ML because of another researcher Saul Kripke[12], at the time of his publication being only 19 and an undergraduate, who created the Kripke-semantics. – a model theory for non-classical logics.

The mathematical structure of modal logics, featuring unary operations (caller modal algebra) are Boolean algebras. The first breakthrough was made by McKinsey in his 1941 work [56], decidability of S4 (described later in this section). Also important was the survey by Goldblatt connecting formal ML and associated mathematics, in his work [57].

1.4.1.2.2 Semantics of Modal Logics

This section will be based on the work of Fitting and Mendelsohn – [58].

The first definition that needs to be done is a *frame*, a non-empty set G, members of this set being called possible worlds. Next definition is a binary relation R, that holds (or not) between the members of G - this relation is called accessibility relation. This builds a pair - $\langle G, R\rangle$ - that is almost describing the model. An example of interpretation can be xRy with the meaning *world x is accessible from world y.*

To complete the description of the model, the true-values needs to be defined, for all worlds of G. Therefore the definition of a relation v, between all possible worlds and positive literals, is required. Then $v(x, P)$, where x being a world and P a literal, means that P is true at x. Hence is the model described as an ordered triple: $\langle G, R, v\rangle$. With this definition of model, it is now possible to recursively define the truth formula at a world in a model (with P and Q being literals and x, y being particular worlds):

- if $v(x, P)$ then $\models P$
- $w \models \neg P$ if and only if[13] $w \not\models P$
- $x \models (P \wedge Q)$ iff $x \models P$ and $x \models Q$
- $x \models \Box P$ iff $\forall y \in G$, if xRy then $y \models P$
- $x \models \Diamond P$ iff $\exists y \in G$ it holds that xRy and $y \models P$
- $x \models \Diamond P$ iff $\exists y \in G$ it holds that xRy and $y \models P$

[12] The surname might sound familiar. He was an inspiration for naming a character of THe Big Bang Theory: Barry Kripke

[13] abbreviated with iff

- $\models P$ iff $w*$ [14] $\models P$

According to the semantics, that ware specified above, the necessity of truth concerning a possible world x means, that it must also be true in all other worlds accessible to x and as the reader could think a possibility needs to be possible in some of the worlds accessible to x. Hence the possibility is depending on the accessibility relation R allowing the expression of the nature of possibility. A clear example of this being a statement: The laws of nature prohibit humans to travel faster than the speed of light, but if other circumstances were present, it could be possible for humans to do so. Which can be translated using the accessibility relation to In all worlds, that are accessible to our world, humans are not travelling faster than the speed of light, but it might be a possibility on any other world that is accessible from the worlds accessible to our world, but not directly to our world.

Accessibility relations enables the distinction of different systems of ML. Accessibility relation can be:

reflexive iff $\forall x \in G : xRx$

symmetric iff $\forall x, y \in G : xRy \rightarrow yRx$

transitive iff $\forall x, y, z \in G : (xRy \wedge yRz) \rightarrow xRz$

serial iff $\forall x \in G, \exists y \in G : xRy$

Euclidean iff $\forall x, y, z \in G : (xRy \wedge xRz) \rightarrow yRz$ and also zRy

From these frame conditions are stemming following logic:

- **K** := no conditions are required
- **D** := R has to be serial
- **T** := R has to be reflexive
- **B** := R has to be reflexive and symmetric
- **S4** := R has to be reflexive and transitive
- **S4** := R has to be reflexive and Euclidean

All of the above specifies logics can also be defined using axioms. First formalisations of ML were, in fact, axiomatic (like for example C. I. Lewis has done). There are following axioms specified (first two being the most known, the other being important for):

- **N - Necessitation Rule**: if p is a trhorem in any system invoking **N**, then $\Box p$ is likewise a theorem

[14] stands for *the actual world*

- **K - Distribution axiom**: $\Box(p \to q) \to (\Box p \to \Box q)$
- **T - Reflexivity axiom**: $\Box p \to p$
- **4**: $\Box p \to \Box\Box p$
- **B**: $p \to \Box\Diamond p$
- **D**: $\Box p \to \Diamond p$
- **5**: $\Diamond p \to \Box\Diamond p$

Which are foundation for following systems (in bold are the axioms and in italics are systems):

- $K := \mathbf{K} + \mathbf{N}$ - Named after Saul Kripke and being the weakest normal modal logic
- $T := K + \mathbf{T}$
- $S4 := T + \mathbf{4}$
- $S5 := T + \mathbf{5}$
- $D := K + \mathbf{D}$

There are logic systems that are built upon ML and are extending the classical ML with some of their inventions or change the meaning of $\Box$ and $\Diamond$. In the listing bellow are some of these logic systems

Alethic logic introduces two other propositions, contingency (something is possible but not necessarily true) and impossibility (something is false, and it has to be necessarily false). That being said, the notion of possibility or necessity is basic, and the other notions are defined according to the terms of De Morgan duality[15].

Epistemic logic originates in the Greek word *episteme* – knowledge, and transforms the symbols in the following matter: $\Box$ has now the meaning: *x knows that something happend* and $\Diamond$ meaning has been changed to *for all that x knows, it may be true that something happend.*

Temporal logic: introduces the notion of time, so some statements, like for example mathematical formulas are true at all times, some statements, like for example statements about human feelings are true only sometimes. There is also a need to introduce two pairs of operators (one for the past and one for the future) and therefore also the meaning of $\Box$ and $\Diamond$ also changes. $\Box$ means in this logic at every time, and $\Diamond$ means at some time. With this being said, there are now several possibilities to form temporal modal logics.

[15] more under `https://en.wikipedia.org/wiki/De_Morgan_duality`

Section on modal logic was based on following sources: [54, 55, 56, 57, 59, 58, 60, 61, 62]

1.4.1.3 Comparing both approaches

Both logic systems are developed independently, while DL being related to ML. Many of the Dls are syntactic variants of ML. The significant change is in the naming conventions like object corresponds with the possible world, concepts to modal propositions and role-bounded qualifiers to modal operators, where role works as accessibility function. All the name-changes can be viewed in table 1.4 from [63]. This topic is also described in [49].

1.4.2 Unified Foundational Ontology

Unified Foundational Ontology, shortly UFO, is the brainchild of Giancarlo Guizzardi, who laid the foundation for UFO in his PhD thesis. [64] His idea and the idea behind UFO was to create a strong foundation for conceptual modelling based on strong philosophical principles and at the same time on human common sense. The foundation logic for UFO is the modal logic, described in section 1.4.1.2.

There are several branches of UFO:

UFO-A is the only type, where the development is no longer ongoing, and there are enough practical examples of usage to prove the concepts of this branch. It is also the foundation for OntoUML. The ideas of this branch will be discussed further in this thesis and UFO-A will be the primary consideration for the thesis.

UFO-B and its main field of action is time modelling. The two main entities are events and time structures, representing time intervals. As it was discovered later in during the work on the practical part, it would be beneficial for the transformation, when UFO-B was already part of OntoUML.

UFO-C is focusing on the modelling of commitments, with Commitment as the central element of the package.

UFO-L domain specific ontology, focusing on legal core.

UFO-S domain specific ontology, which is focusing on the modelling of services.

In the appendix, section B the models of UFO-A to UFO-C will be presented. Figure B.1 presents the structure of UFO-A (all elements in yellow), figure B.2 is presenting the extension of UFO-B to the model of UFO-A (UFO-B elements are in green colour). And finally, in figure B.3, the extension through the UFO-C is presented in pink/red colour.

Table 1.4: Corresponding notions for description logics and modal logics from [63]

Description logic	**Modal logic**
$\mathcal{ALC}$	Multi-modal logic K_m
Concept formula	Modal formula
Concept definition	Modal formula of the kind *concept-name* $\Leftrightarrow$ *formula*
Concept name	Predicate symbol
Concept	Extension of a predicate symbol
Role name	Parameter of a parameterized modal operator
Role	Accessibility relation
Role term	Complex parameter of a modal operator
Role fillers	Set of accessible worlds
T-Box	Set of concept definitions
A-Box entry	Name of a world
A-Box	Description of a partial Kripke structure
Domain	Set of worlds
Object	World
Consistency of a concept formula	Satisfiability of a modal formula
Subsumption between concept formula	Entailment between modal formula
Existential quantifier $\exists r.\varphi$	Diamond operator $\Diamond\varphi$
Universal quantifier $\forall r.\varphi$	Box operator $\Box\varphi$
Number restriction $\|r\| \geq n$	Simple graded modal operator restriction on the number of accessible worlds
Qualified number restriction *atmost r n.*φ	Graded modal operator $\langle r\rangle_{n}\varphi$
Arithmetic constraint for the role fillers	(Not well investigated)

The main idea behind the UFO is the concept of universals and individuals. Where the universals are independent durable musters of entities (for example person, or dog) and individuals are instantiating these universals, ergo providing them with identity (following the case before, instances of persons could be Lukas Bicek, Robert Pergl, Giancarlo Guizzardi and instances of dogs could be Chubby, Nero, Rex and so on). Any known reality can be described using universals, individuals and the relations between those. The relations between universals and individuals can be visible in figure B.1.

1.4.2.1 UFO-A

In this thesis, the UFO-A and its structural aspects will be used. The UFO-A metamodel is divided into three main areas: Substantials, Aspects and Relations.

1.4.2.1.1 Substantials

The firs area introduced will be the Substantials. But before the reader can do that, he has to be made familiar with the meta-attributes that are playing an important role in this section. There are four main attributes: Identity (I), Own identity (O), Rigidity (R) and Dependency (D)

Identity: An object can eighter provide (and have its own) identity or have an identity. Thus the two different labels. Universals with identity are called sortals, and in literature, it is said that sortals provide identity, which means that an individual is uniquely and immutably identifiable to a sortal, but a sortal is not enough to an individual.

Rigidity: There are three types of rigidity.

First of them is *Rigidity* itself with the definition

$$R^{+}(T) \stackrel{\text{def}}{=} \Box(\forall x : T(x) \rightarrow \Box(T(x)))$$

with the meaning in layman terms: If an individual x instantiates T in a world, then it has to instantiate T in every other world. An example of a rigid Universum is a person.

The second type of rigidity is *Anti-Rigidity* with the definition

$$R^{-}(T) \stackrel{\text{def}}{=} \Box(\forall x : T(x) \rightarrow \Diamond(\neg T(x)))$$

that translates into: *If an individual x instantiates T in a world, then it does not necessarily have to instantiate T in every other world.* An example being a teacher role. A person can be a teacher in one world (like the author is the water safety instructor), but in another world, e.g. school the person does not posses the role.

Next type of rigidity is *Non-Rigidity* - a logical negation of rigidity. The definition of non-rigidity is

$$R^{\neg}(T) = \Diamond(\exists x : T(x) \wedge \Diamond\neg T(x))$$

The last type of rigidity is *Semi-Rigidity*. The definition of semi-rigidity is

$$R^{\sim}(T) = R^{-}(T) \wedge \neg R^{\sim}(T)$$

Dependency: has a really simple definition:

$$D^{+}(T, P, R) = \Box(\forall x : T(x) \rightarrow \exists y : P(y) \wedge R(x, y))$$

There are two types of substantials: Sortals and Non-Sortals. Sortals are then furthermore divided into Rigid and Anti-Rigid sortals. Rigid sortals are *Kind*, *Quantity* (representing the nominalisation of substance quantity - such as water or sand) and *Collective* (general representation of a collective - forest, deck of cards) (the so-called substance sortals) and *Subkind*. The details of all the mentioned types will be provided in the OntoUML section of this thesis (section 1.4.3). The Anti-Rigid sortals are *Roles* - entity has a role in given context, such as teacher or student and *Phases* - a complex sectioning of entity, the generalisation relation has to be complete and disjoint (e.g. a human is the phase living or in the phase death). There are three Non-Sortals: *Category* (represents values that have kinds in common - Muscles, Bones) (rigid non-sortal), *Rolemixin* (representing values that roles have in common - Customer (corporate customer and private customer) (anti-rigid non-sortal) and *Mixin* - a semi-rigid crossbreed between *Category* and *Rolemixin*.

1.4.2.1.2 Aspects

Aspects, in the metamodel, called moments. Every aspect is existentially dependent on other universals. Existential dependency is defined as

$$ed(x, y) \stackrel{\text{def}}{=} \Box(\epsilon(x) \rightarrow \epsilon(y))$$

$\epsilon(x)$ means *exist in domain* and the definition is saying that: *Individual x is existentially dependent on another individual y, if and only if y must exist whenever x exists*. The three aspects, defined by UFO-A, are *Mode* - a representation of thoughts and beliefs, *Quality* - representing measurable qualities like color or weight and *Relator* - a truth-maker for relations, like a handshake asfer a sighned contract.

1.4.2.1.3 Relations

Relations are entities connecting universals. There are three main types of relations: *Generalisation*, *Material* and *Formal relations*, and *Whole-Part* relations:

Formal and material relations: Formal and material relations represent the associations. The difference between these two is that material relation changes the history, where formal doesn't. *Material* relation has the restriction in the form of its truth-maker (relator) and the cardinality restriction are based on the mediation relation. Formal relations are then further split into *Characterizations* - a relation between mode or quality and an entity that it specifies, *Mediations* - used in decomposition of the material relation, it always has to be connected to a Relator. *Derivation* - a connection between relator and material relation for which it is the truth-maker and *Domain Formal Relation* (abbreviated as *Formal*) - a relation between multiple entities and has the least restrictions.

Generalisation relations: This relation specifies the relation a type and his subtype, like between a Mammal (supertype) and Human, Dog, Cat (as subtypes). Each subtype is a part of a generalisation set of the supertype. There are two attributes that a generalisation relation can have. The relation can be discrete, annotated as {*disjoint*}, means that it has to be eighter one of the members of the generalisation set. The second attribute is the completeness, denoted as {*complete*}, with the meaning that no other subtypes, apart from the ones that are listed, are allowed. There are all possible combinations permitted. When the generalisation isn't denoted, it has the meaning as non-complete (other possibilities are allowed) and non-disjoint (members can overlap).

Whole-Part relations: UFO-A focuses on these types of relations, using the existing notation in UML, but changing and making their meaning more concrete. There are a total of four types of this relation: *componentOf* - relation between a collective and its members, where the members have each different value (an ALU[16] is a part of a CPU[17]), *memberOf* - relation between a collective and its members,but with the difference that all the members are equal (a tree is a part of a forest), *subCollectionOf* - relation between collections (a collection can have subcollections) and *subQuantityOf* - similar to subCollectionOf, a Quantity can have subquantities. Each type is distinguished by the universals that it connects to. A part can be essential when the whole depends on the part, but also part can be dependent on the whole, then the part is inseparable. The important definitions for relations are:

Generic dependent: *individual y is generically dependent on Universum U iff y exists it is necessary that U exists.*

$$GD(y,U) \stackrel{\text{def}}{=} (\epsilon(y) \rightarrow \exists x : U(x) \wedge \epsilon(x))$$

[16] Arithmetic and Logic Unit
[17] Central Process Unit

Essential parthood: An example of essential parthood is the relation between brain and living human. One cannot change the brain of a human without killing him and thus destroying the whole.

$$EP(x,y) \stackrel{\text{def}}{=} (\epsilon(y) \rightarrow (x \lhd y)$$

$(x \lhd y)$ means x is part of y, and the definition has the meaning: *An individual x is an essential part of another individual y iff y is essentially dependent on x and x is a necessary part of y.*

Mandatory part: An example, following the human body, is the relation between human and his heart. Any human has to have a heart, but it does not have to be his own heart (one can switch heart instances - transplant hearts). Definition

$$MP(U,y) \stackrel{\text{def}}{=} (\epsilon(y) \rightarrow \exists x : U(x) \wedge (x \lhd y))$$

saying that *Individual x is a mandatory part of another individual y, iff y is generically dependent on Universum U, and y is a necessary part of U.*

Mandatory whole: *An individual y is a mandatory whole for the individual x, iff x is generically dependent on Universum U, instantiating y, and x is necessary part of an individual instantiating U.*

$$MW(U,x) \stackrel{\text{def}}{=} (\epsilon(x) \rightarrow (\exists y : U(y) \wedge (x \lhd y)))$$

Inseparable parthood: *Individual x is an inseparable part of another individual y, iff x existentially dependent on y and x is a necessary part of y.*

$$IP(x,y) \stackrel{\text{def}}{=} (\epsilon(x) \rightarrow (x \lhd y))$$

The section about UFO was mainly based on the thesis by G. Guizzardi [64]. Further resources were: a masrtes thesis form University of Economics in Prague (in czech) [65], a paper by Guizzardi and Wagner [66] a slides from Faculty of Electrical Engineering, Czech Technical University in Prague [67] and the habitation thesis of R. Pergl [68] and his slides for a course at Faculty of Information Technology, Czech Technical University in Prague [69](in english) and [70](in czech) and course by the NEMO group [71], which G. Guizzardi is a part of. (Slides from this course are available on the attached CD, see C for the exact location).

1.4.3 OntoUML

In this section, the OntoUML elements, that have been mentioned in the UFO section (section 1.4.2.1) and are used in the thesis will be mentioned. The full

listing of the elements with all their meta attributes and available relations are under [72]. The entire OntoUML metamodel can be found in figure B.4.

OntoUML, as well as its foundations - UFO, is based on the work of Giancarlo Guizzardi, first presented in his PhD thesis [64].

1.4.3.1 Origin and comparison to UML

OntoUML is an ontology modelling language based on UFO (see section 1.4.2). It is built upon the UML class diagram, by introducing new stereotypes, thus still being technically UML profile. By extending the UML class diagram metamodel only by introducing new stereotypes, makes the OntoUML metamodel still compliant to the UML metamodel. The naming of these two languages can often lead to the meaning that they are both based on the same foundation. OntoUML is now accepted and used among academic workes, public and private sectors as well. The downside of this modelling language is the lack of practical case studies because most of the projects are part of private sectors, subject to an NDA [18] or contains sensitive information, that cannot be published.

1.4.3.2 OntoUML notation

As stated before, OntoUML is a profile of UML. It is using the class diagram as its foundation, adding the stereotype. A Stereotype is specified using `<<stereotype>>` notation added to the "class" or relation. By class is meant the universal, which are represented as UML classes. Everything else remains standard to UML, such as element names, cardinality by relations, attributes and methods of classes. An example can be seen in figure 1.2, where subfigure 1.2a portraits an example of universal or aspect, subfigure 1.2b provides a muster for relations.

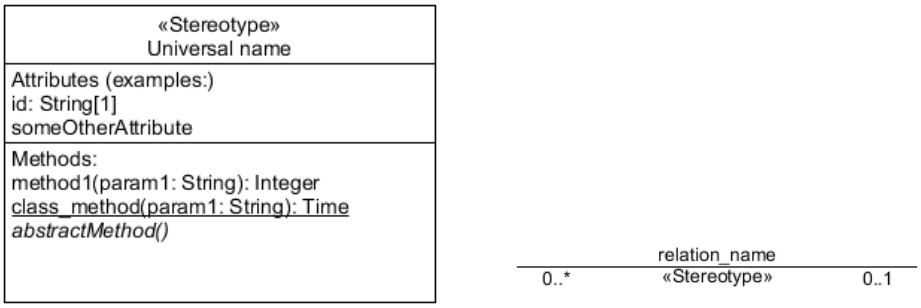
The entities used in this thesis are Kind, Role, Relator, NominalQuality, Mode and relations used are Characterisation, Mediation, Material, Generalisation and simple association. In the table 1.5 the abbreviation for the meta-attributes or meta-properties and their meaning is defined.

Kind, with the Meta-Properties +O, +I,+R and -D, represents a substance sortal, that in real-world natural Kinds like humans, animals and so on, but also artefacts like chairs, cars and so on.

Role (-O, +I, -R, +D) represent phased-sortal, representing a role of given sortal that it is related to. For instance, the role student is played by an instance of the kind Person.

NominaQuality is a specialised type of *Quality*, which is an Aspect, that references to an individual. It specifies the Sortal that it is connected to (e.g. a color of an object).

[18] Non-Disclosure Agreement



(a) Universal example, same applies to aspects

(b) Relation example

Figure 1.2: Examples of OntoUML notation

Table 1.5: Meta-Properties abbreviations from [72]

Prefix	**Meta-Properties (MP)**
+O	Provides identity (Own identity)
-O	Does not provides identity
+I	Identity
-I	No identity
+R	Rigid
-R	Anti-Rigid
R	Semi-Rigid
+D	Dependent
-D	Independent

Relator is also an Aspect, and it is used as a truth-maker on `<<Material>>` relations, providing a statement that verifies the relation. An example being Marriage relator, that verifies the material relation between two humans. It has to be connected to the relation that it is the truth-maker for via a Derivation relation.

Mode is an Aspect, depending on exactly one entity. An example of such can be thought, skills, intentions and so on.

Characterisation relation is specifying a connection between a sortal and a quality. *Mediation* is used in the decomposition of *material* relation, which connects two sortals, into a relation using two mediation relations and a relator. The meaning of the *generalisation* relation is used similiar in programming and so on. It is a relation between a super type and its subtypes.

This section about OntoUML was based on following resources: [72, 68, 69, 70, 64, 73].

1.4.4 Conclusion of Modelling

Modelling section provides the core knowledge for this thesis. It introduces both logic models - description logic for OWL and modal logic for OntoUML. Afterwards, the section presents the Unified Foundational Ontology and introduces its concepts. As next OntoUML, the modelling language for UFO is presented.

The key takeaway is that OntoUML is the representation of deep ontologies and that it will be the used modelling language in the practical section of this thesis.

1.5 Modelling Tools

In this chapter the reader will be introduced the modelling tools, that can be used for the modelling this semantics. Each tool will be briefly introduced, with pictures of their UI, pros and cons. Also some online tools will be included. More info can be found in [74].

1.5.1 OLED

The OntoUML lightweight editor (OLED) (figure 1.3) is an environment for developing, evaluating and implementing domain ontologies with the use of UFO–based modelling language OntoUML. It provides the basic set of features such as syntactical verification, visual simulation, model checking, model inference, but also some more advanced like automatic semantic-anti-patterns detection and correction, validation of parthood relations and ontology patterns. It also supports the models from Enterprise Architect[19].

1.5.2 OpenPonk

OpenPonk (visible in figure 1.4), previously known as DynaCASE, is developed by the CCMI[20] research group at the Faculty of Information Technology, CTU in Prague. The potential of this platform is enormous, but in its current version (1.x), it only supports UML and OntoUML. It's planned, that it should also support BORD ORD[21], FSM[22], Petri Nets and DEMO. The platform is built on clean, pure object-based technology Pharo and written in the Smalltalk language. Smalltalk is a simple language, so with just a basic knowledge of this language, you can implement new custom models and algorithms. [76]

[19]Developed by Sprax System, for further information visit `https://sparxsystems.com/`

[20]Centre for Conceptual Modelling and Implementation, more under `https://ccmi.fit.cvut.cz/en/`

[21]Business Objects Relation Modeling Object-Relation Diagrams, more under [75]

[22]Finite State Machines

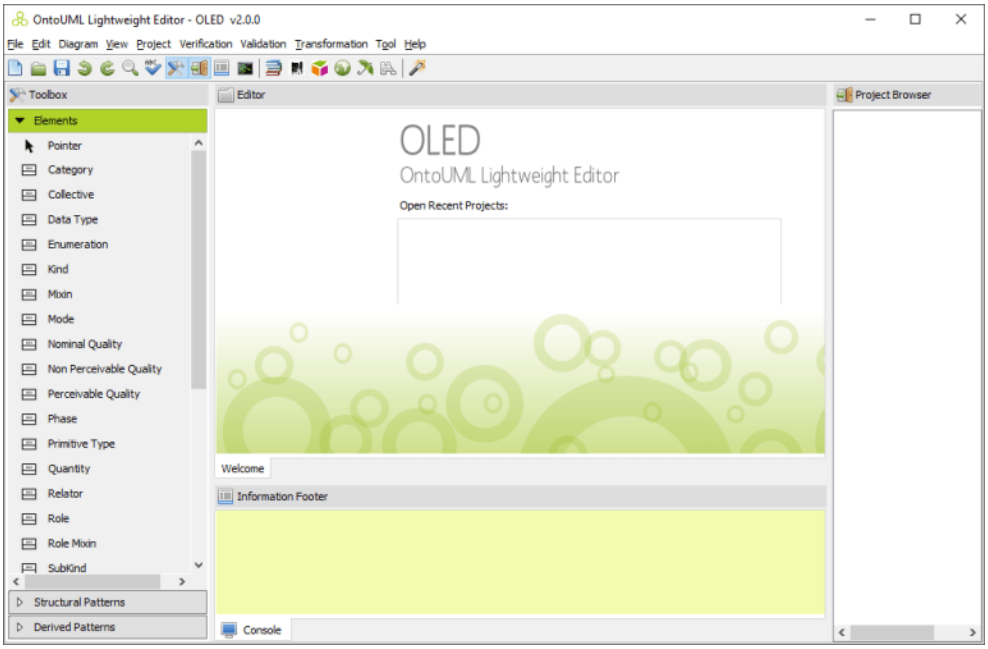


Figure 1.3: OLED UI

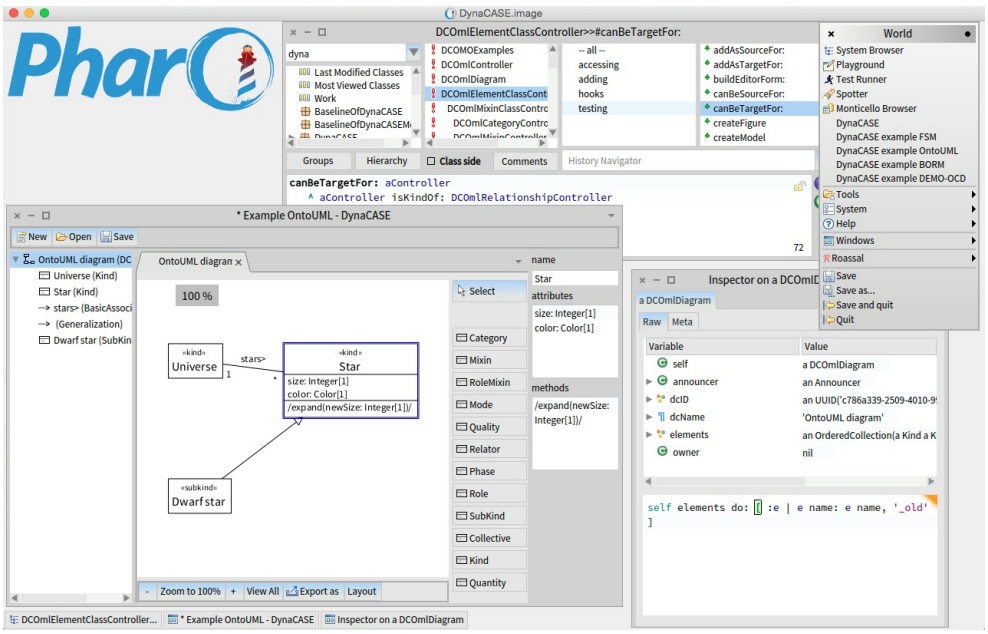


Figure 1.4: OpenPonk UI
With the label of its precursor DynaCASE

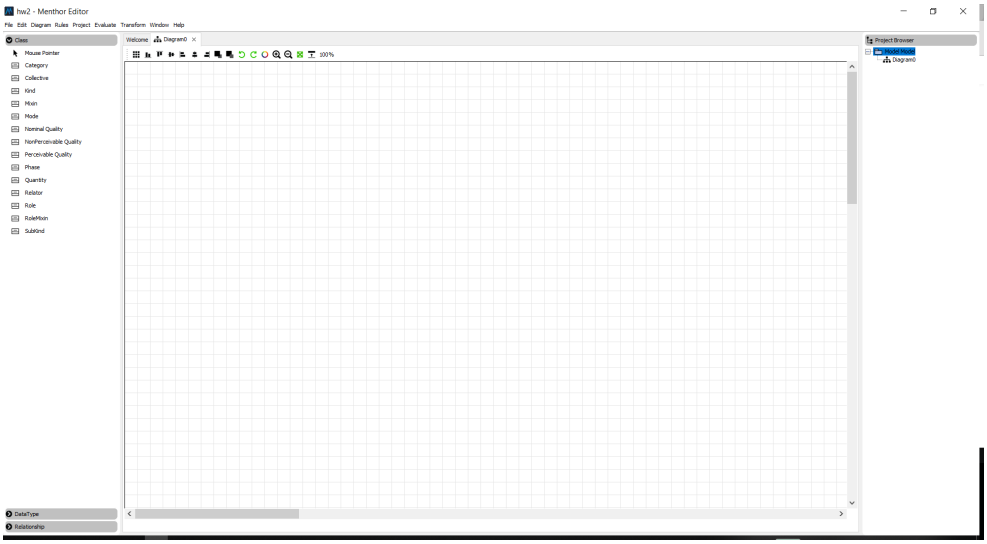


Figure 1.5: Menthor UI

1.5.3 Menthor

Another tool that can be used to model OntoUML diagrams is Menthor, which we can see in figure 1.5.

Menthor is the descendant of the previously mentioned OLED editor. It takes all its pros and adds some more to it. The features include, but not limited to:

- Support of OntoUML and OCL language
- Code generation
- Validation Toolkit
- Documentation tools
- Integration with other modelling tools

The similarity to OLED is visible due to the UI. The modelling surface is situated in the middle, and all available elements are on the left side of the modelling surface. [77]

1.5.4 Umlet

UMLet, a free and open–source tool, is designed for UML and has nice and simple UI (visible in figure 1.6).

It offers a wide range of various features. Starting from primary use–case of drawing diagrams, over to being able to build sequence and activity diagrams straight from plain text, exporting into different formats (e.g. *eps*, *pdf*, *jpg*)

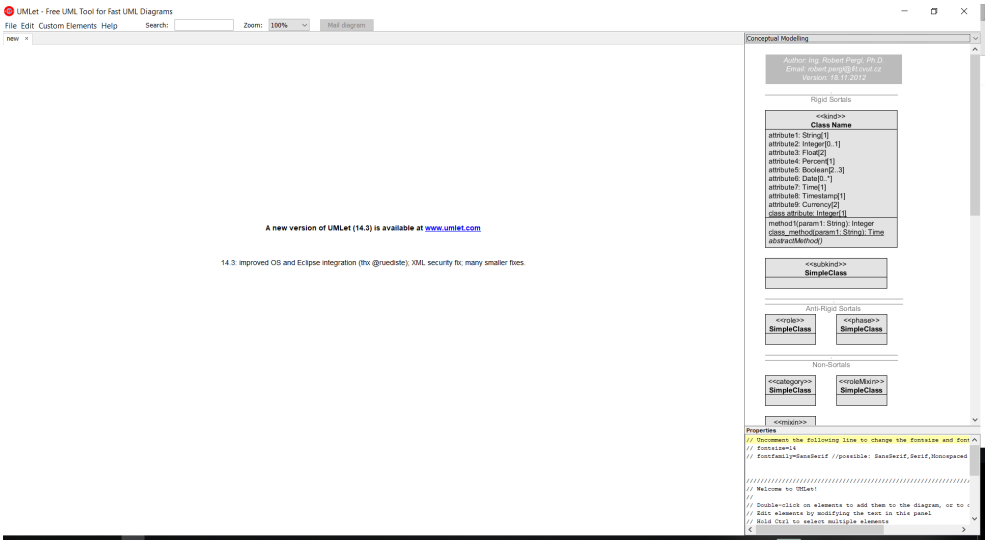


Figure 1.6: UMLet UI
With the pallets for OntoUML

or directly copying it to the clipboard. It also allows the user to design and create new elements, which can then be used in diagrams. It supports all used platforms (*OS X*, *Windows* and *Linux*), it is also available as a plug–in for Eclipse. When using the plug–in version, you can also take advantage of the ability to be able to share your diagrams directly from Eclipse.

UMLet in its default form doesn't support OntoUML. This problem has been solved via three pallets provided by the Faculty of Information Technology[23], CTU in Prague, which can then be extracted into the `pallets` directory of UMLet. [78]

1.5.5 Online tool

There is also an online tool, Draw.io (UI visible in figure 1.7), that offers OntoUML pallets that provide similar functionality as standard desktop applications. The benefit of an online tool is the possibility to work from anywhere, without worrying about having the latest version of the model or having to carry your model on a USB drive for example. All you need is a Google or Microsoft account. The disadvantage is, you need to import the palette[24] from GitHubGist before using it. [79]

[23]Pallets are available under `https://ontouml.org/wp-content/uploads/2016/12/umlet_ontouml_palettes.zip`

[24]Pallets are available under `https://gist.github.com/dunaevskiy/cb7143f4824d05fa65d329f1e3c8cd75`

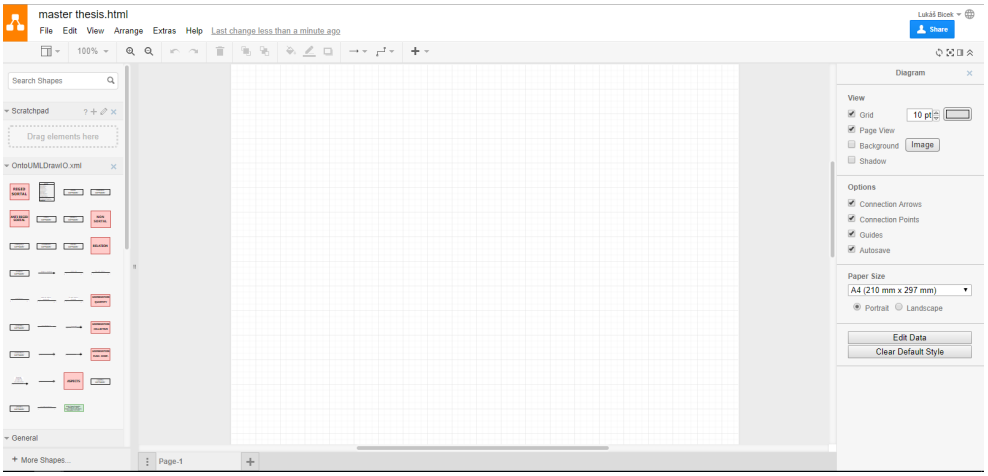


Figure 1.7: draw.io UI
With the pallets for OntoUML

1.5.6 Conclusion and selected tool

In the past, the author of this thesis had the best experience working with UMLet due to it's simple, intuitive and easy-to-use UI. However, the simplicity of UMLet doesn't allow him to use it for this thesis, because it lacks any other features except for exporting into an image output.

Therefore Menthor will be the go–to tool for this thesis, because of the ability to generate also OWL documents, which enables to provide a clear comparison for the provided OWL diagrams prior- and post-enrichment with ontological concepts.

Chapter 2

Transformation of model, a practical example

The practical part of this thesis focuses on transforming a model provided in OWL into a model in OntoUML. The model was kindly provided by [80]. The model was created for the Bring Your Own Data (BYOD) workshop in Rome. This diagram was created, as the author was told, using an opportunistic approach. At the beginning of the work there were three hypotheses set up to be verified by the work on the assignment:

1. Comparing the modelling effort from no model to OWL model, and then from the OWL model to OntoUML model.

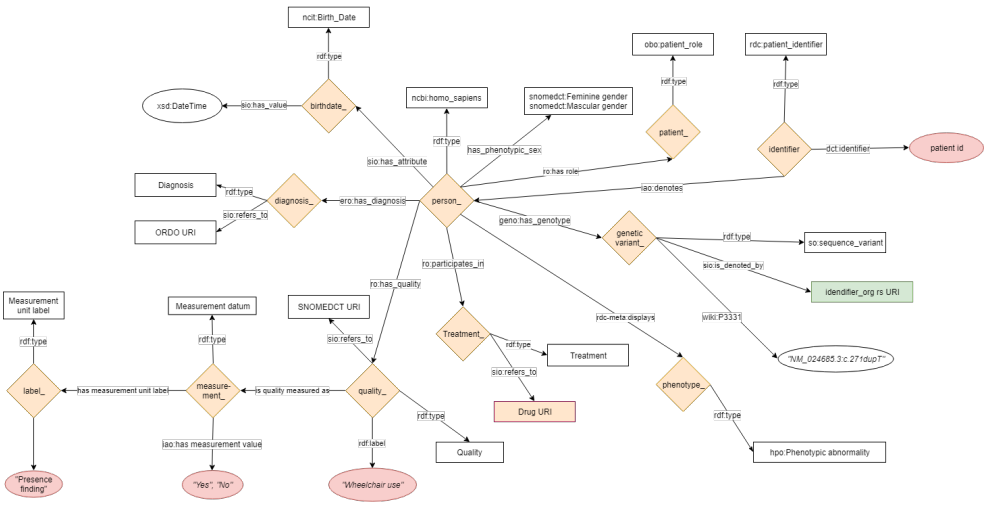


Figure 2.1: Provided model in OWL
Model was part of the BYOD workshop and conference in Rome, courtesy of [80]

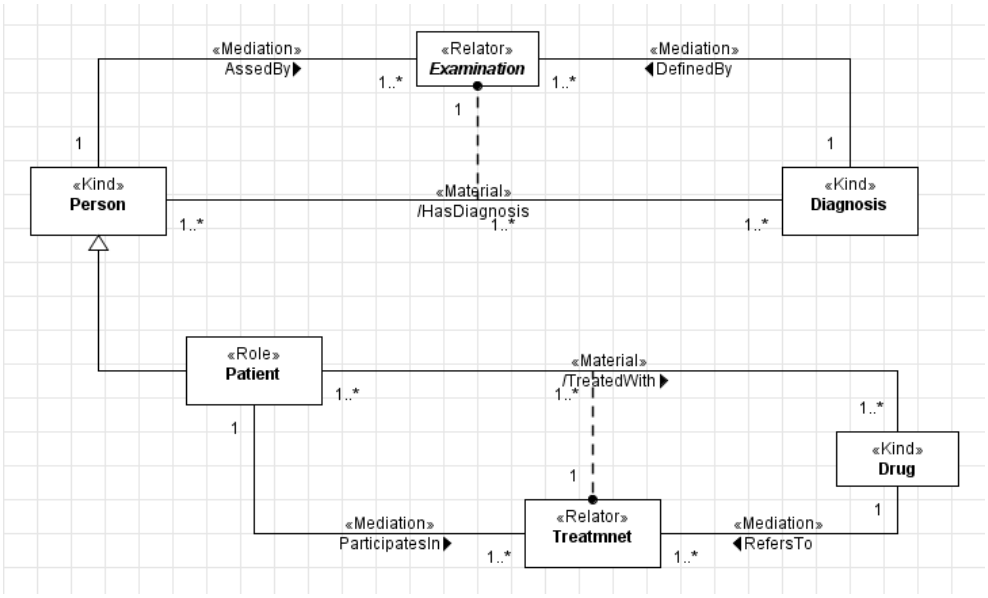


Figure 2.2: First iteration of the model
Model flaws: not all elements of the source model are put in this iteration
Diagnosis has the wrong stereotype

2. The capabilities of the OntoUML improve compared to the OWL model.

3. More in-depth domain knowledge is required, to transform the OWL model into the OntoUML model.

These hypotheses should determine, whether this approach of using deep semantics is feasible and practical for further application in the real world.

2.1 First iteration

The first step was to get familiar with the provided model and to understand all the elements and their meaning. The provided model can be seen in figure 2.1. The fundamental goal during this initial review was to determine the types of components and how they will be connected. This step proved to be difficult, because of the need to follow each type of element or connection to the original ontology, where it is defined. This initial examination has lead to creating of the first iteration of the model (visible in figure 2.2), that was being created using OntoUML. This model is far from optimal or complete. It is missing the left bottom branch of the original model (*quality_*, *measurement_* and *label_*) because there was more clarification and explanation of the meaning of these elements needed. This model also contained wrongly modelled elements.

2.2 Second iteration

The second iteration already contained all the elements from the original model, because of the understanding of the bottom left part of the original diagram. Also, the mistake in the modelling was removed. The type of Diagnosis element was changed from Kind to Mode because it is a characterisation of a role and not a standalone element. There was also an attempt to add attributes to the model. Sadly, at this moment, the author used the types of attributes based on his gut feeling, which in later stages proved to be not optimal, because sometimes, it was the wrong choice. But more on that later in this chapter.

At this stage of the development of the model, the model was also provided to the domain expert for his insights, and this proved to be notably important. The provided insight proved the need to rethink the approach and move one step deeper into the OntoUML modelling language and research the Quality Domains. The usage of normal types of attributes, such as Date, String and so on was insufficient for the general use fo the model. The way out of this trap was by using the Quality Domains, which also allowed to model the different types of measurement.

Quality Domains or Structures provide the ability to model the qualities in detail and also model multidimensional qualities, such as colour representation (RGB[25], HSB[26] or CMYK[27]). In OntoUML, they are represented by Datatypes, which are representing individual Lexical Domains that represent the lexical values of the quality values building the Quality Structure. More on this topic can be found in the slides of [71].

2.3 Third iteration

The third iteration of the model contains the extended modelling of the Measurement NominalQuality and the shift from the usage of bare attributes to Quality Domains which was requested by the domain expert, because of the two different types of measurement. Measurement can be observation based and machine based.

Observation-based measurement represents the observation made without any machine readings. An example can be Ability to walk. The measurement can be observation-based, as well as machine-based. Observation-based measurement could be full ability to walk, limited ability to walk (need of a cane/support) or no ability to walk (e.g. need of wheelchair).

Each observation-based can be supported by a machine-based measurement. That means that the ability to walk measurement can be ”machine”

[25] Red-Green-Blue, a colour model
[26] Hue-Saturation-Brightness, a colour model
[27] Cyan-Magenta-Yellow-Key(blacK), a colour model

measured using walking measuring equipment (e.g. the rehabilitation tool used after traumatic injuries in lower parts of the human body. Machine-based measurement doesn't necessarily mean, that it has to be done by a machine. It means that the measurement is not a pure observation, it is an observation, that can be measured on a specified scale and is objective and not subjective, because it does not involve human judging the measurement. Human is only required to read the value (e.g. meters walked).

2.4 Fourth iteration

The fourth iteration of the model includes the request from the domain expert to include the same detailed description for measurement into the Drug Kind. This was accomplished by adding the Dosage NominalQuanlity and then using the DosageDomain. The contents of the DosageDomain are built on the knowledge and experience of the author.

The liquid dosage domain is self-explanatory. Volume represents the amount of the liquid drug administered. The concentration represents the concentration of the administered drug (e.g. there are different concentrations of disinfection, adrenalin etc.). The unit of liquid dosage is self-explanatory. This can be millilitres, drops, spoons, and so on.

The pill dosage domain is also very self-explanatory. This dosage can have any commonly used drugs, as long as, as the name of the domain suggests, are in pills.

The powder dosage domain was introduced based on the experience of the author of this thesis. The author is using some nourishment drugs for his knees, and one time, he had to use nourishment that was intended to be used for horses (all on the recommendation of a physiotherapist). And these knee nourishments were powder based and dosed based on the weight of the horse. Also later in the evolution of the model, the realisation was made, that the powder based measurement does not necessarily mean it has to have conventional weights units like grams, but there also can be a measuring cup. The progress on the model can be seen in the figure 2.3.

2.5 Fifth iterations

In the fifth iteration, the need for detailed identifier for persons arose. A similar approach was used for other detail modelling of the other Quality Domains. A Quality Domain is encapsulating three domains that can be used for identifying persons. This is not a full listing of ways to identify a person in a system; these three are used as an example. The author chose to model these three ways, based on his experience.

The first domain is the birth date identification domain, the second one is the social security number identification domain, and the last one is the

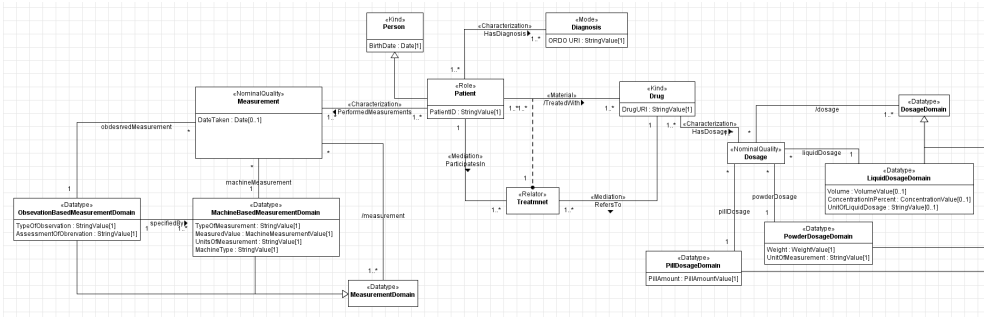


Figure 2.3: Fourth iteration of the model
Model flaws: Missing identification of person, wrong naming in the Powder dosage domain

identification document domain. Under this domain, the reader can imagine the usage of any personal identification card or a passport, hence the two attributes: one for the value and one for the type of the document.

2.6 Sixth iteration

In the sixth iteration, all the small potential problems were addressed. The first issue addressed was the realisation of the wrong naming of one of the attributes in the Powder dosage domain. This issue was easy to fix because it only needed to rename the attribute and the attribute datatype and this didn't change any other, already modelled values.

The second issue addressed was getting rid of the Date Taken attribute in the Measurement NominalQuality. This attribute was transformed into a separate DataType like are the contents of each Quality domain, that is used in this model. This approach was selected due to the multidimensionality of the standard date format. At this point, the decision to model only the standard date type with only numbers was taken, due to the need to model the date as another Quality domain, that would encapsulate both formats.

The third issue that was addressed in this iteration was the gender modelling of person. This point was forgotten by the author, or maybe it was considered automatic, so the focus shifted to different topics. This issue was fixed easily by adding sub-kinds to the Person kind. To comply with all the gender talk that are happening and the gender neutrality, the author chose to model Male, Female and Other gender types, to make the possibility to include instances of people, who don't identify themselves as males, or females.

The final issue addressed was the modelling of the time. Requested was the delay between administration of a drug and the time passed until the drug took effect on the patient. This proved to be a challenge, because of the OntoUML language is based on the UFO-A ontology, which is, like it was

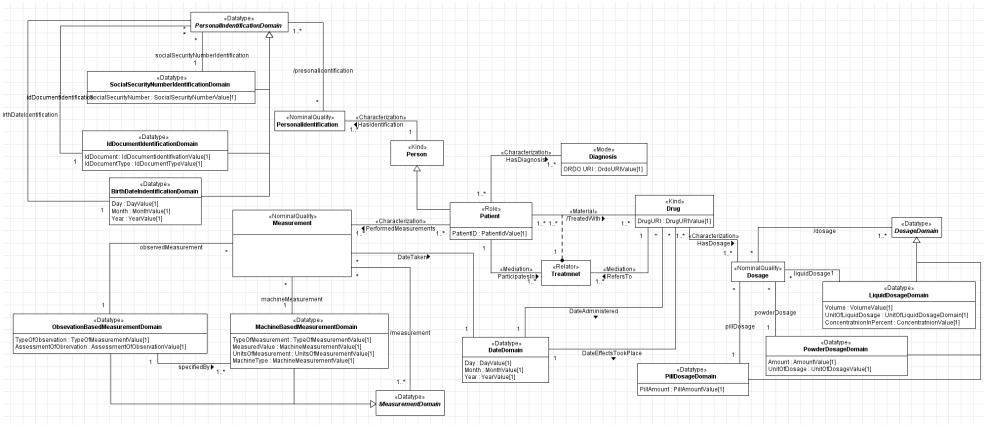


Figure 2.4: Sixth iteration of the model
Further model improvements: Detailed time modelling

mentioned above in the section 1.4.2, is the only one that is concerned finished without any ongoing research and is verified on practical projects. Hence the author decided to model them using a normal Data type and two connections to this Data type (*DateAdministered* and *DateEffectTookPlace*). The author is not sure if this approach is the correct one because it is a part of UFO-B, where the research is still ongoing.

2.7 Simulation of model and issues discovered during the simulation

After the finished process of modelling, the model (in figure 2.4) was then checked using the tools, that the Menthor editor provides. The first tool used is the Check OCL Rules tool, that verifies, whether the OCL [28] rules are valid. This procedure yielded zero errors. Second tool Check OntoUML Model yielded several errors that forced the author to seek the help of his supervisor. After the consultation, most of the errors came from the strict enforcement, that attributes of entities are not part of the entity but connected to the entity via a relation. The author and his supervisor decided to ignore these errors since both representations are syntactically valid.

Then the simulation using the Alloy[29] integration of the Menthor editor. The running the simulation discovered several issues:

1. Individual Identification Domains were associated with their supertype - IdentificationDomain. The correct association if with PersonalIdenti-

[28] Object Constraint Language

[29] More about Alloy can be found under `https://en.wikipedia.org/wiki/Alloy_(specification_language)` or under `http://alloytools.org/`

fication Quality. This can be seen in figure 2.5 (PersonalIdentification does not have any specific domains associated).

2. The specifiedBy relations should have 0..1 cardinality at the side of ObservationBasedMeasurementDomain, not 1.

3. Another crucial issue discovered by simulation was the cardinality by Identification Domains. All relations between them and the PersonalIdentification Quality should have a one at the side of the quality.

2.7.1 Final version of the model

The revelations brought by the simulations transformed the model into the final version of it as it can be seen in figure 2.7 (for full size refer to figure B.10). The changes, that have been done, have been mentioned in the enumeration above when discussing the problems discovered by the simulations.

The resulting model has its centre in the *Patient* role, which is connected to all the important entities. The patient is the role of kind *Person* which has the quality PersonalIdentification, which is connected to the *PersonalIdentificationDomain*. An abstract supertype for:

IdDocumentIdentificationDomain: representing the possibility of identifying a person using documents, such as a passport

SocialSecurityNumerIdentificactionDomain: a social security number has to have every person

BirthDateIdentificationDomain: a person can be identified by their birth date

The *patient* is connected to the mode Diagnosis via a *Characterization* relation. The patient can have measurements done to him. *Measurement*, a Quality, is specified by a Measurement Domain. The same principle was applied as with the *PersonalIdentificationDomain*. This supertype has two subtypes:

ObservationBasedMeasurementDomain: representing a human observation, like the ability to walk, with the observations full ability, moderate ability (walking stick required), no ability to walk

MachineBasedMeasurementDomain: represents measurements, where some equipment is used. Unter equipment one can imagine measuring tapes, all sorts of rehabilitation tools etc. This domain also specifies the *ObservationBasedMeasurementDomain*, by supporting the observation by readings from the equipment.

The second kind in this model, *Drug*, is connected to the Patient role via a *Material* relation, with the truthmaker in the *Treatment* relator. Drug and

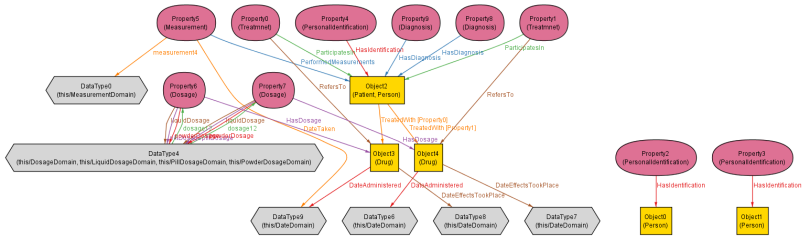


Figure 2.5: Invalid Alloy simulation of the model prior fixing all the issues

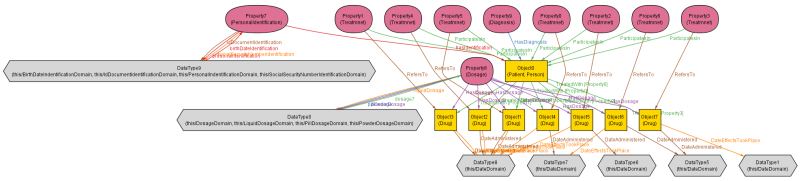


Figure 2.6: Valid simulation of the model in figure 2.7

Measurement are also associated with the *DateDomain*, which is the chosen way to represent time since there are no events yet in OntoUML (they are part of UFO-B, which is not in OntoUML). The drug has a quality, *Dosage*, which is specified by the supertype DosageDomain with the subtypes:

PillDosageDomain: represents the standard for the drugs that are in the form of pills

PowderDosageDomain: author chose to include this domain based on his personal experience using supplements for supporting the ligaments in knees and meniscus

LiquidDosageDomain: is for all liquid drugs in different concentrations and volumes

The simulation of this model can be seen in figure 2.6, where in comparison to figure 2.5 are no loose *personalIdentification* instances.

2.8 Further discussion about improvements

An improvement can be made to the model is focusing at the time modelling mentioned above (in section 2.6). The UFO-B ontology provides the basic knowledge to for this topic. The time notation consists of events, relations between events, that are structured like humans normally structure time, e.g. before, after, during and so on. This implementation could aid and contribute to the research and the development of the UFO-B notation and provide a practical reference point for this theory.

Also, the Quality domains can be extended to comply with all the needs. Measurement Quality domain needs, in the author's eyes no extension. Extensions to the PersonalIdentification domain can be done to match and include all the possible way to identify a human being. The author is confident that there are more ways that he modelled in the model and that's why there is room for domain experts and other people to step in the development of the model. The same procedure can apply to the Dosage domain. There is also a possibility to model the Treatment relator more detailed, because not always there is a drug needed for treatment. Right now the model does not accept treatment where no drug is administered or treatments where the administered drug is a pain medication, and it is only supporting the primary treatment. An example of such treatment are bruises, broken bones, torn ligaments and so on.

The mode Diagnosis is represented as a single entity. This could be potentially improved by extending it by specifying the type of the diagnosis more precisely, apart from including the ORDO URI. There can also be several subtypes of the diagnosis such as fatal, temporary, standing, and so on. Further it there can also be modelled the person who issued this diagnosis, the description of the diagnosis, and so on.

Although the modelling of the Measurement Domains is ontologically correct, there is the need to specify the constraints for the relations between Measurement and the specific domains. Could measurement have both at the same time, or should it be obervation XOR[30] machine. This is also a point for further discussion.

This model provided a strong and stable foundation for future development and was modelled in such way that the potential enrichment of the model can be done easily.

2.9 Conclusion of practical task and thoughts about shallow and deep ontologies

All together the transformation of the model seemed easy at first glance; it proved to be rather complex. Because of the need to follow the links to the vocabularies that describe the elements of the OWL model. The first iterations went smoothly because the author followed the opportunistic approach, same as the authors of the original model did. This approach needed to be rethought because of the needs to model more complex structures, that cannot be modelled out of the blue. This also forced the author to research more on quality modelling and quality structures/domains. The decision to use quality domains made the model more expandable and useable. The final model is presented in figure 2.7.

[30]XOR = exclusive or (it can be only one of the option, not both at the same time)

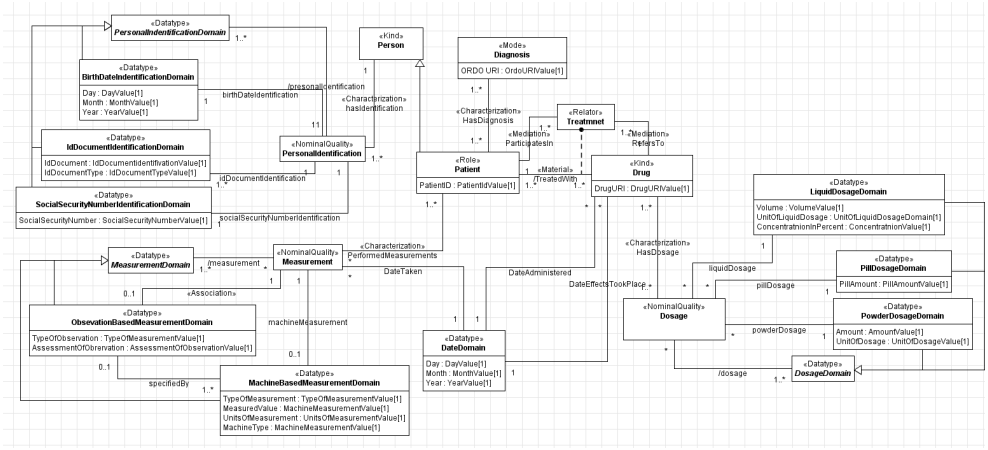


Figure 2.7: Final version of the model, after all the issues and faults were fixed.

The author may bias the comparison between shallow and deep ontologies because he has only experience using the so-called deep ontologies in OntoUML.

The benefits of shallow ontologies contain the easy-to-use approach and modelling. That means the ontologies are ready-to-use vocabularies which only need to be used. However, it is difficult to understand the model, what the ”classes” are, what are the ”attributes” and what are the relations. Also, there is no multiplicity directly at the relations; one must follow the type of relation into the ontology vocabulary to find out the multiplicity of the connection. Also, in the author's opinion, the OWL diagrams are difficult to understand, due to the representation as an undirected graph. Sometimes, as it was the case of the provided model (figure 2.1), the direction of relations can be determined according to the types of those relations, when looked up in the vocabulary.

In the author's opinion, once again, it might be biased due to the limited to none experience working with those models, is the big downside of OWL. The readability of the model is difficult for inexperienced users. However, a contra-argument was made during the creation of the model by the domain expert, for whom it was easier to orient himself in OWL model than in OntoUML model, but the same bias applies here: he is more experienced in OWL and has zero to none experience in OntoUML.

Another downside that comes to the author's mind, is the need of the vocabularies. When someone is only provided with the model, without the vocabularies, he or she has challenging times to understand the model. Also very often the vocabularies overlap themselves or are built upon one another, so to understand the model, the user or reader of the model has to do a lot of reading and research to get him-/herself familiar with all the terms to be

able to understand the model entirely.

To create a brand new model for a domain, that does not have any models, that the creator of the model can lean on, means two things: acquire the skills to by himself, or hire someone with those skills. Since there are no vocabularies present, they have to be created, which is not easy. It will most likely result in hiring a domain architect to create these ontologies for him, which comes hand in hand with enormous cost and time demand. And it is not profitable in the long shot since the skill of creating such domains leaves after the vocabulary is created. The second option is to acquire the skill himself, which is time demanding, but in the long run, provides better outcome - skills stay in-house.

The skill acquisition proved to be the only disadvantage of OntoUML that the author found. Shallow ontologies allow creators after they find the right vocabularies, which may overlap, exist several of them that do not work together, take the terms from those vocabularies and use them without deeper understanding the domain. The popular quote from the internet can be applied to this approach: "If it sits, it fits".

So-called deep ontologies, in this case, UFO and it's notation OntoUML have the benefits of being built upon UML. Therefore they have all the modelling advantages of the Class diagram that UML provides. The relations can be directed eighter by making an arrow out of them or adding a pointer to the description of the relationship. This, however, can prove to be difficult for someone, who isn't familiar with this notation, because to them, it's just boxes and lines. Using OntoUML brings the benefit of not needing to include the references to the vocabularies because all elements of the model (as they are described in section 1.4.3).

This is a double-edged sword because all the information is hidden under the types of the entities and for them, the need to understand and knowing OntoUML is crucial. Some types of objects can be determined according to the name of the type (Kind, Subkind, Role) but other, such as Mode, Relator can be challenging to determine the meaning of them. The same applies to relations.

Overall, the author's opinion is, based on the arguments above, that OntoUML is richer on meta-data than OWL, because all the important information about the model and it's elements is included into the model itself, without any need to link other vocabularies. OntoUML, thanks to its origin, provides standardised notation used in software development. All this being said, this is only the author's opinion and is based on his personal experience.

The usage of OntoUML, thus deep ontologies, means the models force a better understanding of the modelled domain. This goes hand in hand with the concepts of GO-FAIR, that are all about the machine actionability. OntoUML forces a better understanding of the domain, thus better understanding of the entities that have to be modelled. All this knowledge of the domain and its entities enables the modellers to create more precise and expressive

metadata. What, once again, means that overall the model will contain more metadata, which, coming back to the machine actionability, provides more instruments to the machines that they can act on. The machine actionability strongly supports the Reusability principles of GO-FAIR.

Conclusion

The core focus of this thesis was the comparison between shallow and deep semantics and to provide arguments for the usage of deep semantics, such as the Unified Foundation Ontology and its language, OntoUML. At the beginning of the practical task (section 2), the author, his supervisor and the domain experts set up three hypotheses.

Hypothesis evaluation

The first hypothesis, or more likely the time comparison was after some discussion among the participated individuals since it would not have enough statement power, because of the one-sided experience of the author. It would be like comparing apples and pears. The author kept, however, track of the time used for the transformation of the model from OWL into OntoUML. Overall it took somewhere around the 20 man-days to complete the transformation. This includes consultations and discussions about the model, actual modelling as well as additional research of newly discovered gaps.

Second hypothesis questions the capabilities of the transformed model. OntoUML model is more explicit and has more meaningfulness. A simple example is the no-need to follow links into vocabularies to be able to understand the model on its own, thanks to the fact that OntoUML is a profile for UML class diagram, which most of the people in the modelling branch of computer science can interpret. There are also additional ways to add the constraints to the model thanks to the expressive cardinality on the relations. OntoUML also allows helping the reader of the model to navigate the relations between the objects, by allowing the modelling the directions of those relations. All of this provides a strong foundation for a more homogenous interpretation of the model. Also, another capability was the, actually really spontaneously done by the author, shift of the diagnosis, drug or measurement from Person to the Patient role, which is more ontologically correct, because as a person one does

not have any diagnoses, drugs prescribed or measurements taken, this only happens when the persons becomes a patient in any healthcare institution.

Methor issues

The only complication that arose during the work on the thesis were the technical complications caused by Menthor. The editor has bricked[31] the model several times via different means. Once it was a remnant of a deleted relation that remained at the drawing board although, it was correctly removed from the model, next time it ware duplicate relations that weren't showing in the diagram, and so on. Troubleshooting was difficult, and sometimes it required reverting to older versions of the model and redoing all the lost work.

Sometimes the editor hides the attributes of the entities even thou they are set to be displayed. Do not take the author wrong, Menthor editor is a good editor, but it has some issues that need to be solved (all the encountered issues were reported to the author of Menthor editor).

Author's last words

The last hypothesis was questioning the need for a deeper understanding. This hypothesis was proven already halfway through the transformation, where the need to understand, how the measurement is supposed to work, so it can be modelled appropriately. Therefore the more profound knowledge of is required to being able to model the domain accurately and as close to the reality as possible.

In the author's opinion, the thesis fulfilled the task that was set up: To support the usage of the deep semantics to help turn the data more FAIR and therefore the equip the data with more richer, descriptive and meaningful metadata. The thesis also proves that the concept of UFO and OntoUML is the prefered way to proceed because the modelling is explicit and provided models are simple to understand.

The author of the thesis is overall satisfied with the result. The thesis provided him with the possibility to impersonate the role of domain modeller. This opportunity that tested all previously acquired skills during his stay at the Faculty of Informatics of the Czech Technical University in Prague provided him with enough experience to confirm his decision that this is the field of expertise, to which he would like to dedicate his work-life after finishing the university.

[31] bricked = model wasn't showing at the drawing board even thou it was opened. Control elements started to show themselves after being hovered over by the mouse.

Bibliography

[1] GO FAIR. online, 2018. Available from: https://www.go-fair.org/

[2] GO FAIR Initiative. Available from: https://www.go-fair.org/go-fair-initiative/

[3] Wilkinson, M.; et al. The FAIR Guiding Principles for scientific data management and stewardship. *Nature Scientific Data*, , no. 160018, 2016. Available from: http://www.nature.com/articles/sdata201618

[4] Hypertext Transfer Protocol. online. Available from: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

[5] File Transfer Protocol. online. Available from: https://en.wikipedia.org/wiki/File_Transfer_Protocol

[6] Simple Mail Transfer Protocol. online. Available from: https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

[7] JavaScript Object Notation for Linked Data. online. Available from: https://en.wikipedia.org/wiki/JSON-LD

[8] Winskel, G. *The Formal Semantics of Programming Languages: An Introduction*. Cambridge, MA, USA: MIT Press, 1993, ISBN 0-262-23169-7.

[9] Stoy, J. E. *Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory*. Cambridge, MA, USA: MIT Press, 1977, ISBN 0262191474.

[10] Schmidt, D. A. *Denotational Semantics: A Methodology for Language Development*. Dubuque, IA, USA: William C. Brown Publishers, 1986, ISBN 0-697-06849-2.

[11] D. Plotkin, G. A Structural Approach to Operational Semantics. *J. Log. Algebr. Program.*, volume 60-61, 07 2004: pp. 17–139, doi:10.1016/j.jlap.2004.05.001.

[12] Plotkin, G. D. The origins of structural operational semantics. *J. Log. Algebr. Program.*, volume 60-61, 2003: pp. 3–15, doi:10.1016/j.jlap.2004.03.009.

[13] Kahn, G. Natural Semantics. In *Proceedings of the 4th Annual Symposium on Theoretical Aspects of Computer Science*, STACS '87, Berlin, Heidelberg: Springer-Verlag, 1987, ISBN 3-540-17219-X, pp. 22–39. Available from: `http://dl.acm.org/citation.cfm?id=646503.696269`

[14] Hoare, C. A. R. An Axiomatic Basis for Computer Programming. *Commun. ACM*, volume 12, no. 10, Oct. 1969: pp. 576–580, ISSN 0001-0782, doi:10.1145/363235.363259. Available from: `http://doi.acm.org/10.1145/363235.363259`

[15] Cyganiak, R.; Wood, D.; et al. RDF 1.1 Concepts and Abstract Syntax. online, 2004. Available from: `http://www.w3.org/TR/rdf11-concepts/`

[16] Resource Description Framework (RDF). online, 2014. Available from: `http://www.w3.org/RDF/`

[17] Gandon, F.; Schreiber, G. RDF 1.1 XML Syntax. online, 2014. Available from: `http://www.w3.org/TR/rdf-syntax-grammar/`

[18] Brickley, D.; Guha, R. V. RDF Schema 1.1. online, 2014. Available from: `https://www.w3.org/TR/rdf-schema/`

[19] Becket, d.; Berners.Lee, T.; et al. RDF 1.1 Turtle, Terse RDF Triple Language. online, 2014. Available from: `https://www.w3.org/TR/turtle/`

[20] Davis, I.; Steiner, T.; et al. RDF 1.1 JSON Alternate Serialization (RDF/JSON). online, 2013. Available from: `https://www.w3.org/TR/rdf-json/`

[21] Becket, D.; Berners.Lee, T.; et al. RDF 1.1 N-Triples, A line-based syntax for an RDF graph. online, 2014. Available from: `https://www.w3.org/TR/n-triples/`

[22] About, What is a Nanopublication. online. Available from: `http://nanopub.org/wordpress/?page_id=65`

[23] Gray, A. J. G.; Chichester, C.; et al. Nanopublication Guidelines. online, 2014. Available from: `http://www.nanopub.org/2013/WD-guidelines-20131215/`

[24] Harvey, G. *Archelogia philosophica nova, or, New principles of philosophy containing philosophy in general, metaphysicks or ontology, dynamilogy or a discourse of power, religio philosophi or natural theology, physicks or natural philosophy*. London: Geo. Stradling, S. T. P. Rev. in Christo Pat. Gilb. Episc. Lond. a Sac. Domest, 1663. Available from: http://name.umdl.umich.edu/A43008.0001.001

[25] Griswold, C. L. *Platonic writings/Platonic readings*. University Park: Pennsylvania State University Press, 2002, ISBN 978-0-271-02137-9.

[26] Petrov, V. (editor). *Ontological landscapes*. Frankfurt: Ontos Verlag, 2011, ISBN 978-3-86838-107-8.

[27] Gruber, T. What is an Ontology? Available from: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[28] Gruber, T. R. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, volume 43, no. 5, 1995: pp. 907 – 928, ISSN 1071-5819, doi:https://doi.org/10.1006/ijhc.1995.1081. Available from: http://www.sciencedirect.com/science/article/pii/S1071581985710816

[29] Gruber, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition*, volume 5, no. 2, 1993: pp. 199 – 220, ISSN 1042-8143, doi:https://doi.org/10.1006/knac.1993.1008. Available from: http://www.sciencedirect.com/science/article/pii/S1042814383710083

[30] McNeill, F.; Bundy, A. Dynamic Ontology Repair. Available from: http://dream.inf.ed.ac.uk/projects/dor/

[31] BFO: Basic Formal Ontology. Available from: http://basic-formal-ontology.org/

[32] Arp, R.; Smith, B.; et al. *Building ontologies with Basic Formal Ontology*. Cambridge, Massachusetts: Massachusetts Institute of Technology, [2015], ISBN 978-026-2527-811.

[33] Herre, H.; Heller, B.; et al. General Formal Ontology (GFO) - A Foundational Ontology Integrating Objects and Processes [Version 1.0]. 07 2006.

[34] Donnelly, M.; Guizzardi, G. *Formal ontology in information systems*. Washington, D. C.: IOS Press, c2012, ISBN 978-1-61499-083-3.

[35] Taniar, D.; Rahayu, J. W. *Web semantics and ontology*. Hershey, PA: Idea Group Pub., c2006, ISBN 978-1-59140-907-6.

[36] Blaško, M. *Ontologies and Semantic Web*. Available from: https://cw.fel.cvut.cz/old/_media/courses/osw/lecture-06upperontologies-s.pdf

[37] Figay, N. Do you know what Upper Ontologies are and the value it brings? 2017. Available from: https://www.linkedin.com/pulse/upper-ontologies-dr-nicolas-figay/

[38] FOAF. Available from: http://www.foaf-project.org/

[39] The OBO Foundry. Available from: http://obofoundry.org/

[40] Smith, B.; Ashburner, M.; et al. The OBO Foundry. *Nature Biotechnology*, volume 25, no. 11, 2007: pp. 1251–1255, ISSN 1087-0156, doi:10.1038/nbt1346. Available from: http://www.nature.com/articles/nbt1346

[41] Kibbe, W. A.; Arze, C.; et al. Disease Ontology 2015 update: an expanded and updated database of human diseases for linking biomedical knowledge through disease data. *Nucleic Acids Research*, volume 43, no. D1, 10 2014: pp. D1071–D1078, ISSN 0305-1048, doi:10.1093/nar/gku1011, http://oup.prod.sis.lan/nar/article-pdf/43/D1/D1071/17435884/gku1011.pdf. Available from: https://doi.org/10.1093/nar/gku1011

[42] Disease Ontology. Available from: http://www.disease-ontology.org/

[43] Genotype Ontology. Available from: https://github.com/monarch-initiative/GENO-ontology/

[44] eagle-i Research Resource Ontology. Available from: https://open.med.harvard.edu/wiki/display/eaglei/Ontology

[45] Information Artifact Ontology. Available from: https://github.com/information-artifact-ontology/IAO/

[46] Semanticscience Integrated Ontology. Available from: https://github.com/MaastrichtU-IDS/semanticscience

[47] Dublin Core Metadata Initiative. Available from: http://www.dublincore.org/specifications/dublin-core/

[48] Woods, W. A.; Schmolze, J. G. The KL-ONE family. *Computers & Mathematics with Applications*, volume 23, no. 2, 1992: pp. 133 – 177, ISSN 0898-1221, doi:https://doi.org/10.1016/0898-1221(92)90139-9. Available from: http://www.sciencedirect.com/science/article/pii/0898122192901399

[49] Baader, F.; Horrocks, I.; et al. Chapter 3: Description Logics. In *Handbook of knowledge representation*, Boston: Elsevier, 2008, ISBN 978-0-444-52211-5, pp. 135–179.

[50] Horrocks, I.; Patel-Schneider, P. F. The Generation of DAML+OIL. *Working Notes of the 2001 International Description Logics Workshop (DL2001)*, volume 49, 01 2001: pp. 30–35.

[51] Grau, B. C.; Horrocks, I.; et al. OWL 2: The next step for OWL. *Journal of Web Semantics*, volume 6, no. 4, 2008: pp. 309 – 322, ISSN 1570-8268, doi:https://doi.org/10.1016/j.websem.2008.05.001, semantic Web Challenge 2006/2007. Available from: `http://www.sciencedirect.com/science/article/pii/S1570826808000413`

[52] Tsarkov, D.; Horrocks, I. FaCT++ description logic reasoner: System description. In *In Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006*, Springer, 2006, pp. 292–297.

[53] Fensel, D.; Harmelen, F. V.; et al. OIL: An ontology infrastructure for the semantic web. In *McGuinness DL, PatelSchneider PF*, 2001, pp. 38–45.

[54] Lewis, C. I. *The place of intuition in knowledge*. Ph.d. thesis, Harvard University, Cambridge, 1910.

[55] Lewis, C. I.; Langford, C. H. *Symbolicc logic*. Dover Publications, first edition, 1932.

[56] McKinsey, J. C. C. A solution of the decision problem for the Lewis systems S2 and S4, with an application to topology. *Journal of Symbolic Logic*, volume 6, no. 4, 1941: p. 117–124, doi:10.2307/2267105.

[57] Goldblatt, R. Mathematical modal logic: A view of its evolution. *Journal of Applied Logic*, volume 1, no. 5, 2003: pp. 309 – 392, ISSN 1570-8683, doi:https://doi.org/10.1016/S1570-8683(03)00008-9. Available from: `http://www.sciencedirect.com/science/article/pii/S1570868303000089`

[58] Fitting, M.; L. Mendelsohn, R. First-Order Modal Logic. 01 1998, doi: 10.1007/978-94-011-5292-1.

[59] Barcan, R. C. The identity of individuals in a strict functional calculus of second order. *Journal of Symbolic Logic*, volume 12, no. 1, 1947: p. 12–15, doi:10.2307/2267171.

[60] Bobzien, S. *Chrysippus' Modal Logic and Its Relation to Philo and Diodorus*. 01 1993, ISBN 13-978-3515062084, pp. 63–84.

[61] Bobzien, S. Ancient Logic. In *The Stanford Encyclopedia of Philosophy*, edited by E. N. Zalta. Available from: `https://plato.stanford.edu/archives/win2016/entries/logic-ancient/`

[62] Spade, P. V.; Hintikka, J. J. History of logic. Aug 2017. Available from: https://www.britannica.com/topic/history-of-logic

[63] Ohlbach, H. J.; Koehler, J. Modal logics, description logics and arithmetic reasoning. *Artificial Intelligence*, volume 109, no. 1, 1999: pp. 1 – 31, ISSN 0004-3702, doi:10.1016/S0004-3702(99)00011-9. Available from: http://www.sciencedirect.com/science/article/pii/S0004370299000119

[64] Guizzardi, G. *Ontological foundations for structural conceptual models*. Centre for Telematics and Information Technology, Telematica Instituut, 2005.

[65] Marek, J. *Model klienta veřejné správy z pohledu Unified Foundational Ontology*. Master thesis, University of Economics, Prague, 2017.

[66] Guizzardi, G.; Wagner, G. *Using the Unified Foundational Ontology (UFO) as a Foundation for General Conceptual Modeling Languages*. 03 2010, pp. 175–196, doi:10.1007/978-90-481-8847-5_8.

[67] Blaško, M. *Unified Foundational Ontology and Ontology Testing*. Available from: https://cw.fel.cvut.cz/old/_media/courses/osw/lecture-07ufo-s.pdf

[68] Pergl, R. *Conceptualisation: Chapters from Harmonising Enterprise and Software Engineering*. Habilitation thesis, Faculty of Information Technology, Czech Technical University in Prague, Prague, 2018.

[69] Pergl, R. *Lectures of BIE-KOM - Conceptual Modelling*. Available from: https://moodle.fit.cvut.cz/course/view.php?id=225

[70] Pergl, R. *Lectures of BI-KOM - Conceptual Modelling*. Available from: https://moodle.fit.cvut.cz/course/view.php?id=32

[71] Guizzardi, G. *Lectures of Ontology-Driven Conceptual Modeling*. Brasil: Ontology and Conceptual Modeling Research Group (NEMO), Federal University of Espírito Santo.

[72] Bassetti, L. OntoUML Specification. Available from: https://ontology.com.br/ontouml/spec/index.html

[73] Guizzardi, G.; Fonseca, C. M.; et al. Endurant types in ontology-driven conceptual modeling: Towards OntoUML 2.0. In *International Conference on Conceptual Modeling*, Springer, 2018, pp. 136–150.

[74] Tooling. online, 2017. Available from: https://ontouml.org/ontouml/tooling/

[75] BORM ORD. online, 2016. Available from: https://ccmi.fit.cvut.cz/methodologies/borm/

[76] OpenPonk. online, 2016. Available from: https://ccmi.fit.cvut.cz/tools/openponk/

[77] Menthor. online. Available from: https://github.com/MenthorTools/menthor-editor/releases

[78] UMLet. online. Available from: https://www.umlet.com

[79] draw.io. online, 2017. Available from: https://www.draw.io/

[80] Jacobsen, A.; Cornet, R.; et al. Current practices to make rare disease registries FAIR. *OSF*, Feb 2019. Available from: https://osf.io/aym2t/

[81] UFO-A package model. Available from: https://ontology.com.br/ufo/ufo-a/spec/index.html

[82] UFO-B package model. Available from: https://ontology.com.br/ufo/ufo-b/spec/index.html

[83] UFO-C package model. Available from: https://ontology.com.br/ufo/ufo-c/spec/index.html

Appendix A

Acronyms

AI Artificial intelligence

BFO Basic Formal Ontology

BYOD Bring Your Own Data

CMYK Cyan-Magenta-Yellow-Key(blacK), a colour model

DAML DARPA Agent Markup Language

DARPA US Defense Advanced Research Projects Agency

DL Description Logic

DOI Data Object Identifier

ER Entity–Relationship

FOAF Friend of a Friend

FOL First-order logic

FTP(S) File Transfer Protocol (Secure)

GFO General Formal Ontology

HTTP(S) Hypertext Transfer Protocol (Secure)

HSB Hue-Saturation-Brightness, a colour model

ID Identifier, symbol uniquely identifying an object or record

iff if and only if

JSON JavaScript Object Notation

KR Knowledge representation

MIAME Minimum information about a microarray experiment

MIT Massachusetts Institute of Technology

ML Modal Logic

NDA Non-Disclosure Agreement

OBO Open Biomedical Ontologies Foundry

OCL Object Constraint Language

OIL Ontology Inference Language

ORCID Open Research and Contributor ID

OWL Web Ontology Language

SMTP Simple Mail Transfer Protocol

RDF Resource Description Framework

RGB Red-Green-Blue, a colour model

TCP Transmission Control Protocol

UFO Unified Foundation Ontology

UI User Interface

UML Unified Modeling Language

US United States

USB Universal Serial Bus

W3C World Wide Web Consortium

XML Extensible markup language

XOR Exclusive or

Appendix B

Full-sized images used in thesis

In this appendix, fill size figures that are in the thesis will be presented, among them all the images used in chapter 2 will be presented again, in a bigger format, so the reader doesn't need to refer to the included CD into the exports subfolder of the Menthor folder (folder structure visible in appendix C).

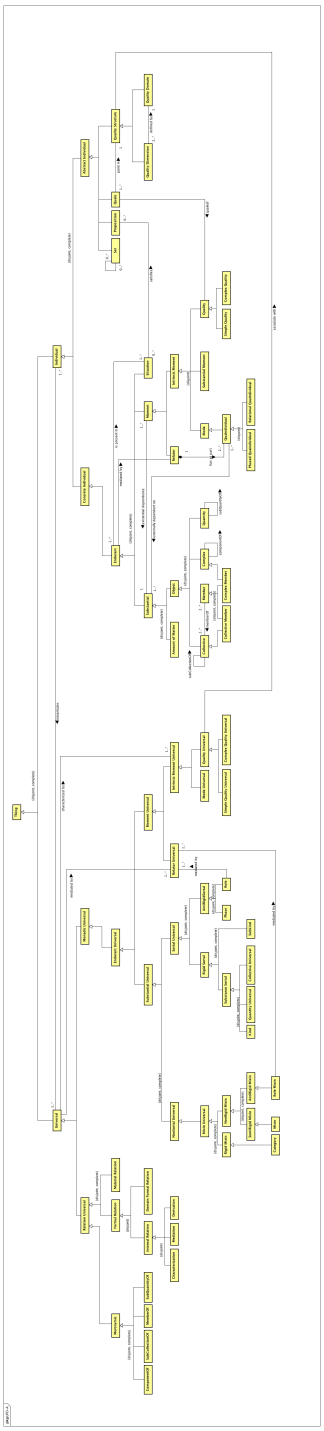


Figure B.1: UFO-A metamodel from [81]

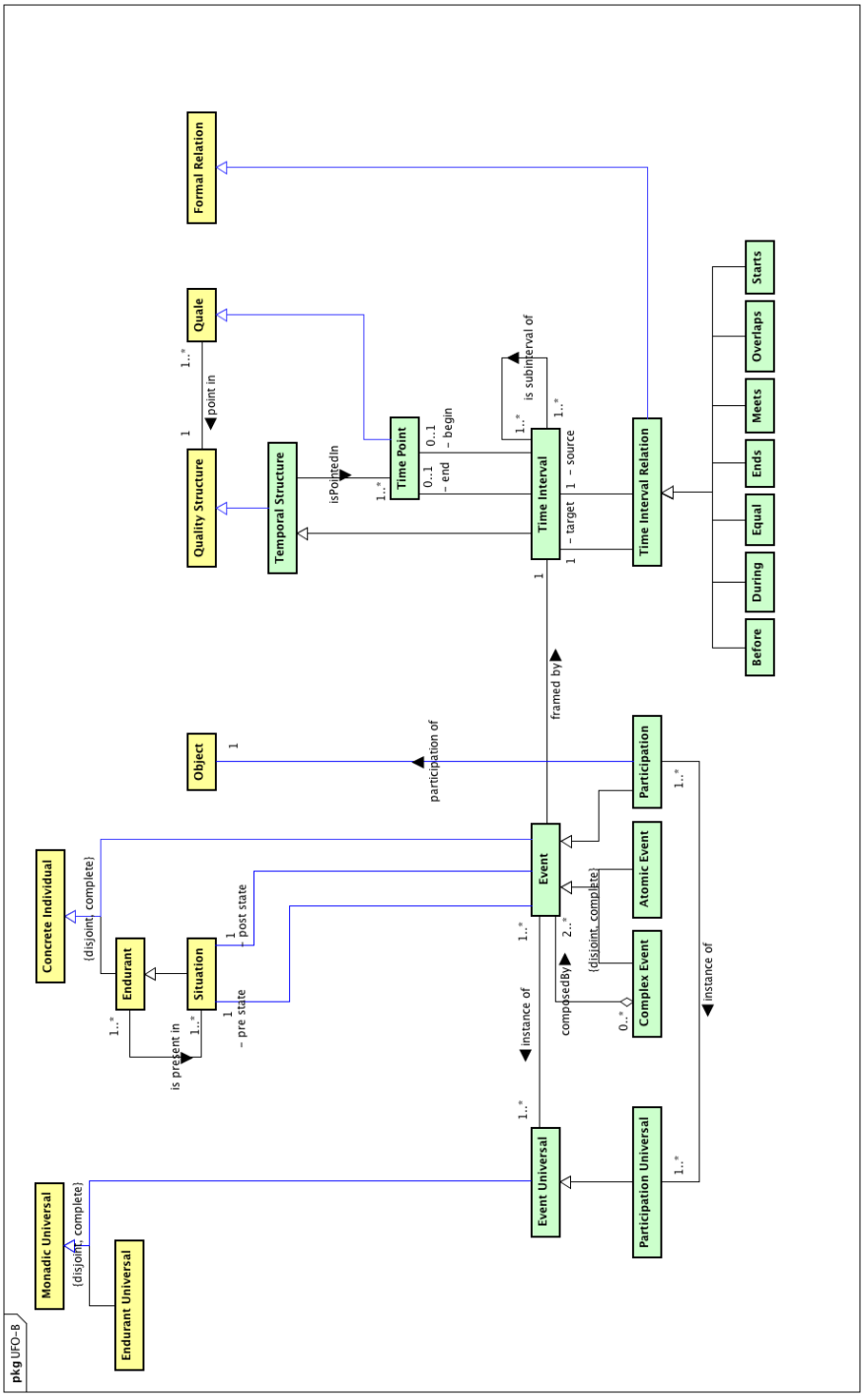


Figure B.2: UFO-B extension of UFO-A (UFO-B elements are in green colour) from [82]

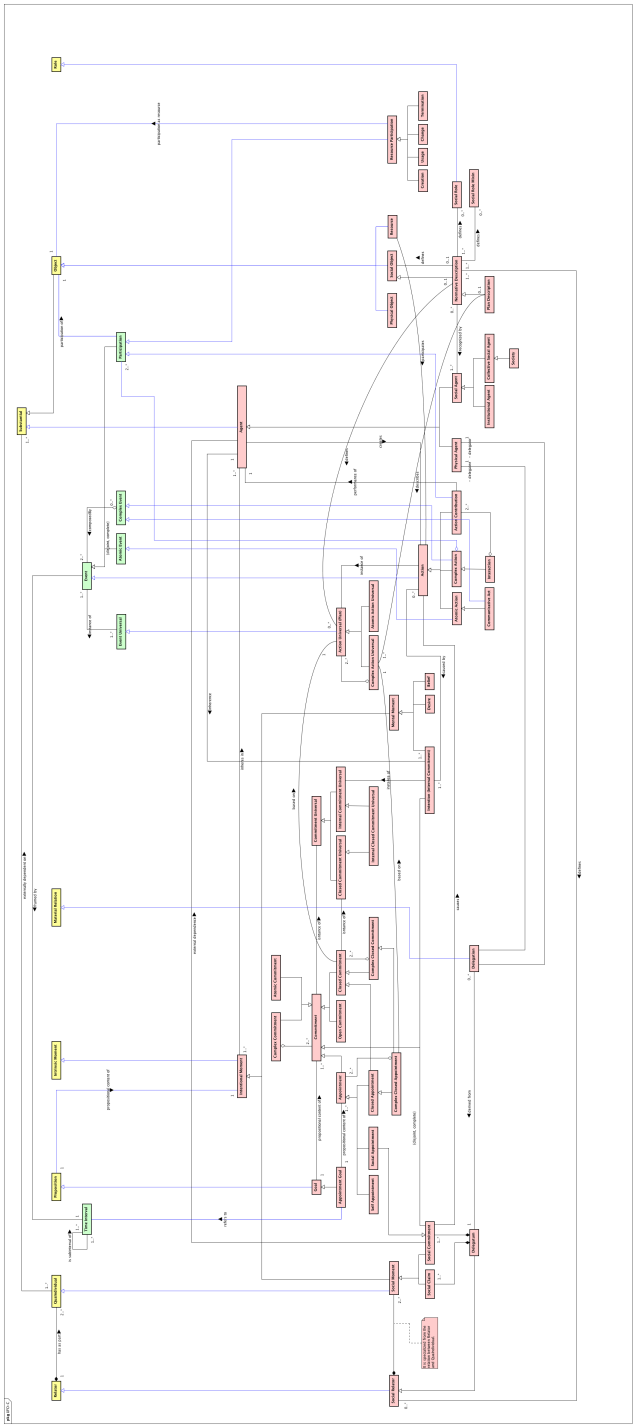


Figure B.3: UFO-C extension of UFO-A with UFO-B extension already included (UFO-C elements are in pink/red colour) form [83]

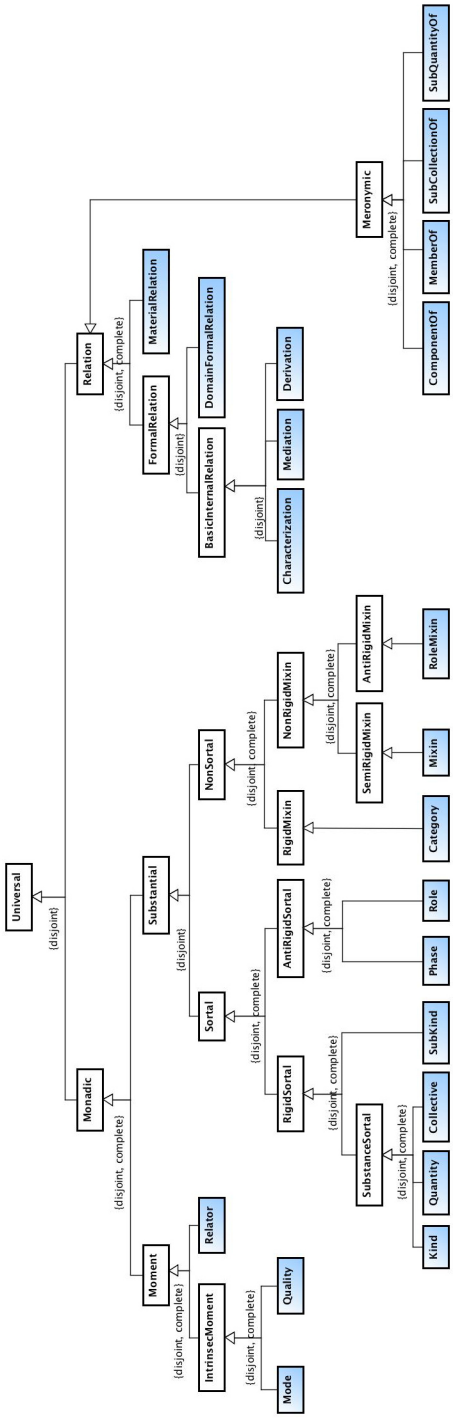


Figure B.4: Metamodel of the OntoUML language form [72]

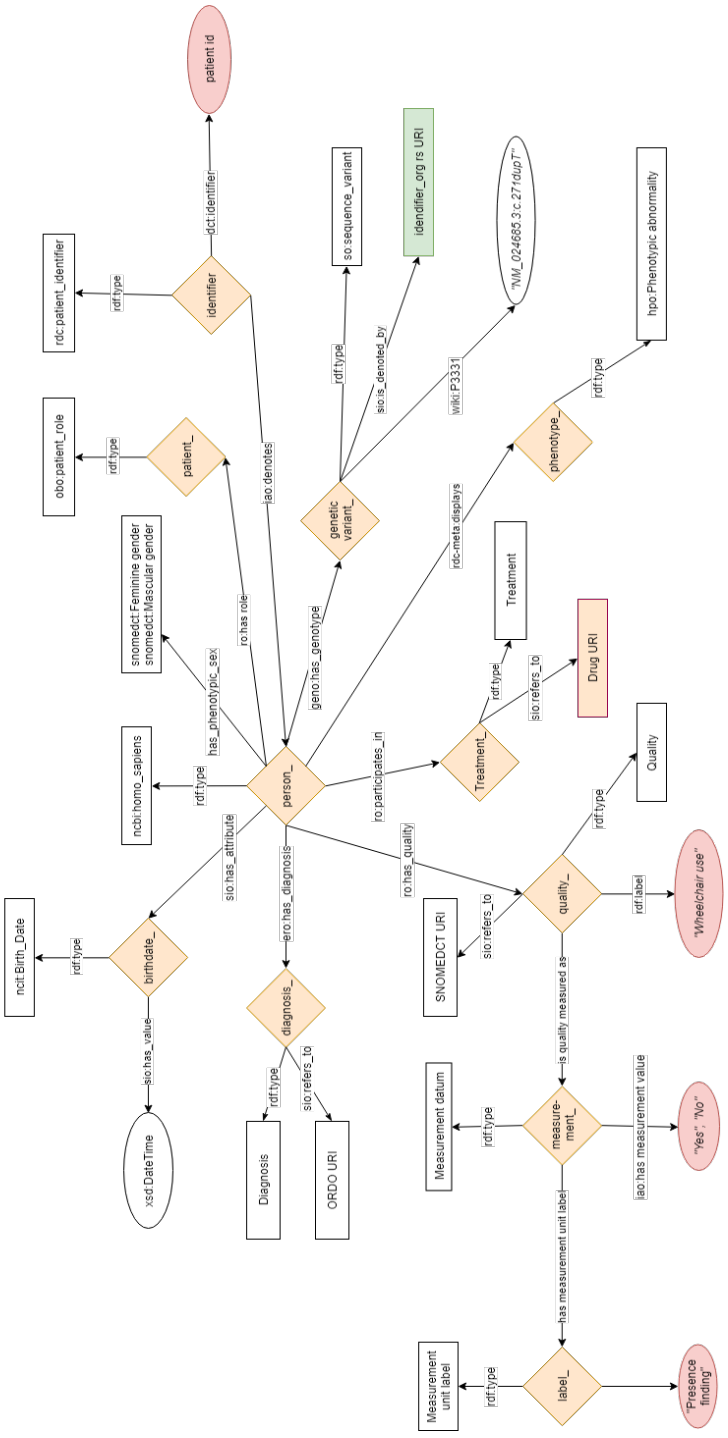


Figure B.5: Provided model in OWL in the full size

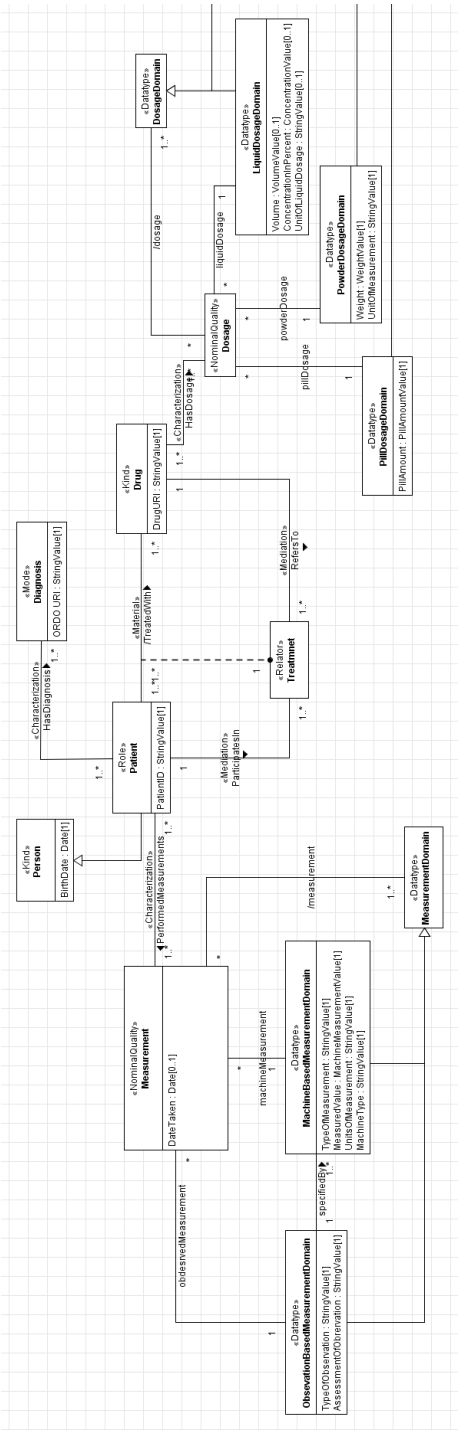


Figure B.6: Fourth iteration of the model in the full size

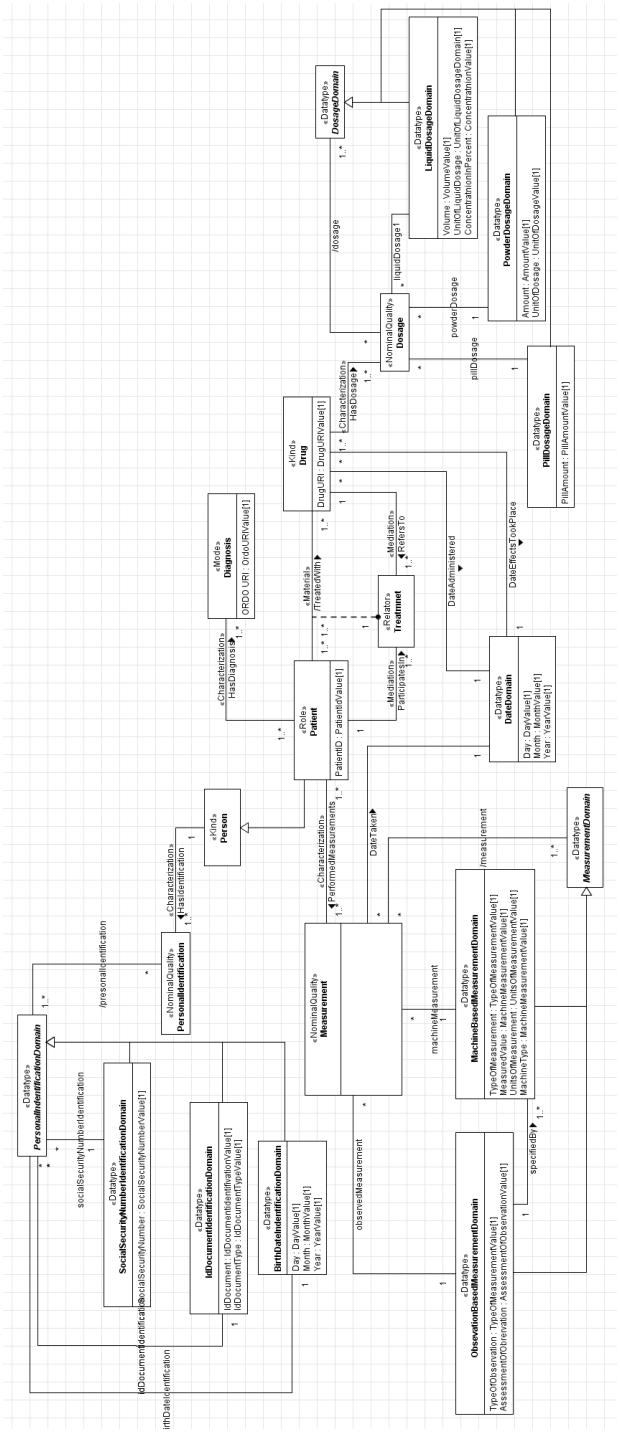


Figure B.7: Sixth iteration of the model in the full size

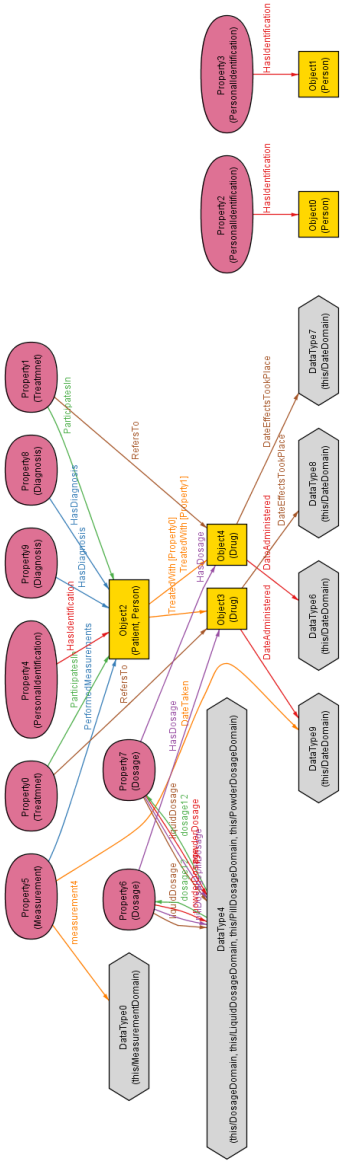


Figure B.8: Invalid Alloy simulation of the model prior fixing all the issues in full size

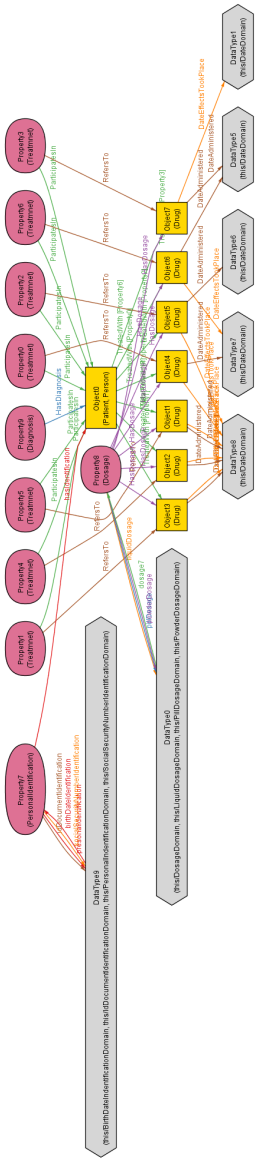


Figure B.9: Valid simulation of the model in figure B.10 in full size

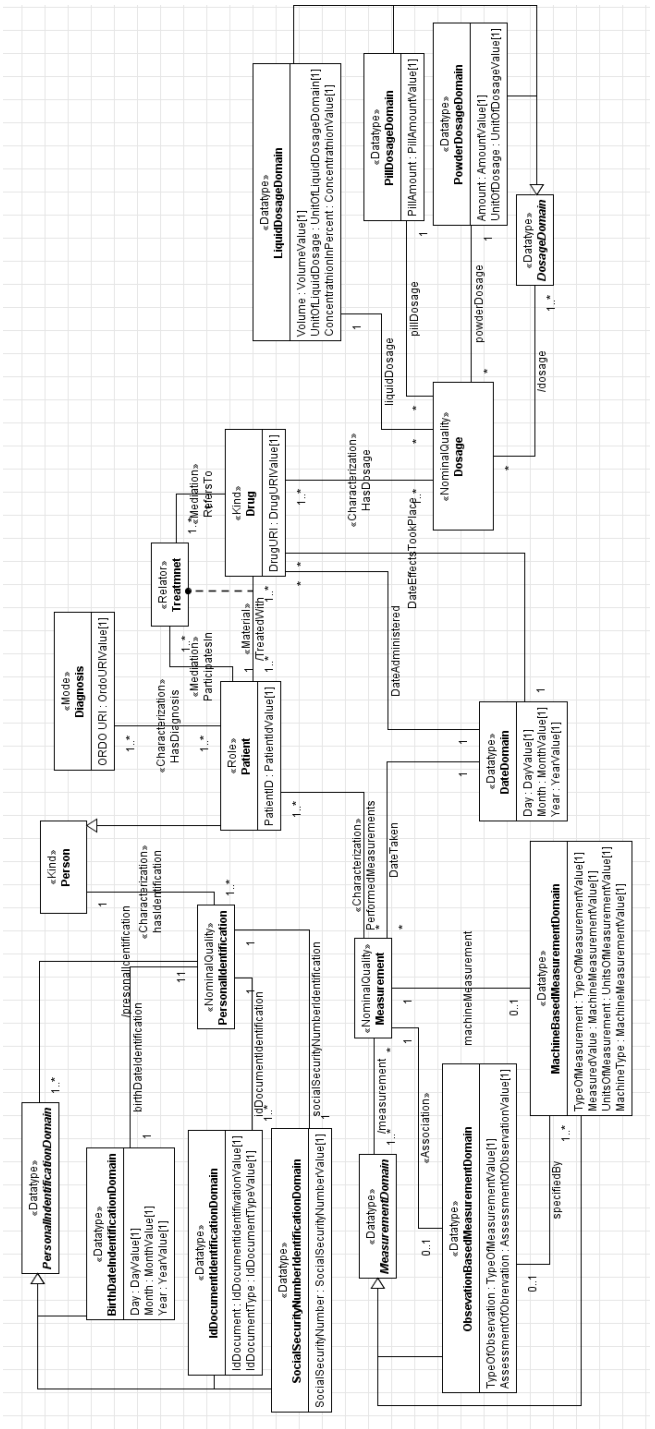


Figure B.10: Final version of the model, after all the issues and faults were fixed in full size

Appendix C

Contents of enclosed CD

- `readme.txt` ..... The file with CD contents description
- `src` ..... The directory of source codes
  - `menthor` ..... The directory of Menthor sources
    - `exports` ..... The directory with exports of the model
  - `thesis` ..... The directory of LaTeX source codes of the thesis
- `text` ..... The directory with the *PDF* files
  - `DP_Bicek_Lukas_2019.pdf` ..... the thesis text in PDF format
  - `research` ..... The folder with literature for the thesis
    - `OntoUML course` ..... NEMO course used in the thesis