NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

TIAGO MIGUEL MOREIRA BOTELHO

BSc in Computer Science

AUTOMATICALLY SELECTING PATIENTS FOR CLINICAL TRIALS WITH JUSTIFICATIONS

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
September, 2022

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

AUTOMATICALLY SELECTING PATIENTS FOR CLINICAL TRIALS WITH JUSTIFICATIONS

TIAGO MIGUEL MOREIRA BOTELHO

BSc in Computer Science

Adviser: Jörg Matthias Knorr
Assistant Professor, NOVA University Lisbon

Co-adviser: João Leite
Associate Professor, NOVA University Lisbon

Examination Committee

Chair: Nuno Manuel Robalo Correia
Full Professor, NOVA University Lisbon

Rapporteur: Eduardo Fermé
Full Professor, University of Madeira

Member: Jörg Matthias Knorr
Assistant Professor, NOVA University Lisbon

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
September, 2022

Automatically selecting patients for clinical trials with justifications

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.9.10) [32].

Acknowledgements

I would like to thank all those who have, directly or indirectly, helped and supported the development and conclusion of this dissertation.

Firstly, I would like to start by expressing my gratitude to my advisers, Professor Dr. Matthias Knorr, and Professor Dr. João Leite for all the support and guidance throughout the elaboration of this dissertation. It was a pleasure to be taught by you over the last year.

I want to thank my family for all of their support, love and encouragement shown during my academic path. Reaching this point would not be possible if it was not for them, specially my mother and father. You always believed in me and always made every effort in the world to give me everything I wanted. Without you I would not be who I am today. I would also like to thank my brother who did not let me sleep late. But above all, even though we get angry sometimes, I know that I can always count on you. You will never fail me. Thank you for everything. I will always be here for whatever you need.

Furthermore, I want to thank all my friends for motivating me and giving me helpful feedback. But I want to especially thank Pedro Maia, Tiago Cotovio, and Gonçalo Mateus. You have been my partners during these five years and helped me whenever I needed it. Without you, this journey would never be the same. Also, thanks to my great friend Rodrigo Cardoso. We have known each other for almost ten years and whenever I needed a friendly word or some help, you were always available and ready to help me. I wish you all lots of success, both personal and professional.

Last, but definitely not least, a big thanks to my girlfriend Sara Reis, who gave me motivation when I needed it, got upset with me when I was not working, for the surprises and for all the help to review my final document. Your support was tireless.

My heartfelt gratitude goes to all of you.

"Automation does not need to be our enemy. I think machines can make life easier for men, if men do not let the machines dominate them." (John F. Kennedy)

Abstract

Clinical trials are human research studies that are used to evaluate the effectiveness of a surgical, medical, or behavioral intervention. They have been widely used by researchers to determine whether a new treatment, such as a new medication, is safe and effective in humans. A clinical trial is frequently performed to determine whether a new treatment is more successful than the current treatment or has less harmful side effects.

However, clinical trials have a high failure rate. One method applied is to find patients based on patient records. Unfortunately, this is a difficult process. This is because this process is typically performed manually, making it time-consuming and error-prone. Consequently, clinical trial deadlines are often missed, and studies do not move forward. Time can be a determining factor for success. Therefore, it would be advantageous to have automatic support in this process. Since it is also important to be able to validate whether the patients were selected correctly for the trial, avoiding eventual health problems, it would be important to have a mechanism to present justifications for the selected patients.

In this dissertation, we present one possible solution to solve the problem of patient selection for clinical trials. We developed the necessary algorithms and created a simple and intuitive web application that features the selection of patients for clinical trials automatically. This was achieved by combining knowledge expressed in different formalisms. We integrated medical knowledge using ontologies, with criteria that were expressed using nonmonotonic rules. To address the validation procedure automatically, we developed a mechanism that generates the justifications for each selection together with the results of the patients who were selected.

In the end, it is expected that a user can easily enter a set of trial criteria, and the application will generate the results of the selected patients and their respective justifications, based on the criteria inserted, medical information and a database of patient information.

Keywords: clinical trials, description logics ontologies, nonmonotonic rules, hybrid knowledge, explanations, justifications

Resumo

Os ensaios clínicos são estudos de pesquisa em humanos, utilizados para avaliar a eficácia de uma intervenção cirúrgica, médica ou comportamental. Estes estudos, têm sido amplamente utilizados pelos investigadores para determinar se um novo tratamento, como é o caso de um novo medicamento, é seguro e eficaz em humanos. Um ensaio clínico é realizado frequentemente, para determinar se um novo tratamento tem mais sucesso do que o tratamento atual ou se tem menos efeitos colaterais prejudiciais.

No entanto, os ensaios clínicos têm uma taxa de insucesso alta. Um método aplicado é encontrar pacientes com base em registos. Infelizmente, este é um processo difícil. Isto deve-se ao facto deste processo ser normalmente realizado à mão, o que o torna demorado e propenso a erros. Consequentemente, o prazo dos ensaios clínicos é muitas vezes ultrapassado e os estudos acabam por não avançar. O tempo pode ser por vezes um fator determinante para o sucesso. Seria então vantajoso ter algum apoio automático neste processo. Visto que também seria importante validar se os pacientes foram selecionados corretamente para o ensaio, evitando até eventuais problemas de saúde, seria importante ter um mecanismo que apresente justificações para os pacientes selecionados.

Nesta dissertação, apresentamos uma possível solução para resolver o problema da seleção de pacientes para ensaios clínicos, através da criação de uma aplicação web, intuitiva e fácil de utilizar, que apresenta a seleção de pacientes para ensaios clínicos de forma automática. Isto foi alcançado através da combinação de conhecimento expresso em diferentes formalismos. Integrámos o conhecimento médico usando ontologias, com os critérios que serão expressos usando regras não monotónicas. Para tratar do processo de validação, desenvolvemos um mecanismo que gera justificações para cada seleção juntamente com os resultados dos pacientes selecionados.

No final, é esperado que o utilizador consiga inserir facilmente um conjunto de critérios de seleção, e a aplicação irá gerar os resultados dos pacientes selecionados e as respetivas justificações, com base nos critérios inseridos, informações médicas e uma base de dados com informações dos pacientes.

Palavras-chave: ensaios clínicos, ontologias de lógicas de descrição, regras não monotónicas, conhecimento híbrido, explicações, justificações

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence *(p. 2)*
API Application Programming Interface *(p. 25)*

CWA Closed World Assumption *(p. 3)*

DLs Description Logics *(p. 2)*

EHR Electronic Health Record *(p. 22)*
ERD Entity Relationship Diagram *(p. 11)*

FOL First-order Logic *(p. 15)*

GDPR General Data Protection Regulation *(p. 24)*

HST Hitting Set Tree *(p. 25)*

ICD International Classification of Diseases *(p. 22)*
ICPC International Classification of Primary Care *(p. 23)*
IDE Integrated Development Environment *(p. 53)*
IRI Internationalized Resource Identifier *(p. 42)*

KRR Knowledge Representation and Reasoning *(p. 2)*

LOINC Logical Observation Identifiers Names and Codes *(p. 23)*
LP Logic Programming *(p. 14)*

MKNF Minimal Knowledge and Negation as Failure *(p. 4)*

NIH National Institutes of Health *(p. 1)*

NIST National Institute of Standards and Technology *(p. 22)*
NLP Natural Language Processing *(p. 22)*
NoHR Nova Hybrid Reasoner *(p. 4)*

OBDA Ontology-Based Data Access *(p. 18)*
OBDC Open Database Connectivity *(p. 18)*
OWA Open World Assumption *(p. 3)*
OWL Ontology Web Language *(p. 2)*

RDF Resource Description Framework *(p. 9)*
RIF Rule Interchange Format *(p. 9)*

TREC Text Retrieval Conference *(p. 22)*

UML Unified Modeling Language *(p. 11)*
UNA Unique Name Assumption *(p. 15)*

W3C World Wide Web Consortium *(p. 9)*

XML Extensible Markup Language *(p. 15)*

1

Introduction

This initial chapter, presents an overview of the main problem that this dissertation aims to address, providing its context and discusses the motivation behind it. Also, it presents the main contributions and the skeleton of the remaining document.

1.1 Context and Motivation

The main purpose of a clinical trial is research, using human volunteers, that is intended to increase medical knowledge related to the treatment, diagnosis and prevention of diseases or conditions [12]. Clinical trials are the primary way that researchers use to find out if a new treatment is safe and effective in people. It is frequently conducted to determine whether a new treatment is more effective than the current treatment and/or has less harmful side effects than the standard treatment.

Since clinical trials include human subjects, these research adhere to strong scientific norms and standards in order to protect participants and offer accurate and reliable data [42]. Every clinical trial is overseen by a lead investigator, who is usually a medical doctor. There is also a research team for clinical studies that may comprise doctors, nurses, social workers, and other health-care experts. Medical researchers analyze medical records of the available patients and check if a patient satisfies the trial's criteria requirements or if there is anything that would exclude him from the trial.

A substantial number of clinical trials fail to recruit their target patients. According to the National Institutes of Health (NIH)[1], 80 percent of clinical trials miss their patient recruitment deadlines, and many fail to enroll the minimal number of patients needed to power the study as planned [8]. This is due primarily to the fact that, despite the existence of the public portal ClinicalTrials.gov, which allows new candidates to apply for new studies if they meet the requirements, there are few candidates for the majority of the studies. Consequently, it is necessary to utilize clinical data from other institutions, such as hospitals. In addition, finding suitable candidates for clinical trials based on patient records is a labor-intensive and error-prone process that requires consideration

[1] https://www.nih.gov/

of a huge amount of different data, including the patient's medical history, treatment, medications, and medical test results. This has been a manual process [12], hence errors may occur.

In the last years, several attempts have been made to improve efficiency and effectiveness in the selection of patients for clinical trial [35]. Therefore, to select patients for clinical trials without making mistakes and in a more effective manner, we need to use knowledge expressed in different formats. With regard to representing knowledge, an area that tries to represent knowledge about the world or a topic of interest using logic-based formalisms, and that allows automated systems to reason with this knowledge and derive conclusions from it is Knowledge Representation and Reasoning (KRR), which is an area in Artificial Intelligence (AI).

Knowledge can be expressed in many different ways. In order to choose patients for clinical trials, it is necessary to represent medical knowledge. To be selected, a patient must meet the selection criteria. Criteria can be considered of two types: inclusion criteria and exclusion criteria. For example, age, stage of disease, sex, genetic profile, family history, and whether or not you have a study partner who can accompany you to subsequent appointments are all possible inclusion criteria for a trial. In contrast, specific health disorders or drugs that potentially interfere with the treatment being investigated could be used as exclusion criteria [45].

Taking all of this into account, it is vital to have stored the information of various patients, as well as ways to represent the various criteria and existing medical knowledge and information, such as treatments, diseases, and medicines. For example, patient data could be stored in a relational database with patient's information. To represent medical information, it is necessary to find a way to represent that information and its relationships.

Ontology languages are used to represent existing medical information. Ontology languages are widely used to represent rich and complex knowledge about things, groups of things, and relations between things [25]. Ontology Web Language (OWL)[2], in particular, is a semantic web language founded on Description Logics (DLs) [34], which are commonly decidable fragments of first-order logic. It also allows us to reason about abstract relationships between classes of objects without having to refer to actual instances or objects, as well as to reason with unknown individuals, or objects that are inferred to exist while not matching any known object in the represented knowledge. A simple example is: if there is a disease of type A and a disease of type B, both could be type Z diseases, and in a way they are related. This relationship is not directly demonstrated and sometimes the names are different, which could lead to the incorrect exclusion of a patient. With a language that allows representing this type of knowledge, everything becomes simpler. Therefore, an ontology language fits perfect to represent medical knowledge.

On the other hand, as mentioned previously, criteria may be inclusive or exclusive.

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

A good way to represent the criteria, is by using nonmonotonic rules. There are a wide range of rule languages - declarative languages, in which inferences from premises to conclusions are represented on the level of concrete instances. Nonmonotonic rules, as defined in Logic Programming, are of particular importance in these languages [25]. The word 'nonmonotonic' refers to the possibility of revising these findings in the event of new knowledge.

As described, it is not enough to use only ontologies or just rules. In fact, we need to use both at the same time. [48, 29], are examples of applications that frequently require the integration of rules and ontologies. For this dissertation, we need to be able to conduct this integration in order to choose patients for clinical trials where we have ontologies for medical knowledge, relational databases with patient data, and rules, such as the clinical trial criteria.

Many studies have been carried out within the scope of this integration in different areas. A broad case study was conducted in [36] in order to investigate the application of ontology reasoning in the field of medicine. This study emphasizes even further that integrating Open World Assumption (OWA) with Closed World Assumption (CWA) is a key to improve. OWA states that no conclusions can be drawn just because certain knowledge is missing, and CWA is used in nonmonotonic rules, which states that if something cannot be inferred to be true, it is presumed to be false. One particular example given by the authors for this integration is: i) we need open world reasoning in radiology and laboratory data in the clinical domain, because we can't make arbitrary assumptions about the outcomes unless the lab test declares a negative finding; ii) however, in pharmacy data, we need closed world assumption, because we can conclude that a patient is not on a prescription if it is not stated. The results of this study suggest that employing ontology matching to automate clinical trial matching, which is currently a manual process, is a viable option. Also, some of the existing problems were solved - dealing with incomplete patient information; identifying and eliminating noise in the patient data; mapping a different terminology to a standard terminology, known as SNOMED CT[3].

In the clinical domain, SNOMED CT is essential. This is a systematically organized computer processable collection of medical terms providing codes, terms, synonyms and definitions used in clinical documentation and reporting. It supports the development of comprehensive high-quality clinical content in electronic health records and it is considered to be the most comprehensive, multilingual clinical healthcare terminology in the world. As a result, SNOMED CT has been adopted as a national health-care standard in the United States.

Scalability was referred as a problem for large ontologies [36]. So, the selection of patients can be rather slow and a time-consuming process for the integration of large ontologies, with a large amount of data and the trial criteria.

Another problem is related to justifications and understandability. The generation of

[3]https://www.snomed.org/

explanations, or justifications, for inferences computed by a reasoner is a very required feature [19]. Despite the improvements made in the previous study, the process of selecting patients for clinical trials provides no explanations as to why they were selected. In this sense, displaying the justification of why a certain patient has been chosen, can be a crucial feature that not only gives information about why an entailment is true but also, reduces the risk of errors and helps to detect any problems in advance. This enhances the credibility of our application because, instead of presenting only the selected patients, we provide justifications. This makes it easier to check for potential errors or inconsistencies and to confirm that the selected patients meet the criteria based on the justification provided.

In conclusion, there are some questions that arise:

- How can we create explanations for why a patient was selected?
- How can we create a simple and easy to use application to select patients for clinical trials?

These questions will be answered throughout this dissertation with the development of a viable solution.

1.2 Solution and Contributions

Whenever a problem arises, regardless of its nature, there is always someone seeking for a solution. When the need emerged to combine OWA with CWA, namely the integration of rules with ontologies, several tools were created to help with this matter, such as: DReW [47], HD Rules [9] and Nova Hybrid Reasoner (NoHR)[4] [20]. A prominent tool in this regard is NoHR. While both DRew and HD Rules are based on different formalisms to combine nonmonotonic rules and ontologies, and both formalisms differ from Minimal Knowledge and Negation as Failure (MKNF) knowledge bases [17], NoHR is a tool that allows its users to integrate and efficiently reason about OWL and nonmonotonic rules, the two standard languages for the Semantic Web, where the approach uses the well-founded semantics for MKNF knowledge bases. Moreover, NoHR's overall performance is better compared to the other two, allowing efficient querying of combinations of DL ontologies and nonmonotonic rules. This tool was developed as a plugin for Protégé[5], to allow the integration of ontologies and rules. In this sense, it is quite useful, despite not having any mechanism that presents explanations. This is also where the usability issue arises.

This tool requires a set-up that can be difficult, especially for individuals who are unfamiliar with programming and technologies. Taking into consideration the benefits

[4] https://nohr.di.fct.unl.pt/
[5] https://protege.stanford.edu/

of NoHR, this dissertation outlines the creation of an application that overall solves this problem. As was mentioned, a clinical trial needs to process different types of data hence, integrating different formats of knowledge is a requirement.

To answer the questions raised in the previous section in a more concise manner, the development of required code was divided into two major components. The first is the back-end component, where all of the code required to select patients for clinical trials based on the data we have and the criteria entered was implemented. This was accomplished through the integration of NoHR, which, as previously stated, is a tool that allows for the efficient combination of ontologies (i.e., SNOMED-CT) and nonmonotonic rules (i.e., criteria) and, run queries on the resulting hybrid knowledge base. Furthermore, the algorithm that generates the justifications for each selected patient was implemented in the back-end. Initially, because we could not use a real database due to safety concerns, it was necessary to create a new database containing patient information that was similar to a real database so that we could simulate clinical trials. This was accomplished by utilizing Synthea™[6], a generator of synthetic data that is extremely similar to real data. Thus, having all the data necessary to simulate clinical trials, we were able to generate results from which patients were selected based on the criteria that were entered.

However, with this component alone, it would not be possible for any user to simulate clinical trials without programming knowledge. The usability problem persisted since NoHR requires a set-up that can be a quite complicated and the back-end requires some extra knowledge. Also, the way rules are written are a little complex. To solve this problem, it was necessary to develop a web application that was connected to our back-end to obtain the results and then display them to the user.

With the work carried out throughout this dissertation, it was always intended that anyone could enter the selection criteria easily without the need for any prior programming knowledge. The web application was created from scratch and allows an efficient, and effective selection of patients for clinical trials. This app, as mentioned before, was built with the goal of not only presenting the results of which patients were chosen, but also the respective justifications for each selection. Furthermore, we always kept in mind it was essential that the application should be intuitive and easy to use, regardless the user's technological knowledge, this is because it is not expected that, for example, a doctor, has any programming knowledge.

In summary, the major contribution to these problems was the creation of an intuitive and simple web application, to select patients for clinical trials, due to the fact that it aims to be widely used, without any programming knowledge required. In the end, this application contains the integration of different types of data. Namely, patient's information, ontologies and rules, by using NoHR, where the patients selected for each clinical trial will be presented in the result. Along with the result, explanations for their selection are also provided so that validating the results is easier and less time consuming

[6]https://github.com/synthetichealth/synthea

than manually searching the database and the ontology for everything. When compared to the time it takes to manually select patients, the application performance results are quite encouraging. It is also a plus that the application can be easily extended to accept more criteria, as it does not require major changes to the application or the creation of a new app.

1.3 Document Structure

The rest of this document is structured as follows:

- **Chapter 2 - Background:** summarizes the key knowledge required to comprehend the subsequent chapters in this dissertation. Firstly, it is explained what ontologies are, going into some details about OWL and the three different profiles (*EL*, *QL* and *RL*). Within this context, description logics are also introduced. Later it is explained what are rules and the difference between monotonic and nonmonotonic rules. That said, the idea of integration between ontologies and rules is presented. This chapter also features the Protégé framework and NoHR is covered in more detail.

- **Chapter 3 - Related Work:** addresses some of the studies carried out that are related to our dissertation. In particular, we begin by presenting some studies in the medical domain, more specifically clinical trials and patient selection. Subsequently, some studies with regard to justifications and what are the different ways to present them. To conclude we finish the chapter by mentioning what will be the possible option for us to to incorporate in the development of our application.

- **Chapter 4 - Patient Selection:** starts by providing an overview which conceptually describes the architecture of the entire back-end, followed by a detailed description of the implementation of all necessary components developed to achieve the objective of automatically selecting patients with the proper explanations for each selection.

- **Chapter 5 - Justifications:** provides a more detailed description of how the justifications were created and how they are presented to the user with some examples. To make them more readable, some improvements and adaptations that were developed are also described.

- **Chapter 6 - Web app:** explains the approach and technologies utilized to develop the front-end application in general. Despite being considered simple and easy to use without any prior knowledge in the domain, the user manual and interface are presented so that all the features of the application are understood.

- **Chapter 7 - Evaluation:** outlines the results, time comparisons between the SNOMED-CT ontology and a smaller version of SNOMED-CT ontology, and the potential effects on computation time when using a higher dimensional database.
- **Chapter 8 - Conclusion:** concludes the dissertation, providing the main contributions and thoughts for making improvements for future work.

2

Background

The knowledge needed to grasp the relevant theoretical terminology and foundations is presented in this chapter. It is also presented in greater detail, the main tool needed for this dissertation.

2.1 Ontologies

An Ontology is a representation scheme that describes a formal conceptualization of a domain of interest. Typically, an ontology consists of a limited list of terms and the relationships between those terms. These terms represent important concepts (object classes) of the domain. For example, in a university, students, professors, staff and courses are some of the important concepts.

Ontologies can include a hierarchy of classes. This means, for example, if there are two classes A and B, and all objects of A also belong to B, we can conclude that A is a sub-class of B.

In addition to class hierarchies, ontologies include information such as:

- **Properties:** one example of possible property is "Person buys Ticket";
- **Disjunction between two classes:** sometimes it is necessary to mention that two different classes are disjoint, such as "Pizza and Hamburger are disjoint classes";
- **Relations between two classes:** for example "a Team must have at least 10 Players";
- **Value restrictions:** used when we want to restrict something, such as "only Players can participate in the Tournament".

Ontology languages are useful for defining information that adheres to the OWA, which states that encoded knowledge is incomplete and that conclusions that cannot be deduced from an ontology are treated as agnostic [10].

Ontologies can be quite complex, like Snomed CT, which are healthcare ontologies that contain thousands of terms.

The role of ontologies in the semantic web is to facilitate the integration of information, for example, when there are ambiguities with terms used to define different sets of information, or when an extra piece of knowledge leads to the discovery of new relationships. Considering the example of ontologies in the field of health care, they are used to represent knowledge about treatments, symptoms and diseases.

The complexity of the ontologies intended to use depend on the applications, in which some may decide to use large and complex ontologies, and it depends on the application logic. Other applications may prefer to use simpler ontologies, or it ends up not being necessary to use such a complex ontology.

To meet these needs, the World Wide Web Consortium (W3C)[7] offers a wide range of standardized languages to describe and define different forms of ontologies in a standard format. These include OWL [24], Resource Description Framework (RDF)[8] and RDF Schema, and Rule Interchange Format (RIF)[9], although the RIF defines a standard that allows the exchange of rules between rule systems.

2.1.1 RDF and RDF Schema

Based on XML syntax, RDF is used to represent information on directed, labelled and typed graphs. RDF is a framework for expressing information about resources. Documents, people, physical objects, and abstract concepts are all examples of resources.

RDF is designed for cases where web information needs to be processed by applications rather than only displayed to consumer. RDF provides a standardized framework for representing this data so that it may be shared between applications without losing its meaning. Application designers can take use of the availability of standard RDF parsers and processing tools because it is a common framework. Since information may be exchanged between other apps, it can be made available to applications other than those for which it was initially produced [15]. Essentialy, RDF is made up of resources, declarations, properties, and graphs [41].

Following is an informal example of RDF Schema triples adapted from [41]:

$< Person >$ **<type>** $< Class >$
$< is\ a\ friend\ of >$ **<type>** $< Property >$
$< is\ a\ friend\ of >$ **<domain>** $< Person >$
$< is\ a\ friend\ of >$ **<range>** $< Person >$
$< is\ a\ good\ friend\ of >$ **<subPropertyOf>** $< is\ a\ friend\ of >$

[7] https://www.w3.org/
[8] https://www.w3.org/RDF/
[9] https://www.w3.org/TR/rif-overview/

A more specific example also adapted from [41] is shown in the following:

< *Bob* > **<is a>** < *person* >
< *Bob* > **<is a friend of>** < *Alice* >
< *Bob* > **<is born on>** < *the 4th of July* 1990 >
< *Bob* > **<is interested in>** < *the Mona Lisa* >
< *the Mona Lisa* > **<was created by>** < *Leonardo da Vinci* >
< *the video* 'La Joconde à Washington'> **<is about>** < *the Mona Lisa* >

Triples can be seen as a connected graph. Graphs consists of nodes and arcs. The nodes in the graph are the triples' subjects and objects, whereas the arcs are the predicates. Figure 2.1, whose source was [41], represents the graph produced by the sample triples.

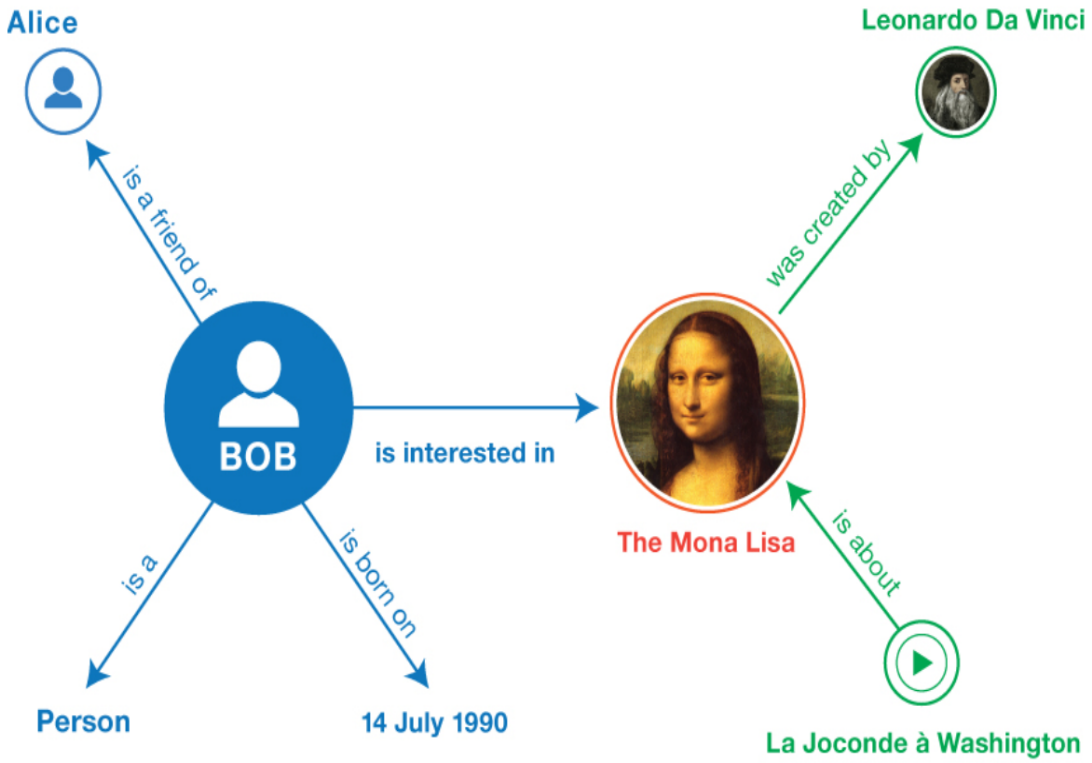

Figure 2.1: Graph example of sample triples

2.1.2 OWL 2

The OWL has been standardised by the W3C as a powerful knowledge representation language for the Web. OWL is not limited to the Web, and it has been effectively used for knowledge modeling in a variety of applications.

Modeling information in OWL has two practical advantages: as a descriptive language, it can be used to convey expert knowledge in a formal way, and as a logical language it

can be used to draw inferences from this knowledge; the second component is what sets OWL apart from other modeling languages like Unified Modeling Language (UML).

Computing all interesting logical conclusions of an OWL ontology, on the other hand, can be a difficult job, and reasoning is often exponential. OWL 1 established two major languages, OWL DL and OWL Full, as well as a syntactic subset known as OWL Lite. However, it turned out that this was not sufficient [14]. In order to address the limitations OWL 2 defines three different profiles[10].

OWL 2 is an extension and revision of the OWL 1, and like OWL 1, it is designed to facilitate ontology development and sharing via the Web, with the ultimate goal of making Web content more accessible to machines. OWL 2 increased expressive power for properties, extended support for datatypes and some other new features. This has resulted in significant efficiency and scalability gains, making the OWL 2 profiles highly appealing to practitioners [28, 33]:

- **OWL 2 EL:** is used in huge biomedical ontologies. This profile is especially beneficial in applications that use ontologies with a high number of attributes and/or classes. It captures the expressive capability employed by many such ontologies for which the core reasoning issues may be solved in polynomial time as the ontology grows in size. There are dedicated reasoning methods for this profile that have been shown to be implementable in a highly scalable manner. The EL abbreviation refers to the EL family of description logics(EL++), and the "E" referes to Existential quantification. OWL 2 EL is used in huge biomedical ontologies [28].
- **OWL 2 QL:** provides database applications with an ontological data access layer. This profile is designed for applications that employ huge amounts of instance data and where the most significant reasoning activity is query answering. Conjunctive query answering in OWL 2 QL can be implemented utilizing relational database systems. With respect to the quantity of the data, sound and complete conjunctive query answering can be achieved in logarithmic space. Polynomial time methods can be employed to solve the ontology consistency and class expression subsumption reasoning difficulties, just like in OWL 2 EL. Although it includes most of the primary aspects of conceptual models such as UML class diagrams and Entity Relationship Diagram (ERD), the expressive power of this profile is restricted. The abbreviation QL refers to the fact that under this profile, query answering is accomplished by rewriting queries into a standard relational Query Language.
- **OWL 2 RL:** became the preferred approach for reasoning with Web data. This profile is intended for applications that require scalable reasoning without compromising expressive capability. It is intended for OWL 2 applications that may trade full expressivity for efficiency, as well as RDFs applications that require some additional expressivity. Rule-based reasoning engines can be used to create OWL 2 RL reasoning systems. The RL abbreviation refers to the fact that this profile's

[10] https://www.w3.org/TR/owl2-profiles/

reasoning can be implemented using a standard Rule Language.

In each case, the most important computational tasks are tractable, this means that they can be performed in polynomial time.

2.2 Description Logics

Previously, we mentioned DLs as fragments of first-order logic whose reasoning is generally decidable. They are logics specifically designed to represent and reason on structured knowledge. To affirm knowledge, assertions are used, also known as logical axioms. Many DLs are more expressive than propositional logic but less expressive than first-order logic.

More specifically, DLs are defined more explicitly over disjoint countably in finite sets of [25]:

- **Concept Names N_C:** corresponding to classes of individuals (unary);
- **Role Names N_R:** corresponding to relationships (binary);
- **Individual Names N_I:** corresponding to objects (constants).

A Description Logics ontology or also, knowledge base, is a pair $\mathcal{O} = \langle \mathcal{A}, \mathcal{T} \rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox [4]:

- **A DLs TBox consists of a set of assertions on concepts and roles:**
  - Inclusion assertions on concepts: $C_1 \sqsubseteq C_2$
  - Inclusion assertions on roles: $R_1 \sqsubseteq R_2$
  - Property assertions on atomic roles: *transitive P; functional P; symmetric P; reflexive P; domain P C; range P C; ...*
- **A DLs ABox consists of a set of assertions on individuals:**
  - Membership assertions for concepts: $A(c)$
  - Membership assertions for roles: $P(c_1, c_2)$
  - Equality and distinctness assertions: $c_1 \approx c_2$; $c_1 \not\approx c_2$

It should be observed that the admitted constructors, via which complex concepts and roles are inductively generated, can distinguish each DL, that is, each fragment of first-order logic. Table 2.1 shows an overview of the admitted constructors in the DL underpinning the OWL 2 language, it denotes the description logic $\mathcal{SROIQ}$.

Name	Syntax	Semantics
inverse role	R^-	$\{(x,y) \in \Delta^I \times \Delta^I \mid (y,x) \in R^I\}$
universal role	U	$\Delta^I \times \Delta^I$
top	$\top$	Δ^I
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^I \setminus C^I$
conjunction	$C \sqcap D$	$C^I \cap D^I$
disjunction	$C \sqcup D$	$C^I \cup D^I$
nominals	a	$\{ a^I \}$
universal restriction	$\forall R.C$	$\{ x \in \Delta^I \mid (x,y) \in R^I \text{ implies } y \in C^I \}$
existential restriction	$\exists R.C$	$\{ x \in \Delta^I \mid \exists y \in \Delta^I, (x,y) \in R^I \wedge y \in C^I \}$
Self concept	$\exists R.Self$	$\{ x \in \Delta^I \mid (x,x) \in R^I \}$
qualified number	$\leqslant nR.C$	$\{ x \in \Delta^I \mid \#\{(x,y) \in R^I \text{ and } y \in C^I\} \leq n \}$
restrictions	$\geqslant nR.C$	$\{ x \in \Delta^I \mid \#\{(x,y) \in R^I \text{ and } y \in C^I\} \geq n \}$

Table 2.1: (Adapted from [25]) Syntax and semantics of role and concept expressions in $\mathcal{SROIQ}$ [16] for an interpretation $\mathcal{I}$ with domain Δ^I, where C,D are concepts, R a role, and $a \in N_I$.

2.3 Rules

There is a huge assortment of rule languages, and as the name implies can be expressed as IF and THEN relations, that can describe knowledge about the world [25].

The rules have the following format:

$$A_1,...,A_n \rightarrow B$$

where A_i and B are atomic formulas which means that if $A_1,...,A_n$ are true, then B is also true.

Rule languages, which are proponents of the CWA, have been conceptualized as partners for ontologies to decrease the conservative position of the OWA. This assumption preserves a closed perspective of the world, hence the name. Everything that cannot be deduced from this type of knowledge base is assumed to be false [10].

It is important to note that description logics and logic programs are orthogonal in the sense that neither is a subset of the other. But, its obvious that some DL axioms can be translated into rules as an example in [27]:

- $A \sqsubseteq B$ becomes $A(x) \rightarrow B(x)$
- $R \sqsubseteq S$ becomes $R(x,y) \rightarrow S(x,y)$

where A and B are concepts and R and S correspond to relationships, interpreted as binary relations on objects. In [27], we can find several other examples and explanations of how the translation of DL axioms to rules is done.

One interesting example from [1], imagine we want to represent a class of happy spouses as those who are married to their best friend, in DLs it is impossible to define it, but we can easily represent this knowledge using rules:

$$married(X, Y), bestFriend(X, Y) \rightarrow happySpouse(X)$$

Rules, on the other hand, cannot generally assert:

- **class negation/complement**
- **existential quantification:** for example, that all persons have a father;
- **disjunctive/union information:** such as, whether a person is a man or a woman.

In contrast, OWL can express the complement and union of classes, as well as some types of existential quantification [1].

2.3.1 Monotonic Rules

In monotonic rules, we can only conclude something with the premises that are present in the rules without having to evaluate other rules. With these rules, the number of assertions that can be deduced is not reduced by additional knowledge, in other words, fresh information does not invalidate conclusions reached before [25, 38].

2.3.2 Nonmonotonic Rules

In Logic Programming (LP), nonmonotonic rules have been extensively investigated [25]. The answer set semantics and the well-founded semantics, both of which have efficient implementations, are the most extensively used nowadays. If certain conclusions can be disproved by adding further knowledge, this is called nonmonotonic logic [38]. In the system of nonmonotonic rules, even if all the premises are known, the rule may not be applied, since it is necessary to consider the chains of reasoning that may contravene the rule.

A nonmonotonic rule has the form:

$$H_1 \vee ... \vee H_l \leftarrow A_1 ,..., A_n, notB_1 ,..., notB_m$$

where H_i, A_j and B_k are atoms. This rule is divided into the head $H_1 \vee ... \vee H_l$ and the body A_1 ,..., A_n, $notB_1$,..., $notB_m$, where "," represents a conjuction between atoms. As a result, this rule is read as: if A_1 ,..., A_n are true but B_1 ,..., B_m are not, then at least one of H_1 to H_l must be true [25].

2.3.3 RIF

The RIF activity within the W3C aimed at developing a Web standard for exchanging rules. RIF can be seen as a collection of dialects, or a group of languages with well-defined syntax and semantics. This refers to the ability to add additional dialects if there is enough interest, and the languages are expected to share much of the syntactic and semantic. In [23] is presented the main idea behind rule exchange via RIF. That idea is that different systems will be able to semantically map their languages to and from the proper RIF dialects, allowing rule sets and data to be transmitted from one system to another if both systems can discover a dialect that they both support. The format expected for the intermediate RIF language is Extensible Markup Language (XML).

2.4 Integration of rules with ontologies

So far we have seen what ontologies and rules are, but the integration of ontologies, which employ the OWA formalism, with rules, which employ the CWA formalism, is not a trivial task at all.

Following a detailed examination of the two formalisms, DL ontologies and nonmonotonic rules, as well as their differing characteristics and benefits in terms of what kind of knowledge can be represented and reasoned with, the question of what to do if we want to use the best features of both at the same time arises. This question has been addressed in the literature, with a variety of approaches being provided[25].

From a syntactic perspective, the integration between ontologies and rules, means that a hybrid knowledge base with a TBox, an ABox and a set rules can be written.

Some results on the definition of logic-based systems integrating ontologies and rules are presented in [40]. In particular, ontologies are expressed in Description Logics and rules expressed in Datalog (a declarative logic programming language that syntactically is a subset of Prolog). Despite the great interest in this integration, some problems have emerged. We present the following goals or issues presented in [40]:

- **CWA vs. OWA:** since rules are based on CWA, while DLs are fragments of First-order Logic (FOL) and the semantics is based on OWA, hence different semantics. How do we properly integrate the OWA of DLs with the CWA of rules? Specifically, how may monotonic and nonmonotonic logical subsystems be merged from a semantic point of view?
- **Unique Name Assumption (UNA) vs. non-UNA:** Some DLs, such as OWL, are not UNA-based (recall that the UNA requires that distinct terms identify different objects). The standard semantics of Datalog rules, on the other hand, is based on the UNA [5]. How may a non-UNA-based semantics for DLs and rules be defined?
- **Decidability preservation:** In systems that combine DL knowledge bases and Datalog rules, the decidability and complexity of reasoning is critical. In general, this combination does not preserve decidability.

- **Modularity of reasoning:** Is it possible to do reasoning in DL knowledge bases that have been augmented with rules in a modular manner, with the DL component and the rule component clearly separated?

So, the previous article presents as the main issues arising when trying to combine DLs and Datalog in a single formalism are related to: i) Syntax; ii) Semantics; iii) Reasoning.

Since the integration between ontologies and rules started to be something very requested, some tools were developed as we previously mentioned. NoHR was developed as a tool for querying efficiently combinations of DL ontologies and nonmonotonic rules. Between NoHR and the other tools such as DReW and HD Rules, we chose to use NoHR in this sense, because it has a better performance overall and it uses the well-founded semantics for MKNF knowledge base.

In the section 2.6, we will go through what's been built and upgraded so far in terms of this tool in order to achieve the intended objective.

2.5 Protégé

Protégé [13] is a open-source ontology editor and framework for building intelligent systems. Protégé has a large network of academic, government, and business users who use it to create knowledge-based solutions in fields like biomedicine, e-commerce among others.

Why would someone use Protégé? The plug-in architecture of Protégé can be used to create basic and complex ontology-based applications. Protégé's output can be used with rule systems or other problem solvers to create a wide range of intelligent systems. Also, the latest OWL 2 Web Ontology Language and RDF specifications from the W3C are fully supported by Protégé.

Learn how NoHR works within Protégé rather than using only Protégé directly was very important to validate the results also in Protégé using NoHR plugin. Protégé was extremely helpful not only at the beginning of our application to ensure that the simplest initial examples were correct, but also throughout the dissertation. Exploring the Protégé allowed us to recognize that it provided explanations for the inferences and aided us in developing an algorithm that also legibly presents justifications for patient selection.

2.6 NoHR Plugin

NoHR is tool and also a plug-in for the ontology editor Protégé that allows us to query knowledge bases that contain both a collection of reasoning rules and an ontology and in any of the tractable OWL 2 profiles.

We can query the resulting hybrid knowledge base by loading an ontology and a set of nonmonotonic reasoning rules into NoHR. We can take advantage of the expressive capabilities of both representation techniques in this sense.

Using a top-down reasoning approach, NoHR combines the capabilities of ELK[11] for OWL 2 EL, and a dedicated direct translation for OWL 2 QL and OWL 2 RL, respectively, as well as Konclude[12] and HermiT[13], for the combination of these profiles' characteristics, with the rule engine XSB Prolog[14] to provide fast interactive response times. ELK, Konclude and HermiT are reasoners, i.e., software able to infer logical consequences from a set of asserted facts or axioms.

2.6.1 Hybrid MKNF

As previously mentioned, NoHR allows us to query the combined knowledge base. The approach used as the formalism is the Hybrid MKNF [17]. So, before we dive into NoHR (in 2.6.2) we need to understand what Hybrid MKNF Knowledge Bases means.

Hybrid MKNF knowledge bases combine DLs with disjunctive logic programs interpreted according to Lifschitz's logic of minimal knowledge and negation as failure (MKNF) [27].

Typically, a Hybrid MKNF knowledge base $\mathcal{K} = \langle \mathcal{O}, \mathcal{P} \rangle$, where $\mathcal{O}$ is a DL Ontology, $\mathcal{P}$ is a disjunctive program that is composed of DL-safe rules of the form:

$$KH_1 \vee ... \vee KH_l \leftarrow KA_1 \text{ ,..., } KA_n, notB_1 \text{ ,..., } notB_m$$

where H_i, A_j and B_k are first-order formulas. The core idea is that "if all A_j are known to hold (in the sense of truth in all models), and all B_k are not known to hold, then one of the H_i is known to hold"(i.e., one of the H_i is true in all models) [25].

However, NoHR uses the well-founded semantics for MKNF knowledge bases and this formalism does not allow disjunction on the head [17]. The two reasons presented to use Hybrid MKNF under the well-founded semantics are: i) the overall approach provides a very general and flexible framework for combining DL ontologies and nonmonotonic rules; ii) has a lower polynomial data complexity and is amenable for applying top-down query procedures, to answer queries based only on the information relevant for the query, and without computing the entire model [17]. This is a critical feature when dealing with large ontologies such as SNOMED CT.

Therefore, a rule has the following form:

$$H \leftarrow A_1 \text{ ,..., } A_n, notB_1 \text{ ,..., } notB_m$$

where the head of the rule, H, and all A_i with $1 \leqslant \mathrm{i} \leqslant \mathrm{n}$ and B_j with $1 \leqslant \mathrm{j} \leqslant \mathrm{m}$ in the body of the rule are atoms.

[11] https://github.com/liveontologies/elk-reasoner
[12] https://www.derivo.de/en/products/konclude/
[13] http://www.hermit-reasoner.com/
[14] http://xsb.sourceforge.net/

2.6.2 Characteristics and Architecture

The first approach made to NoHR took into account the OWL 2 EL profile, allowing the user to query the combined knowledge base between the ontology and a set of nonmonotonic rules, and as we previously mentioned, the basic formalism for this approach is the well-founded semantics for MKNF knowledge base [17].

In [17], even queries to a very large ontologies as SNOMED CT, could be processed at a good response time. However, this tool had a lot to improve, such as the user interface, taking advantage of XSB language features.

The next step in [6], was to extend NoHR to deal with OWL 2 QL profile. This approach reduced pre-processing time by eliminating the need for classification of the ontology axioms, in return for a slightly longer query response time on average. This was achieved by translating the ontology directly into rules. The authors present the new results and the motivations to improve. Namely, the extension to OWL 2 RL profile, improving this OWL 2 QL profile using the classifier integrated in ontop or even the reasoner Konclude and also, using a relational database for the data as in Ontology-Based Data Access (OBDA) would be an interesting approach.

In [31], NoHR version was improved to support all polynomial OWL 2 profiles, and allowing for its usage with real-world ontologies that requires union of their language features and not only a single profile. The rule editor and parser was also improved, both in terms of user-friendliness and applicability. The authors concluded that the integration of database access could considerable reduce the time it takes to load prolog files with huge amounts of data in XSB, and the next version would feature this upgrade.

The last and current version of NoHR is presented in [21]. While the previous version (3.0) sometimes could have performance issues, essentially in terms of memory [31], this last version (4.0) has native support for databases, solving not only memory consumption problems, but also improving the average reasoning times.

The results confirm that using the new version is highly beneficial, especially for use cases involving large amounts of data, such as network inventory validation, because it avoids the overhead of transforming the database into facts in NoHR, reducing memory usage and time significantly.

Figure 2.2 whose source was [21], represents the current version of the NoHR system architecture. When compared with NoHR v3.0, the architecture has been extended with external databases, corresponding Open Database Connectivity (OBDC) drivers and the integration of the mapping knowledge base (marked with a green background).

The plugin can have the following input files: an OWL file; a rule file; a mapping file (predicates are created using mappings and are populated with the results of queries to external databases).

All three components can be edited in Protégé, with the ontology being edited using the built-in interface and the rules and mapping components being edited using the NoHR Rules and NoHR Mappings tabs, respectively.

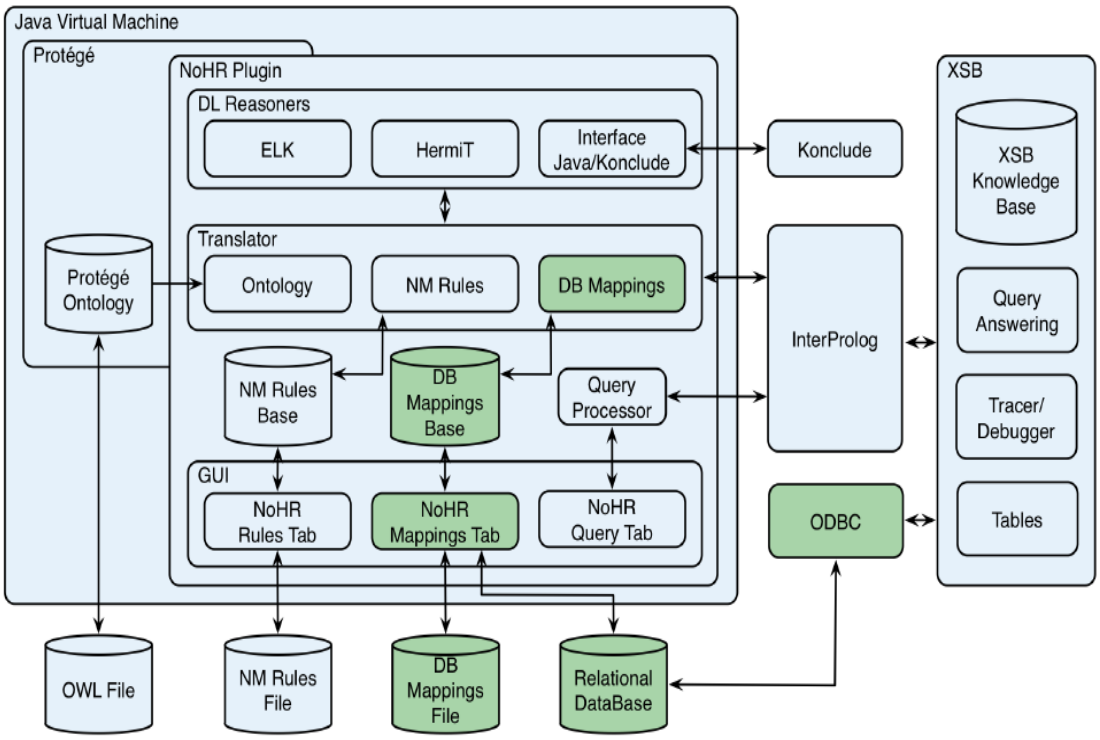

Figure 2.2: System architecture of NoHR v4.0 with native database support

The ontology is converted into a set of rules using one of the offered reasoners - ELK, HermiT, or Konclude - depending on the DL in which the ontology is written, after the inputs (which might be empty) and the first query are given. The resulting set is then combined with the input's rules and mappings. This combined result is passed to XSB Prolog via InterProlog[15], an open-source Java front-end that allows communication between Java and a Prolog engine, and the query is submitted to XSB to be executed over the same interface.

Mappings provide facts from external databases as they are asked in the reasoning process during execution. This method is supported by the installed ODBC connections and managed by XSB. The query processor receives the answers and displays them in a table to the user (Query Tab). The user can ask further queries, and the system will send them straight to XSB without any additional pre-processing. When a portion of the knowledge base is changed, the system just recompiles that part [21].

The Figure 2.3 illustrates the interface of Protégé with an example from [25] using NoHR. As we can observe, we have an ontology loaded, some rules, a query (with the answers) and the tabs mencioned before (NoHR Rules, NoHR Mappings and NoHR Query).

[15]http://interprolog.com/java-bridge/

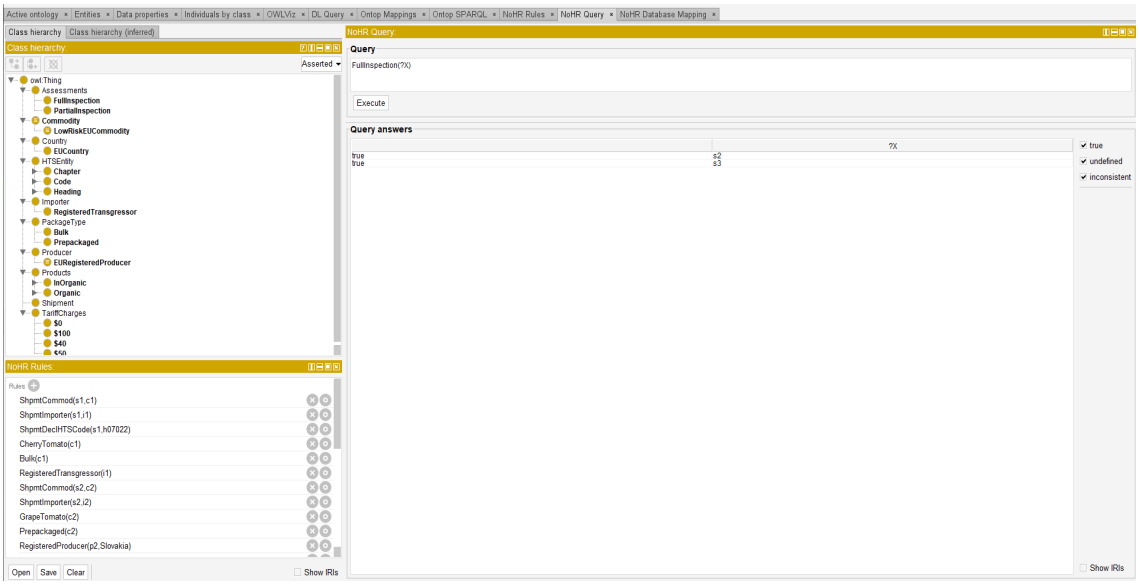

Figure 2.3: Protégé interface, with an example using NoHR

3

Related Work

In this chapter, we explore at some of the existing works that are related to ours. This chapter ends with a brief conclusion about everything that has been presented, and how these mentioned studies can be useful in the development of the dissertation.

3.1 Clinical Trials

Clinical trials have been the target of several studies that try to improve existing problems [8]. Around 80% of studies fail to recruit on time, requiring a study extension or the addition of new study sites. Furthermore, insufficient participant retention until the end of the study is a problem. The remaining number of patients will be insufficient to adequately answer the initial research questions. Aside from that, it has various ethical and economical implications, as well as delaying the study.

The patient selection process has been a manual process which turns out to be quite time consuming and error prone. In recent years, with the advancement of technology, some studies have emerged that allow improving this process using advanced technologies and other approaches [35]. Artificial intelligence, machine learning, deep learning, and the internet of things all provide ways of improving the efficiency and productivity of clinical trials at different stages.

For the purposes of this dissertation, a case study is described in [36], that investigates the application of ontology reasoning to problems in the medical domain. We briefly mentioned before that the authors refer some of the existing problems encountered, such as:

- **Terminology:** since the goal was to achieve a high degree of accuracy and coverage through a semi-automated process, it was necessary to map concepts in MED to the concepts in SNOMED CT. There are different terminologies, so the need to adopt a standard terminology emerged. In this sense, SNOMED CT has been adopted as national health-care standard in the United States.
- **Scalability:** for large ontologies, the performance was not the best, so it was necessary to try to find mechanisms to improve scalability.

  - **Large ABoxes:** instead of keeping in memory the map from refined individuals to real individuals, is now maintained in a database.
  - **Large TBoxes:** as the number of TBoxes was quite large and not everything was needed for reasoning. So, the concepts that were not needed from the original TBox were eliminated.

- **Noisy and Incomplete Data:** since the data in the patient records was incomplete with regard to SNOMED CT definitions, a support was offered to the users, specifying which terms of the definition were required. This is refered as query weakening. Also, clinical data is inconsistent. One example is that sometimes, two different laboratories run a test for the same disease and can result in contradictory results. To resolve this inconsistency, SHER was designed to efficiently detect multiple data inconsistencies.

Futhermore, clinical trials, which are mostly unstructured or free text, do not interact well with semantic interoperability standards developed for hospital systems [26]. It was necessary to use some standard terminology to overcome this problem. In the previous article, the authors describe how they used SNOMED CT standards to make clinical trials from a trial registry available to clinicians within an Electronic Health Record (EHR) system. The EHR itself is coded in either International Classification of Diseases (ICD) or SNOMED CT. They used a Natural Language Processing (NLP) engine to annotate clinical trials with their corresponding disease conditions. The results of this NLP process are clinical trials with associated disease conditions coded in SNOMED CT.

In 2018, a study was carried out in a way to design automated techniques that could support clinical researchers [3]. One particular idea in this study was to use a temporal extension of existing approaches for accessing data through ontologies written in DLs. For example, if we want to a more complex query such as "in some point in the past, we want to select patients with type X diabetes with a duration of at least 12 months", in such case, there are temporal operators for this complex temporal queries. The increased usage of EHRs in hospitals, according to the study, presents a promising opportunity to improve the recruitment process by automating portions of it. Another important aspect that was mentioned for future research is the problem of automatically translating natural language descriptions of eligibility criteria into formal queries in regard to a given medical ontology. This topic is very important for our dissertation.

In the last years, Text Retrieval Conference (TREC)[16] become quite important and is co-sponsored by the National Institute of Standards and Technology (NIST). Its purpose is to support and encourage research within information retrieval community by providing the infrastructure needed for large-scale evaluations of text retrieval strategies and accelerating technology transfer from the lab to the market. To be more specific, TREC

[16] https://trec.nist.gov/

is an ongoing series of workshops focusing on a variety of information retrieval research areas, or tracks.

The TREC evaluation campaign has run a lot of tracks in the medical domain. In TREC 2017 Precision Medicine Track a system for the participate at TREC 2017 was developed taking into account some considerations [30]:

- The typical issues of natural language processing and information retrieval, such as how to deal with synonyms or related terms, were covered in this track.
- Inclusion and exclusion criteria in ClinicalTrials.gov documents are combined in the criterion field in the XML documents. Precision could be improved by parsing the criteria field and distinguishing between inclusion and exclusion criteria.
- Individual eligibility requirements could be thought of as logical constraints, where exclusion criteria could be negated to become inclusion criteria. For example, an exclusion condition such as "age < 18", could be turned into "age >= 18" as inclusion criterion.
- Since medical texts often contain negated concepts or negative findings, to prevent mismatches and improve precision, a concept for should be represented as a negated concept, if found within a negation scope. For example, "no findings for tumor" should be represented as "not-tumor".

The system developed for the previous track task comprises a combination of approaches aimed to address the challenges previously mentioned. So, the results presented allows to conclude that a high performance could be achieved mostly because of document pre-processing (i.e. parsing criteria) and after overcoming some problems of the vocabulary [30].

Also, in TREC 2018 Precision Medicine Track in the Clinical Trials matching task, the NOVASearch team developed a system, where the information contained in the documents could be filtered, in order to provide only the relevant information. Therefore, it was created fields such as: Gender - genders acceptable for the trial; MinAge - the minimum age of patients acceptable for this trial; CriteriaInc - specific inclusion criteria for the patients; among other fields. So the filters implemented, allowed to remove clinical trials that didn't match some criteria [2]. In this dissertation, our goal is not read from a document but these studies are relevant, because they allow us to understand how different things are done, as is the example of information filtering, which could end up being useful.

3.1.1 SNOMED CT

As previously mentioned, SNOMED CT is very important for the development of this dissertation. It is currently the standard used in clinical trials.

There are other healthcare terminologies. For example, ICD, Logical Observation Identifiers Names and Codes (LOINC) and International Classification of Primary Care

(ICPC). The reasons why we chose to use SNOMED CT are:

- it is the most comprehensive, multilingual clinical healthcare terminology in the world;
- enables consistent representation of clinical content in Electronic Health Records;
- is mapped to other international standards.

To emphasize the importance of SNOMED CT even more, a recent study [43] was conducted on the pandemic that the entire world has been dealing with in recent months (COVID-19). The appropriateness of SNOMED CT in COVID-19 datasets was investigated in this research, and it was shown to be limited. However, the authors did show that SNOMED CT concepts capture typical data elements relevant to COVID-19 studies.

3.2 Justifications

Technology has advanced a lot in the last years and the use of AI methods more and more in everyday life, the European Union has proposed the General Data Protection Regulation (GDPR). Since AI approaches have become more complex affecting the understandability of the solutions, GDPR tries to tackles some problems, such as decisions made by AI systems should present explanations [11].

There has been a substantial amount of work on how to present justifications for both OWL and logic programming.

3.2.1 OWL Justifications

A justification for an entailment in an ontology in the context of knowledge-based systems is a minimal subset of that ontology that allows the entailment to hold. The set of axioms used to represent the justification should be minimal, so that if one of the axioms is eliminated, the justification is no longer valid.

Justifications are necessary for debugging unsatisfiable classes and contradictions in general. For large and complex ontologies, the availability of justifications improves the understandability. The capacity to generate explanations, or justifications, for inferences computed by a reasoner is often regarded as a valuable feature for both ontology development and reuse. The formal definition of justification is as follows: let $\mathcal{O} \models \alpha$ where $\mathcal{O}$ is a consistency ontology and α is an axiom, $\mathcal{J}$ is a justification for α in $\mathcal{O}$, if $\mathcal{J} \subseteq \mathcal{O}$, $\mathcal{J} \models \alpha$, and $\mathcal{J}' \not\models \alpha$ for every $\mathcal{J}' \subset \mathcal{J}$ [19].

There are two types of algorithms for finding justifications: 1) glass-box algorithms are reasoner dependent, i.e, the justifications are computed as a direct consequence of reasoning; 2) black-box algorithms are independent of the reasoner, i.e, only use reasoning

to see if a set of axioms leads to an entailment. Black-box algorithms typically require some satisfability tests and they only rely on the availability of a sound and complete reasoner for the logic in question [19]. The process of determining justifications for an entailment is described as transforming the entailment to an unsatisfiable class and then computing the justifications for that unsatisfiable class.

Computing a single justification using a black-box technique consists of two stages: (i) *expand* - to find a superset S of a justification; (ii) *shrink* - to find the final justification, because the set of axioms S might contain unnecessary axioms, so the algorithm contracts the set until it is minimal [18].

From an initial justification, we can compute the remaining ones using multiple techniques. The most used technique is the Reiter's Hitting Set Tree (HST) algorithm [39]. This technique is independent of the reasoner, i.e. a black-box technique, and practically effective. This algorithm constructs a finite labeled tree whose nodes in an HST are labeled with minimal conflict sets, and edges are labeled with axioms of $\mathcal{O}$. Given the duality of Reiter's algorithm, it may also be used to dynamically find all conflict sets, i.e. justifications, instead of only find all minimal hitting sets given a collection of conflict sets.

Computing all justifications using a glass-box technique is not straightforward as it requires to turn off some of the optimizations in the reasoner. Since these optimization techniques are important to the outstanding performance of existing OWL reasoners, disabling them makes this technique impracticable [19].

To be even more precise about the advantages of both techniques, black-box algorithms can be easily and robustly implemented, since they only rely on the availability of a sound and complete reasoner for such a DL. On the other hand, the glass-box algorithms usually have a better performance since they benefit from the tight integration with the reasoner. However, as previously mentioned, this technique requires to turn off some of the optimizations in the reasoner and frequently, as a result, this technique it is not always a viable option [18].

Protégé is currently the most popular tool for working with OWL ontologies. The value of explanations while dealing with OWL ontologies it is an important feature, and Protégé offers a dedicated extension point for plugins that provide explanations. It defines a high-level Application Programming Interface (API) for obtaining entailment explanations. Any explanation services plugin is responsible for two things: gathering information about why an entailment holds and presenting that information to the user in the form of a UI widget. Protégé shows a specific type of explanation known as justifications, which are the minimal subsets of the ontology required for the entailment to hold, and computes them using a black-box technique [22].

More recently, glass-box algorithms for computing justifications have been improved. Therefore, additional Protegé plugins were developed, such as justification explanation, which was improved as a result of the performance improvements of the glass-box algorithms. Since the number of justifications that were displayed could be quite big, one

solution was to set a limit on the amount of justifications displayed, with the option of loading more. The main goal was to optimize this process. This was achieved by separating the computation of all justifications from the plugins that display explanations to the user [22].

3.2.2 Justifications for Logic Programming

Also, it is possible to present justifications for logic programming. Understanding why a given query is true or false in a model of a program, under a certain set of semantics, is a difficulty that users of logic programming systems face. There are different ways to obtain and present the justifications to the user. This section will introduce one of them.

In order to obtain program justifications, the majority of approaches rely on the non-deterministic construction of complex structures, typically graph-based, or provide algorithmic approaches [7].

An explanation graph is a general method of representing justifications using a graph structure. [37] provides a complete definition of explanation graph. Importantly, an explanation graph is labeled and directed (N, E). There are some special nodes:

- $\top$ meaning nodes that denote true facts and their truth does not depend on other atoms;
- $\bot$ atoms without rules - the falsity is independent on other atoms;
- *assume* – used for atoms for which no explanations are sought.

Each edge of the graph connects two annotated atoms or an annotated atom with one of the nodes in $\{\top, \bot, assume\}$, and it is labeled with one of the symbols from $\{+, -\}$. Positive edges are denoted by the symbol '+', whereas negative edges are denoted by the symbol '–'.

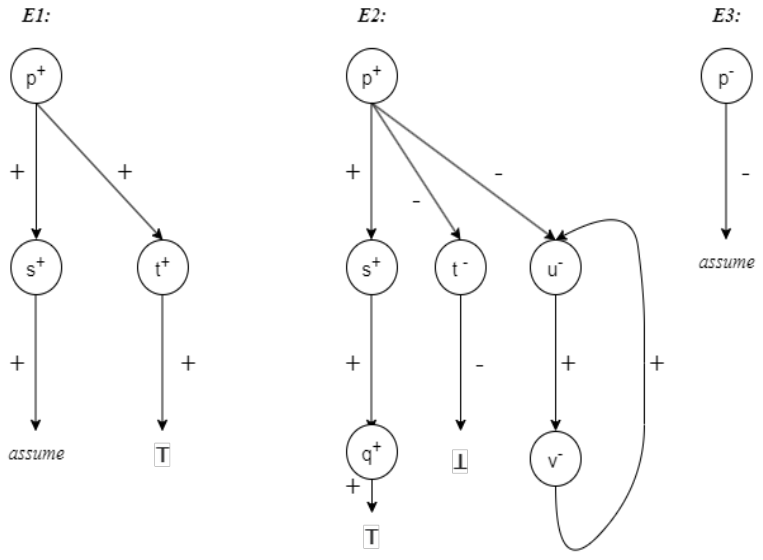

Figure 3.1: (Adapted from [37]) Explanation graphs

Figure 3.1 illustrates three examples of explanation graphs:

- The graph ***E1:*** describes the true state of p by making it dependent on the true states of s and t; s is assumed to be true and t is fact. The following program corresponds to this graph:

$$p \text{ :- } s, t. \qquad s \text{ :- } t. \qquad t.$$

- The graph ***E2:*** explains more complex dependencies. Notice that u and v are both false and mutually dependent, as in the case of a program containing the rules:

$$u \text{ :- } v. \qquad v \text{ :- } u.$$

  Moreover, note that t is defined as false because there are no rules that define it.

- The graph ***E3:*** asserts that p has been assumed to be false.

Given an explanation graph and an atom, it is possible to extract from the graph the elements that contribute directly to the truth value of the atom. However, and despite the fact that there are additional described methods for presenting logic programming's justifications - Off-Line Justifications and On-Line Justifications [7, 37], we realized that their presentation was not ideal for achieving our goal. This is because the way they would be presented would necessitate knowledge of logic programming for the user to comprehend them, or it would be necessary to convert graphs into sentences, which would affect the algorithm's performance.

We intend to provide simple, comprehensible and legible justifications for all users, regardless of their knowledge of logic and programming.

3.2.3 Conclusion

Overall, we noticed a lot of research were conducted, to improve clinical trials success rate and also to find justifications. However, when it comes to create a mechanism to present justifications, things get a little more tricky. There is nothing concrete that presents justifications when integrating rules with ontologies.

After what has been mentioned, we highlight the fact that Protégé already has a number of plugins developed to provide explanations. Explanations of OWL entailments in Protégé are provided as justifications. Although they are not yet fully optimized as there is room for improvement in the future, they could be useful.

In order to produce justifications, we could have used some other tools and services, and develop the algorithms necessary for our application. There are some explanation services responsible for the computation about why an entailment holds and others are responsible for displaying that information to the user. Thus, we could have used plugins

that can be provided using an OSGi[17] framework or, specifically, its implementation used for the Eclipse IDE [22]. By implementing an explanation service, we could have picked two plugins such as, 'black-box justifications'[18] for computing justifications using a black-box algorithm and 'justification explanation'[19] for displaying the justifications. A more detail approach for the services and plugins is presented in [22].

There are some viable options for achieving this goal, some of which may be preferable in terms of algorithm performance, whereas others are preferable in terms of providing the user with more perceptible justifications. We need to find a balance between performance and legibility of justifications. Therefore, the use of a black-blox algorithm appears to be a good option for justifications regarding ontology terms, according to the studies, because it is efficient and easier to implement.

[17] https://www.ibm.com/docs/en/was-nd/9.0.5?topic=applications-osgi-framework
[18] https://github.com/liveontologies/protege-black-box-justification
[19] https://github.com/liveontologies/protege-justification-explanation

4

Patient Selection

In this chapter, we address an overview of the back-end architecture and a detailed description of patient selection for clinical trials implementation using the NoHR tool. Furthermore, we discuss some decisions made and a positive conclusion in favor of automating the patient selection process.

4.1 Overview

The objective of this dissertation, as previously stated, was to develop an application that not only automatically selects patients for clinical trials, but also presents the respective justification for why each patient was chosen. To this end, it was necessary to divide the development into two main steps. First, we focused only on the development of the back-end and the creation of our patient database, testing the implementation as well as necessary algorithm improvements. Later, we aimed to the development of the front-end application and the connection to our back-end.

It is worth noting that the entire back-end was implemented in Java. The java project created is actually a maven project. Maven is a build automation tool used primarily in Java projects developed by Apache [20], it serves to manage dependencies and automate the builds. It addresses two aspects of software development: how the software is built and its dependencies. The software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins are all described in an XML (pom.xml) file. Our project is divided into modules: i) API module; ii) two modules regarding NoHR - reasoner and benchmark; iii) the OWL explanation module. These modules are defined in the project's general configuration file *pom.xml* and thus allow interacting with the features of each module.

In section 6.3, we elaborate in detail all of the application's steps, beginning with the application receiving the criteria on a request and all the steps until the application produces the outputs of which patients were selected and the respective explanations for each selection.

[20] https://maven.apache.org/

This section describes the back-end architecture in detail. Its development incorporates NoHR, which is the tool that allows the integration of hybrid knowledge required for patient selection. A representation of the back-end architecture is presented in Figure 4.1, where the NoHR and XSB blocks are reduced to a simple component since this architecture has already been described in a previous chapter and presented in Figure 2.2.

Through this diagram, we illustrate the essential components developed so that it was possible to automate the selection of patients, as well as the justifications, which are the purposes of this dissertation. The entire implementation of the justifications, necessary adaptations and decisions taken are covered in chapter 5.

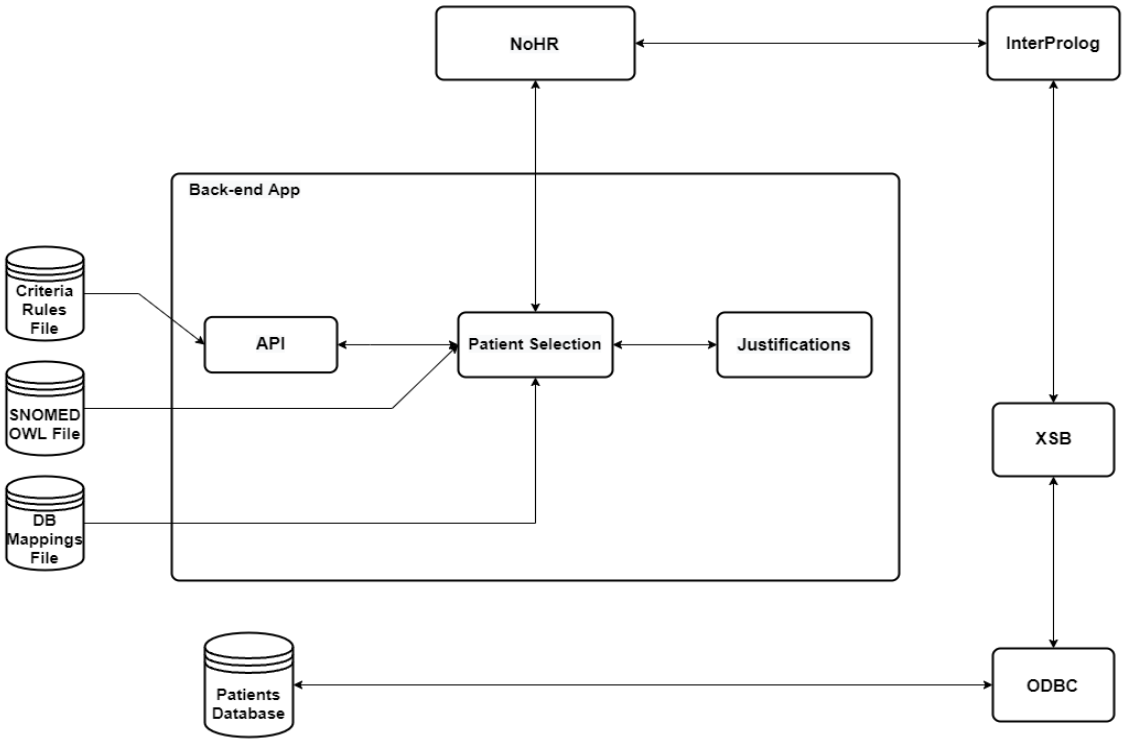

Figure 4.1: System architecture of back-end

Initially, the back-end architecture had no API because it was possible to test the functionalities without one, in which case the criteria were passed directly to the Patient Selection block. Recognizing that it would be necessary to have an API that would allow the front-end to interact with the back-end in the future, this API was implemented early so that a user-supplied set of criteria could be accepted. The trial criteria are transmitted via a POST request and the selected patients are obtained via a GET request. After receiving a request, the API provides this set of criteria as input to the Patient Selection block.

The API thus interacts with the block that deals with selecting patients. Here we have the SNOMED-CT ontology as input and also the file with the database mappings. These mapping files enable us to extract the data and information required for our application

to work properly for the criteria that we have implemented from the patient database. Unless the developer manually changes the file with the data to be mapped, the mapping file is always static. The file currently contains mappings for:

- **Age**;
- **Alive Patients:** because we only want to select alive patients for the studies;
- **Patient Information:** Id, first name, last name, SSN, and birth date;
- **Allergies**;
- **Medications**.

This block passes these inputs to NoHR as well as the file with the rules and, at the end of its execution, returns which patients were selected. Section 4.2 goes into full depth about the Patient Selection block.

The final step is to generate explanations of why each patient was selected. This block is a little more complex as it requires special attention for certain types of criteria, particularly regarding ontology-based justifications, as well as concern in terms of readability.

4.2 Implementation

The initial step towards automating the selection of patients for clinical trials was the creation of a database containing information from several individuals in order to acquire some outcomes. Using healthcare data for research can be complicated, and there may be multiple legal and financial hurdles to overcome in order to use certain data. This is especially true when dealing with particular patient information. To avoid these issues, it was necessary to use synthetic data (very similar to the real ones) to represent the possible candidate patients. We used a generator called Synthea™ to create a patient database that we could use to test our implementation. It is an open-source software package that simulates the lifespans of synthetic patients. Synthea™ provides a high-quality synthetic, realistic but not real, patient data and associated health records covering all aspects of healthcare.

Although there are other data generators such as HAPI FHIR[21], we chose to use Synthea™ for several reasons. All other options were incomparable to Synthea™ because they lacked the completeness that Synthea™ offers, being the most popular and best option in different studies [46, 44]. Moreover, Synthea™ generates data with no cost, privacy, or security constraints. It is free to be used for a wide range of secondary purposes such as research, industry, and government. J. Walonoski et al. [44] introduced Synthea™ that advances previous works in several ways. First, by accepting only publicly

[21] https://hapifhir.io/

available information and health statistics as inputs, Synthea^{TM} ensures fully synthetic output. Therefore, all of the data generated by Synthea^{TM} is based on real data, with the generated patients having random names with a few numbers, as well as extremely realistic synthetic data. Second, their method generates data based on clinical workflow and disease progression models that can be easily inspected and modified, facilitating transparency and continuous improvement. Third, their method is scalable because it facilitates collaboration among experts from various clinical and technical backgrounds. Finally, the data generated by Synthea^{TM} is in accordance with most of the terms of our SNOMED-CT ontology and is thus an asset to the success of this application.

4.2.1 Criteria

Before we started developing the application, we needed to examine samples of criteria from real Clinical Trials provided by the U.S. National Library of Medicine to figure out what kind of mappings from our database we would need and then how we would display the selected patients.

Obviously, it would be impossible to implement all existing criteria in the real world, so we focused on some types of criteria that are used in clinical trials. However, the database is already prepared to accept the addition of certain criteria; all that would be required is the addition of the new mappings. This is not a barrier because the application is extendable and no changes to the architecture are required.

The criteria chosen for this dissertation are: i) regarding the maximum and minimum age; ii) gender specification; iii) relating to allergies; iv) medication taken. We believe that choosing these criteria over others was a good decision because they are common in many trials, there is usually a lot of information on each patient about their allergies and medications taken, and they are easier to understand. With these criteria that we have implemented, it was possible to demonstrate that it is indeed possible to follow this path to achieve the goal of automating the patient selection process. For the application to accept additional allergies and medications as criteria, it would only be necessary to add the names associated with allergies and medications to the application's menu, as the database is already prepared and the mappings for these types of criteria have already been created.

4.2.1.1 Minimum and Maximum Age

One of the most common criteria regarding clinical trials is the question of maximum and minimum age required. For this, it was implemented for this selection of patients, the option to define what would be the minimum and maximum age necessary for a patient to be selected to participate in the trial.

The user may insert the value of minimum age and/or maximum age required for the study and only patients who meet the age requirements are possible candidates to be

selected. This is a necessary criterion because there are actually many studies that refer to maximum and/or minimum age values. Mainly in terms of being adult patients or not.

4.2.1.2 Gender

The patient's gender is another criterion that is frequently included in trial's criteria. Sometimes it is irrelevant if the patient is male or female, but for some trials it is necessary to be either exclusively male or exclusively female.

The gender criterion is also taking into consideration and evaluated. When there is no information regarding the gender criterion of each patient, then it is because both male and female patients can be selected. Otherwise, if some criteria are specified regarding gender, then it is because for this study we only want patients of that specific gender, excluding all patients of the opposite gender.

4.2.1.3 Allergies

To begin testing the mappings related to the created database and to confirm that the patient selection was accurate, allergy-type criteria served as the starting point. Which are in fact examples of typical clinical trial criteria. This can be both inclusion or exclusion criteria depending on whether we want to select patients with a given allergy or without a given allergy.

Within allergies, there is a very wide range of distinct types of allergies and relationships between allergies that it would not be possible to map all of them. For this dissertation, based on some clinical trials and based on existing allergies in the database, we selected some allergies to implement as criteria. The database is prepared to accept additional allergies as criteria; however, we only implemented a few of them in the application so that the user did not have complete freedom to enter whatever he desired, thereby preventing input errors. Possibly, the association file between individuals and ontology terms would need to be updated in order to ensure that the new allergies have the appropriate association and thus can be added as a criterion to the application.

The allergies implemented and functional are:

- *Eggs* (*edible*) (*substance*)
- *Seafood* (*substance*)
- *Soya bean* (*substance*)
- *Animal dander* (*substance*)
- *Pollen* (*substance*)
- *Latex* (*substance*)
- *Mold* (*organism*)

- *Food allergen* (*substance*) - in this particular case, we are dealing with a more general criterion. This means that being allergic to food requires the analysis of several allergies. For example, if the patient is allergic to eggs he will also be allergic to food.

After a deeper analysis of the SNOMED-CT ontology and the names assigned to each allergy, we came to the conclusion that it would be better to use the exact same name for each allergy corresponding to the ontology in the application for the criteria. Although it would seem more elegant to appear only, for example - Allergic to *Soya bean*, it would be more error prone because for each case, the program would have to validate the similar terms to associate them later with the ontology classes. In this way, the criteria are exactly consistent with the ontology terms and so we avoid validating them one by one as they will be compatible.

4.2.1.4 Medications

Similar to allergies, there are many different medications on the market, making it hard to map them entirely. For this reason, only a limited number of medications were considered. However, it would also work for other medications. It would only be necessary to review the file containing the associations between individuals and ontology terms, and the addition of this new medication as a criterion in the application.

This criterion may alternatively be exclusion if we only want patients who have not taken a particular medication, which would exclude any patients with a history of that medication, or inclusion if we only want patients who have taken a particular medication.

Medications considered functional in the application are:

- *Lisinopril* (*product*)
- *Ibuprofen* (*product*)
- *Naproxen* (*product*)
- *Penicillin V* (*substance*)
- *Non – steroidal anti – inflammatory agent* (*product*) - this is another example that is a more general criterion. This allows us to observe more complex justifications in a more general criterion.

For the same reasons mentioned in the allergies sub-section, these medications also have the names that correspond exactly to their SNOMED-CT names.

4.2.2 Database

To simulate clinical trials, we required access to patient data and their medical records. Within the scope of this dissertation, a database was required so that we could simulate trials based on patient data and medical information. Because we did not have access to

any database that we could use, and primarily for privacy reasons, it was necessary to create one. This database was created using MySQL[22], based on the files generated by Synthea™.

After Synthea™ is executed, the CSV exporter will generate the following files:

- **allergies.csv** - patient allergy data.
- **careplans.csv** - patient care plan data.
- **claims.csv** - patient claim data.
- **claims_transactions.csv** - transactions per line item per claim.
- **conditions.csv** - patient conditions or diagnoses.
- **devices.csv** - patient-affixed permanent and semi-permanent devices.
- **encounters.csv** - patient encounter data.
- **imaging_studies.csv** - patient imaging metadata.
- **immunizations.csv** - patient immunization data.
- **medications.csv** - patient medication data.
- **observations.csv** - patient observations including vital signs and lab reports.
- **organizations.csv** - provider organizations including hospitals.
- **patients.csv** - patient demographic data.
- **payer_transitions.csv** - payer transition data (i.e., changes in health insurance).
- **payers.csv** - payer organization data.
- **procedures.csv** - patient procedure data including surgeries.
- **providers.csv** - clinicians that provide patient care.
- **supplies.csv** - supplies used in the provision of care.

These files are generated in excel files by configuring the export to CSV format provided by Synthea™, but we could have opted for one of the other allowed formats. As a result, the following files were converted into tables in the newly created database: **i)** allergies.csv; **ii)** careplans.csv; **iii)** conditions.csv; **iv)** devices.csv; **v)** encounters.csv; **vi)** immunizations.csv; **vii)** medications.csv; **viii)** organizations.csv; **ix)** patients.csv; **x)** payers.csv; **xi)** providers.csv. On the Synthea page [23], there is more information available about each CSV table, including the primary and foreign keys, which was very useful for us to have a patient database with data that was very similar to the real ones without worrying about security and privacy issues.

[22] https://www.mysql.com/

[23] https://github.com/synthetichealth/synthea/wiki/CSV-File-Data-Dictionary

This is a static database for the scope of this dissertation reducing the number of possible errors. This is due to the fact that no triggers, updates or removes were created that could cause inconsistencies which could lead to errors.

The majority of csv files generated by Synthea^{TM}, as previously stated, have a corresponding table created. Those not mentioned were considered to be too much additional information that would not be necessary for the purposes of the dissertation. Even so, of all the tables that were created, not all of them were actually used. However, since they are already created, they simplify the process of adding new criteria to these tables, because it would only be necessary to create the new mappings and add them to the application's menu.

The most important tables, which we used to carefully evaluate the criteria we chose to implement, are the following:

- **Patient:** this table contains information about each patient, such as the date of birth, their id, their gender, among other information.
- **Medication:** this table contains information about which medications were taken, by which patient they were taken, cost, description, among other information.
- **Allergie:** this table contains information about the allergies that each patient has, the reactions they had, when they were recorded and the severity of the reaction.
- **AgeInfo:** in order to compare the minimum and maximum age criteria with the age value of each patient, we created a view so that each patient had an integer number associated with their age. The patient table only allows for the retrieval of the patient's date of birth and a date as a criterion is a little more complicated than just entering a minimum and maximum age value. A view is a database object that allows generating a logical subset of data from one or more tables. Since we wanted to get the age of the patients from the table of patients that had the date of birth, a view is enough, not having to create another table for this purpose.

In addition to the mentioned tables, others that contain additional types of information that can be used to develop new criteria have been created. Thus, these tables make the application easy to extend. Some of the tables created that could potentially be used in the future to implement new types of criteria and the respective mappings are:

- **CarePlan:** if it is intended to select patients with a specific care plan, within a certain period of time;
- **PatientCondition:** in case it is intended to select patients who have been diagnosed with a condition;
- **Device:** when studying some devices, this table may allow the creation of a mapping to select only patients who are using a particular device or not;
- **Immunization:** when it is intended to study immunizations, this table may allow to extend the application in this sense by creating a map of the applied immunizations.

Furthermore, it was necessary to create a file with base rules so that everything works properly. This is because initially the ontology does not actually have any patient associated with the classes, which would end up not generating correct results. As a result, it was required to create a rules file, different from the criteria file, for the association of individuals with the ontology terms. In the event that new criteria are added, this file will also need to be updated, which is not a particularly difficult task.

This enabled the simulation of clinical trials that are very similar to real ones. In particular, we focused on Clinical Trials studies provided by the U.S. National Library of Medicine.

Also, as the information that exists in the database regarding each patient is vast and it is only important to present the patient's name, age and gender, a PatientInfo object class was created so that a list of the selected patients can be saved only with the necessary information.

4.2.3 Trial Example

One of the actual clinical trials analyzed from ClinicalTrials.gov was the following: *Athlete Whey Protein Sensitivity: Prevalence and Performance*.

The objective of this study was to identify the prevalence of whey protein sensitivity in athletes and to assess the effectiveness of 4-weeks of whey versus plant-based protein supplementation on athletic performance and recovery, specifically in those with whey sensitivity.

This study had the following criteria:

- **Ages Eligible for Study:** 18 Years and older
- **Sexes Eligible for Study:** All
- **Inclusion Criteria:**
  - NCAA Division I collegiate athletes at the University of the Incarnate Word;
  - must complete a medical history form;
  - must be cleared by sports medicine staff for intercollegiate athletic participation.
- **Exclusion Criteria:**
  - individuals with a diagnosed food allergy, lactose intolerance, or inflammatory bowel disease.

We were able to conclude from these criteria that there is a minimum age to participate, which is 18, and that both female and male athletes can be chosen. In this case, it would make no difference whether or not the gender criterion was implemented, but in other studies analyzed this is indeed important and essential.

Although this is not an impediment, it is important to note that some of the analyzed criteria cannot be represented using this tool because it is impossible to translate them into rules. The focus of this dissertation is to demonstrate that it is possible to automate

the patient selection process for the criteria that this automation really makes sense, and its implementation. Of course there are certain exceptions. For example, in this presented trial it would be impossible to guarantee that the athlete would have completed the medical history form automatically. This is the case that would be something approved after our application selects the patients. This usually occurs when it is absolutely necessary to obtain confirmation from a superior that a document has been signed or completed. In these circumstances, the idea is that once the patients are automatically selected, only those who truly present what may be missing are chosen. Nonetheless, automating this procedure of selecting candidates for a trial is beneficial because the majority of the requirements can be written as rules and that is what we focused on.

This previous example and other studies, allows us to conclude that the criterion regarding allergies is indeed important. Other studies carried out, which have been analyzed, also often refer to some criteria regarding the drugs taken. Therefore, these were the criteria that we decided to implement and test during the development of the dissertation.

4.3 Decisions

During the development of the back-end application with the NoHR integration, some issues arose which ended up hampering the beginning of development, having to make some decisions and make some adaptations.

Starting with the Java version, some problems occurred whenever we attempted to run with a version of Java other than 8. Therefore, we decided to use only Java 8 throughout the development and testing of this project.

As expected, it was not so linear to integrate NoHR with the early versions of our code (back-end), which we needed to test. The most challenging part was figuring out how to simulate the criteria that would be subsequently sent to NoHR as nonmonotonic rules. Initially, we chose to manually test the criteria and always compare the results to what Protégé returned as output using the NoHR plugin. Later, we decided that it would be better if, when a user fulfills the desired criteria, it is only those that we allow to ensure that there are no errors in names that are incompatible with the SNOMED-CT ontology. This was also achieved because the criteria are always passed with exactly the same name as the terms of the SNOMED-CT ontology. Otherwise, it would be necessary to have an intermediate step in the algorithm where it would go through each criterion and associate the respective criterion with the corresponding name in the ontology. In this way, this step is no longer necessary, thus being more efficient. As a result, we can ensure the success of the criteria and a smoother conversion from the front-end to the back-end.

Since the database contained only the date of birth for each patient, and it was necessary to obtain the ages of each patient as integers, an AgeInfo view was created to store

the age of each patient as an integer number in addition to the date of birth. This allowed both the minimum and maximum age criteria to become ready for implementation.

Despite the benefits of NoHR in the integration of hybrid knowledge that have been stated, the initial ontology loading time with new rules necessary to make a query was around an hour or longer each time a new test was performed. Due to the static nature of our SNOMED-CT ontology and the fact that only the criteria (the rules) change, we have implemented our back-end in a similar manner to how the NoHR plugin version was created. The main idea is that the program only reloads when the rules have changed (new trial) and we just have to update the rules. Due to the fact that the mappings file is also static and does not change between trials, we do not need to update the mappings. Additionally, the size of the original SNOMED-CT ontology was reduced further to enhance the performance of the application. This is because many of the classes were unused, and consequently, some of them were eliminated. Later, these results will be presented in Chapter 7.

5

Justifications

In this chapter, we cover the implementation regarding to the presentation of automatic justifications for selected patients. The differences between more complex and simpler justifications are also addressed, along with some results as well as some improvements made to the original algorithm.

5.1 Algorithm approach

The development of a mechanism that provides automatic justifications for why a particular patient has been selected is very important in the scope of this dissertation. Since we wanted to automate the process of selecting patients for clinical trials, it is important to ensure that the results obtained are valid and that there are no errors. This does not validate that the patient selection is correct, but it simplifies the entire validation process and makes it easier to detect errors because explanations are provided for each of the criteria, allowing us to verify whether it is true more efficiently than if we were to manually examine each criterion individually. This is due to the fact that it is possible to verify whether each selected patient actually has a valid justification that makes sense according to the criteria and the information related to that patient.

There is no point in automating something where the expected results are the wrong ones. Without these justifications it would be difficult to validate if a certain patient was selected correctly and this would in fact require manual verification of each selected patient. The justifications, being generated automatically, allow validating all the results of each trial and detecting any errors in advance.

The approach to achieve a correct and feasible solution had to take into account hybrid knowledge. In this case, we have data of different types. We have the SNOMED-CT ontology, the patient database, the mappings created and the rules regarding the criteria of each trial.

Thus, being something little explored until today due to the complexity involved and because it is non-linear to create something that automates the patient selection procedure with their explanations while still producing reliable results, we tried to investigate the

best option to generate reliable explanations. Taking everything into consideration, we came to the conclusion that it would be advantageous to divide the justifications into two main steps. These steps were crucial since it was through them that we were able to deal with multiple error cases and ensure accurate results.

5.1.1 First Step of the Algorithm

To begin with, after obtaining the selected patients, the algorithm will go through the list of patients, and for each of them, it will go through each of the trial criteria and queries the patient's database to obtain a possible justification.

In other words, the algorithm tries to find something with that criterion that is present in the database regarding that patient. If it is the case that this information is present in the database related to that patient, then this is a plausible justification. In section 5.2.1 an example of this type of justifications is presented and later, a visualization of these justifications provided by the web application is displayed in a later section supported by Figure 6.5.

On the other hand, if such information cannot be obtained and since the patient has already been selected, it is necessary to consult the ontology's current terms and relationships to obtain a plausible and explanatory justification. It is required to use the ontology rather than another file because the ontology contains all of the medical information, such as all existing allergies, relationships between diseases, medications, and so on. This situation may not be so simple to find a solution because a simple ontology query is not enough. It is necessary to find the relationships between the classes in the ontology until a relationship is found between the class corresponding to the criterion and a class corresponding to the information found for that patient. Therefore we move on to the second step of the algorithm.

5.1.2 Second Step of the Algorithm

This part, as already mentioned, is much more complex. It is not enough to make a simple query to the ontology as is done to the database in the previous step. The implementation of something that provides justifications for entailments regarding SNOMED-CT Ontology was confirmed to be feasible thanks to [19] and specially to this source code [24]. A. Kalyanpur et al. presented several algorithms for computing justifications of an entailment in an ontology. It was through some of the studied algorithms and approaches, more precisely a black-box algorithm, that we were able to find a viable solution that resulted in valid justifications.

However, since our ontology did not contain any individuals at the beginning and the goal was to discover explanations for some individuals (selected patients), it was necessary

[24] https://github.com/matthewhorridge/owlexplanation

to modify the algorithm and make some adaptations. It would not be ideal to add all patients and their allergies, medications and so on to the ontology, even before knowing which patients were selected. Therefore, we decided to add this information only after the patient selection process was complete, so that justifications for the trial criteria would be generated only for those patients who were selected. Otherwise, we would be adding more information, which would increase processing time and may have an effect on the algorithm's performance. In the end, there were other options for achieving the same goal, but this one seemed to be the most appropriate because we ensured that the associations of individuals to ontology classes were correct.

First, to make it possible to obtain the justifications for each patient, it was necessary to obtain all the information regarding each criterion of the trial for each patient by performing a query to the database and add each individual to the corresponding classes in the ontology. To clarify with an example, if a criterion were - allergic to X - what the algorithm does is search for all allergies the patient has, and add that individual to the classes that correspond to each allergy of that patient. As a result, the ontology is modified to include individuals - selected patients for the current trial - who fit into the classes, making it possible to generate justifications.

We always considered the readability of the explanations and the manner in which they would be presented. We considered two possible options in this regard. First, we considered presenting justifications with the ID of each individual (selected patients) and the Internationalized Resource Identifier (IRI) of each class. Alternatively, instead of presenting IDs and IRIs, we tested how they would appear if justifications were presented with the patient's first and last name and the names corresponding to the ontology classes. This second option appeared to be the best course of action, as it allowed for a more comprehensible reading and a clearer understanding of the justifications, thereby making it easier to detect any errors.

An example of this kind of justification is also provided in section 5.2.2, though it is not so easy to read at first. To ensure everything is accurate, a more careful and detailed interpretation is required. In a later section, we present an example of these justifications provided by the web application supported by Figure 6.6.

5.2 Examples

In this section, some examples of obtained justifications are presented. These results were obtained using only back-end tests and no front-end application. For this purpose, it was created a class to test the features before even connecting the back-end to our front-end.

These results are obtained by connecting to the previously mentioned patient database, as well as by using the SNOMED-CT ontology and a set of criteria as inputs to NoHR. Subsequently, the examples are later presented in the chapter regarding the front-end application.

5.2.1 Example without ontology

Figure 5.1 shows an example of a trial performed to test the back-end. The criteria for this trial were: i) minimum age of 16; ii) maximum age of 50; iii) must be female; iv) allergy to Soya bean; v) have taken Penicillin medication.


```
{
  "The selected patients are": [
    {
      "Id": "51408513-6d8a-6afe-11e1-883a9bcb448f",
      "Name": "Frieda277 Konopelski743",
      "Age": "37",
      "Gender": "F",
      "This patient was selected because":{
        "MinAge 16": "patient has 37",
        "MaxAge 50": "patient has 37",
        "Gender Female": "patient is Female",
        "Allergy to Soya bean (substance)": "patient has Soya bean (substance) allergy",
        "Took Medication Penicillin V (substance)": "patient consumed Penicillin V Potassium 500 MG Oral Tablet"
      }
    },
    {
      "Id": "26f4fc28-66d8-919a-c37e-ee1389267592",
      "Name": "Connie24 Bartell116",
      "Age": "24",
      "Gender": "F",
      "This patient was selected because":{
        "MinAge 16": "patient has 24",
        "MaxAge 50": "patient has 24",
        "Gender Female": "patient is Female",
        "Allergy to Soya bean (substance)": "patient has Soya bean (substance) allergy",
        "Took Medication Penicillin V (substance)": "patient consumed Penicillin V Potassium 250 MG Oral Tablet"
      }
    }
  ]
}
```


Figure 5.1: Selected Patients with some database justifications

In this example, the justifications do not require to consult the ontology, since the queries to the database already provides the necessary information to present the proper justifications.

5.2.2 Example with ontology

Figure 5.2 demonstrates another example of a back-end testing trial. The criteria for this trial example were: i) minimum age of 18; ii) maximum age of 75; iii) must be male; iv) allergy to any food substance; v) have taken any non-steroidal anti-inflammatory agent product; vi) not have taken any medication related to Lisinopril products.

In this case, the food allergy criterion, as well as the criterion of having taken anti-inflammatory agent products require another step in the algorithm to find the proper justifications. As no information in the database could justify this selection, the justification's algorithm for these criteria must consult the SNOMED-CT ontology. Then, it is necessary to look for relationships between the classes corresponding to the information in the database and the class corresponding to the criterion. To be more precise, for example, if a patient has allergy to eggs and the criterion is to be allergic to food, he was selected because the food allergy class in the hierarchy is a higher class than the egg allergy class which is a sub-class.

```
{
   "The selected patients are":
    [
      {
        "Id": "710190e8-ced9-cfb4-4721-b35ec2dfc906",
        "Name": "Francis500 Kirlin939",
        "Age": "57",
        "Gender": "M",
        "This patient was selected because":{
          "MinAge 18": "patient has 57",
          "MaxAge 75": "patient has 57",
          "Gender Male": "patient is Male",
          "Allergy to Food allergen (substance)": "patient has [Explanation <ClassAssertion(<Food allergen
(substance)> <Francis500 Kirlin939>)> ClassAssertion(<Eggs (edible) (substance)> <Francis500 Kirlin939>)
SubClassOf(<Eggs (edible) (substance)> ObjectIntersectionOf(<Foods (substance)> <Food allergen (substance)>)) ]",
          "Took Medication Non-steroidal anti-inflammatory agent (product)": "patient consumed [Explanation
<ClassAssertion(<Non-steroidal anti-inflammatory agent (product)> <Francis500 Kirlin939>)>
ClassAssertion(<Ibuprofen (product)> <Francis500 Kirlin939>) SubClassOf(<Ibuprofen (product)>
ObjectIntersectionOf(<Non-steroidal anti-inflammatory agent (product)> ObjectSomeValuesFrom(<Has active ingredient
(attribute)> <Ibuprofen (substance)>))) , Explanation <ClassAssertion(<Non-steroidal anti-inflammatory agent
(product)> <Francis500 Kirlin939>)> ClassAssertion(<Naproxen (product)> <Francis500 Kirlin939>)
SubClassOf(<Naproxen (product)> ObjectIntersectionOf(<Non-steroidal anti-inflammatory agent (product)>
ObjectSomeValuesFrom(<Has active ingredient (attribute)> <Naproxen (substance)>))) ]",
          "Not have taken the Medication Lisinopril (product)": "patient did not take any medication or related
medications with Lisinopril (product)"
        }
      },
      {
        "Id": "b707e378-53be-03c1-6bc5-18d0255e9958",
        "Name": "Nicky270 Beier427",
        "Age": "30",
        "Gender": "M",
        "This patient was selected because":{
          "MinAge 18": "patient has 30",
          "MaxAge 75": "patient has 30",
          "Gender Male": "patient is Male",
          "Allergy to Food allergen (substance)": "patient has [Explanation <ClassAssertion(<Food allergen
(substance)> <Nicky270 Beier427>)> ClassAssertion(<Shellfish - dietary (substance)> <Nicky270 Beier427>)
SubClassOf(<Shellfish - dietary (substance)> ObjectIntersectionOf(<Food allergen (substance)> <Seafood
(substance)>)) ]",
          "Took Medication Non-steroidal anti-inflammatory agent (product)": "patient consumed [Explanation
<ClassAssertion(<Non-steroidal anti-inflammatory agent (product)> <Nicky270 Beier427>)> ClassAssertion(<Naproxen
(product)> <Nicky270 Beier427>) SubClassOf(<Naproxen (product)> ObjectIntersectionOf(<Non-steroidal anti-
inflammatory agent (product)> ObjectSomeValuesFrom(<Has active ingredient (attribute)> <Naproxen (substance)>))) ]",
          "Not have taken the Medication Lisinopril (product)": "patient did not take any medication or related
medications with Lisinopril (product)"
        }
      }
    ]
```


Figure 5.2: Selected Patients with some complex justifications

6

Web App

This chapter describes the application in more detail, then introduces the interface that was developed as well as the technology that was used to create the front-end web application. A user manual is also included with some examples of the results obtained.

6.1 App approach

In previous chapters, we explained the entire implementation of the back-end to be able to automatically select patients for clinical trials and how we generate explanations for each selection. This requires the user to know what type of data is being used, how to work with ontologies, how to pass inputs (criteria) correctly, among others. It is actually necessary to have some knowledge in the IT area to be able to test things with just the back-end or using Protégé.

As previously stated, Protégé was an important tool for double-checking back-end results. Indeed, it is required some knowledge to use Protégé or even the back-end application we developed with NoHR, the adaptations we have made, the optimizations and the new features. Because NoHR's setup is not linear, it becomes a major barrier to automating this process. Even if we had opted for another tool similar to NoHR, regardless of the tool used, the problem always appears to be the same, the usability issue.

This raised some questions:

- Will the user (a doctor for instance) require computer skills in order to use an application to make the patient selecting procedure automatic?
- Will the user need to have any training or take a particular course?

The front-end application has always been designed with the intention of being a straightforward tool that accomplishes the required output. To simulate clinical trials, obtain the patients who were selected as well as their individual justifications. The focus of the interface has always been the question of simplicity.

Our application has to be able to let the user enter the desired clinical trial selection criteria, quickly obtain the selected patients, recognise each patient, and verify the appropriate justification for each selection.

The ability to save previously run trials was another feature that was considered and developed. Through a dashboard of saved trials, the user could load those criteria directly or even verify which patients were selected for each saved trial rather than constantly having to manually enter everything. When a trial is saved, it saves not only the criteria, but also the patients who were selected and their justifications.

6.2 Interface

Since it is typically simpler to manage everything through a computer and not a smartphone, we decided to develop a web application rather than a mobile one. The entire application interface was developed in React [25]. We chose to use React to develop the web application because it is quite efficient, supports front-end libraries that help to build rich user interfaces, offers fast rendering and it has a strong community. Moreover, it is a popular and widely-used front-end development framework.

The user interface design is simple and very easy to use. The important thing was to ensure that the application would be very intuitive, easy to use regardless of the experience that the user has. The user does not have to install anything because the setup is already done and embedded inside the application. To begin working and conducting clinical trials, the user merely needs to open the application in his browser.

This actually speeds up the entire patient selection process because it avoids the need to manually browse through each patient to find a potential candidate. The user only needs to enter the clinical trial criteria, which are also pretty straightforward since they are based only on clicks, and the application takes care of everything in a brief amount of time.

Several examples are presented in section 6.4 together with a user manual regarding the developed application and its features.

6.3 Process description

To supplement the presented system architecture and implementation in Figure 4.1, the flow diagram in Figure 6.1 depicts all of the processes from the start of a new trial to the conclusion of it.

[25] https://reactjs.org/

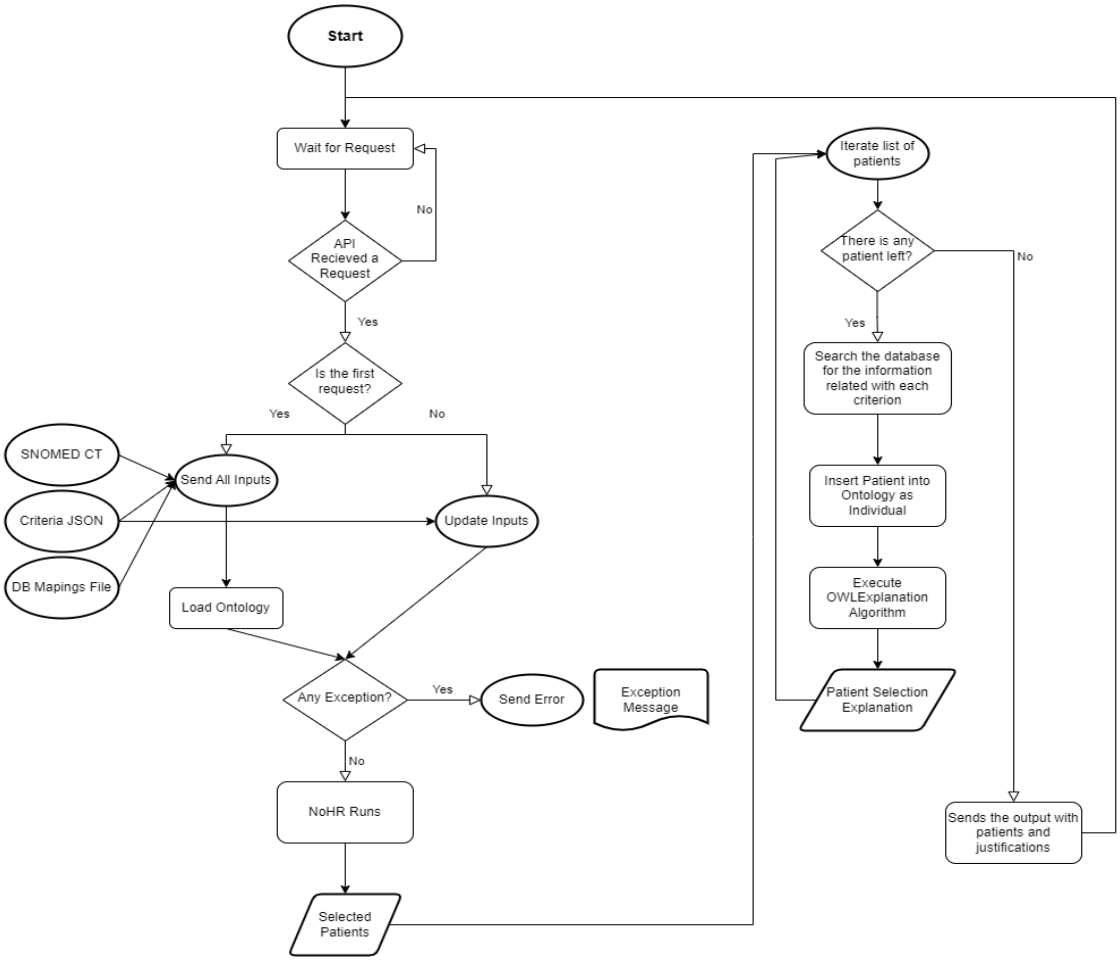

Figure 6.1: Flux Diagram

The application waits until it receives a request from a user to start a new trial. After receiving a request, it verifies whether it was the first request since the application was launched or not. If it is indeed the first request, then the back-end loads the SNOMED-CT ontology and the other inputs (rules and mappings files), otherwise we just need to load the new criteria (rules file). This is a good feature because it is not necessary to load the ontology whenever there is a new trial, as it is static, just load it on the first request and then from trial to trial only the criteria needs to be loaded.

NoHR will be responsible for processing the data and returning the selected patients, keeping in mind that the input will always be correct because the user will only be able to enter permitted criteria. NoHR returns a list of IDs that are the selected patients and the back-end later returns the intended information for each selected patient. Bearing in mind that the patients' important information to be highlighted is only their Id, first and last name, age and gender. Only that information is filtered to be presented. If for some reason some request, in this case some input was badly formatted, an exception would be thrown and the program would return to the initial state. However, such errors are

not expected because the only inputs that the user can change are the criteria, and these criteria can only be those provided by the application.

The final step of the algorithm is the generation of justifications for why each patient was selected. In the preceding chapter, this algorithm was presented and described in detail, along with all adaptations made and some back-end testing examples.

6.4 User manual

This section contains a number of usage examples and explanations for the application. Despite the fact that we believe it to be simple and intuitive, our user manual assures that all the features are clarified and well defined in case of any doubts.

6.4.1 Login

Since the program it is still an early version of a potential future application, only two valid credentials were created to test the application, in order to move smoothly to the dashboard screen. The application can only be used by users who have access to these credentials. This is primarily due to concerns about security and confidentiality. Therefore, we do not offer the possibility of new registrations. Figure 6.2 represents how the login was implemented.

In attachment I, we display the message that appears in case of successful login and in case the username or password is wrong, the error message that appears.

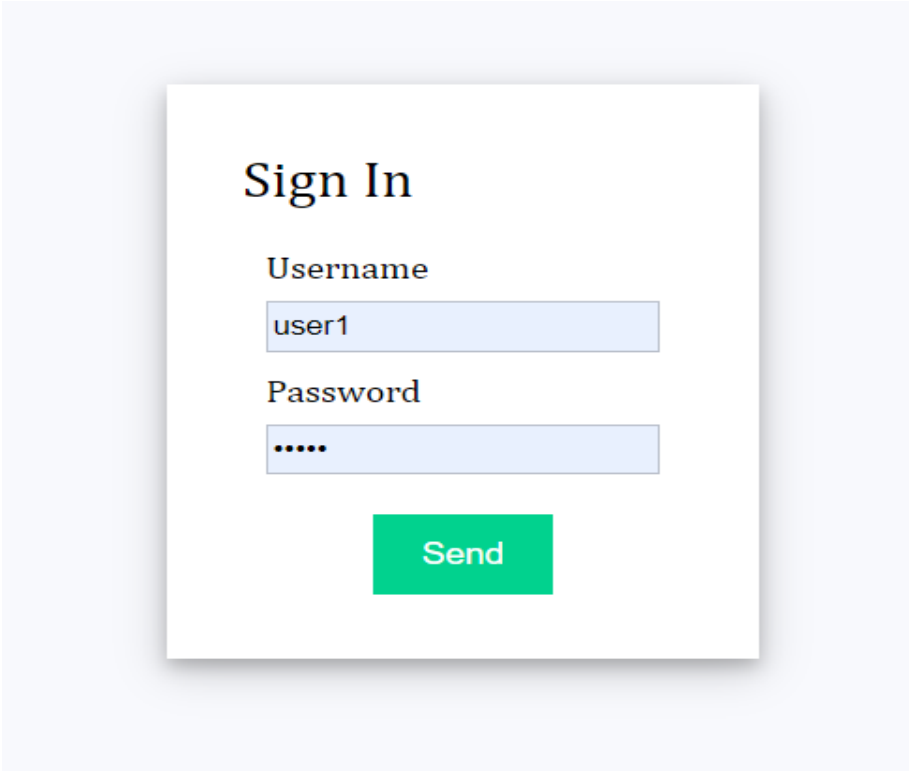

Figure 6.2: Login Screen

6.4.2 Dashboard

After successfully logging in, on the dashboard screen the user can see the trials that have been saved (if any) and also start a new trial. In Figure 6.3, we can see how the dashboard is presented to the user.

If the user has saved any trials, he can observe the criteria for each trial, obtain which patients have been selected without having to run the entire program again, and even load these criteria which will change to the screen where the criteria are inserted. In this way the user does not need to enter each criterion by hand if he just wants to edit some criterion of a previously done trial.

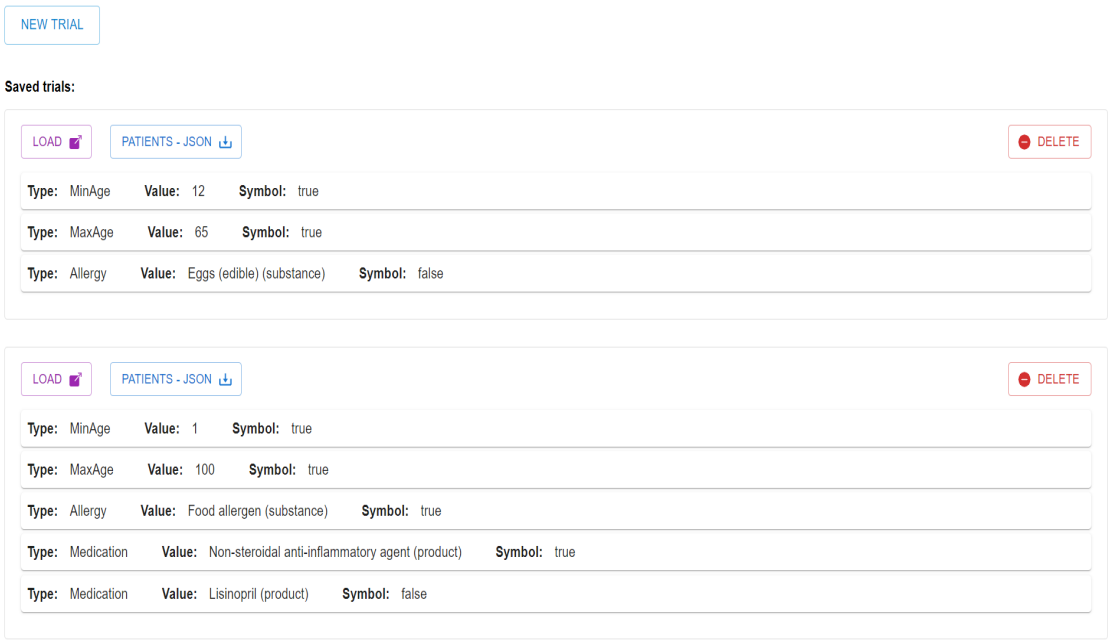

Figure 6.3: Dashboard Screen

6.4.3 Trial - Criteria Insertion

On this screen, the user inputs the trial-specific criteria that is required.

The user may load a previously stored trial that he wants to modify or proceed from, or it may be a brand-new trial. On this screen, represented in Figure 6.4, the user can click the *Add* button to add a new criterion, or if the user wants to remove or edit any of the criteria. After entering the desired criteria, the user must click *submit* to send the criteria, followed by the *Get Patients* button to obtain the results of the selected patients for the entered criteria.

The criteria accepted by the application are those previously mentioned:

- **MinAge:** the minimum age required for a user to participate in the trial.
- **MaxAge:** the maximum age allowed for a user to participate in the trial.

- **Gender:** if specified, it is because only male-only or female-only patients are accepted. When not specified, it can be male or female.

- **Allergies:** in this type of criterion, we may want patients with a certain allergy or patients without a certain allergy. In this type of criterion there is a choice of positive or negative symbol.

- **Medications:** similar to allergies here it can also be the case where we want to select patients who have taken a certain medication or who have not consumed a certain medication.

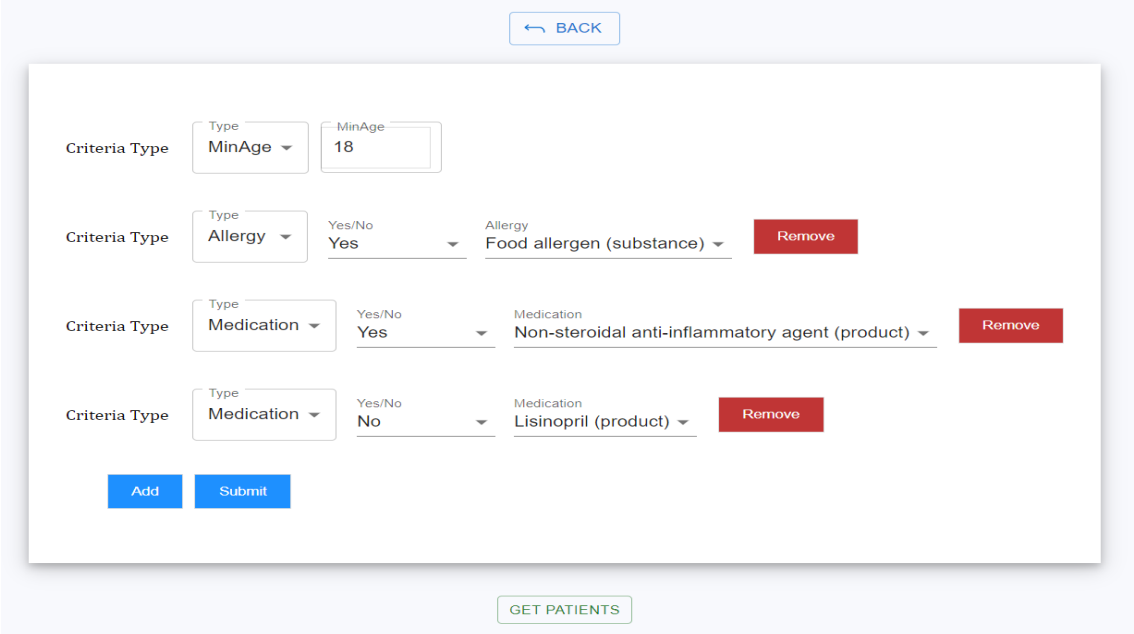

Figure 6.4: Trial Criteria Screen

6.4.4 Result - Selected Patients

Finally, the screen where the results of the selected patients from each trial will appear.

The following details are provided for each selected patient: his ID, first and last name, age, gender, and lastly the justification of why each one was selected. Figure 6.5 provides an example where the justifications are simpler and more readable and it was not necessary to run the algorithm that searches for information in the SNOMED-CT ontology, as this information for these criteria was in the database for each patient.

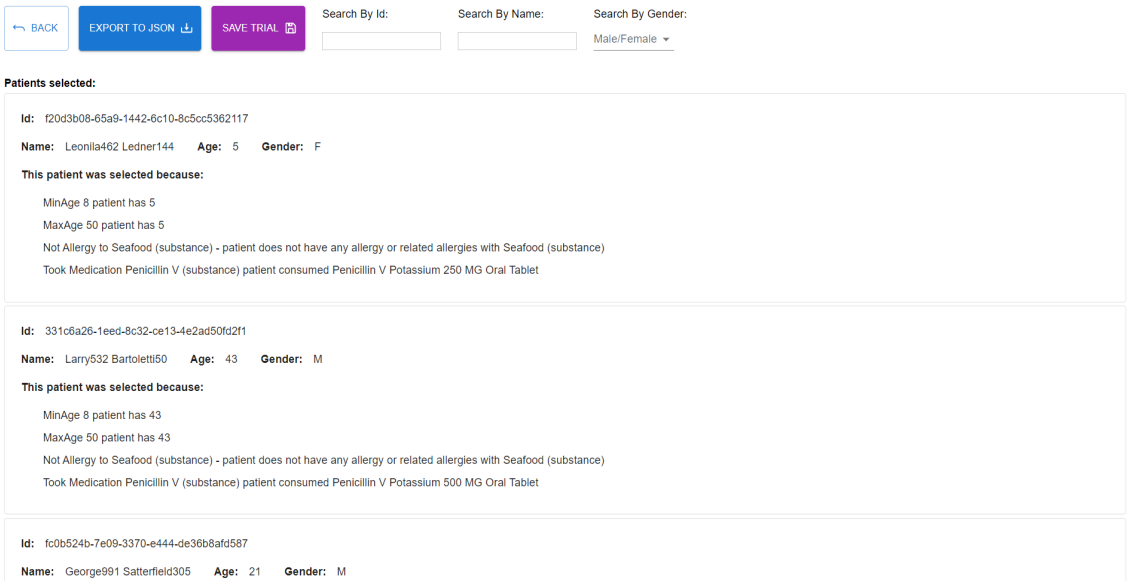

Figure 6.5: Selected Patients - Simple Justifications

On the other hand, Figure 6.6 is an example where the justification for the selection in relation to the criteria is now more complex, requiring the algorithm to search for the relationships of the classes in the ontology.

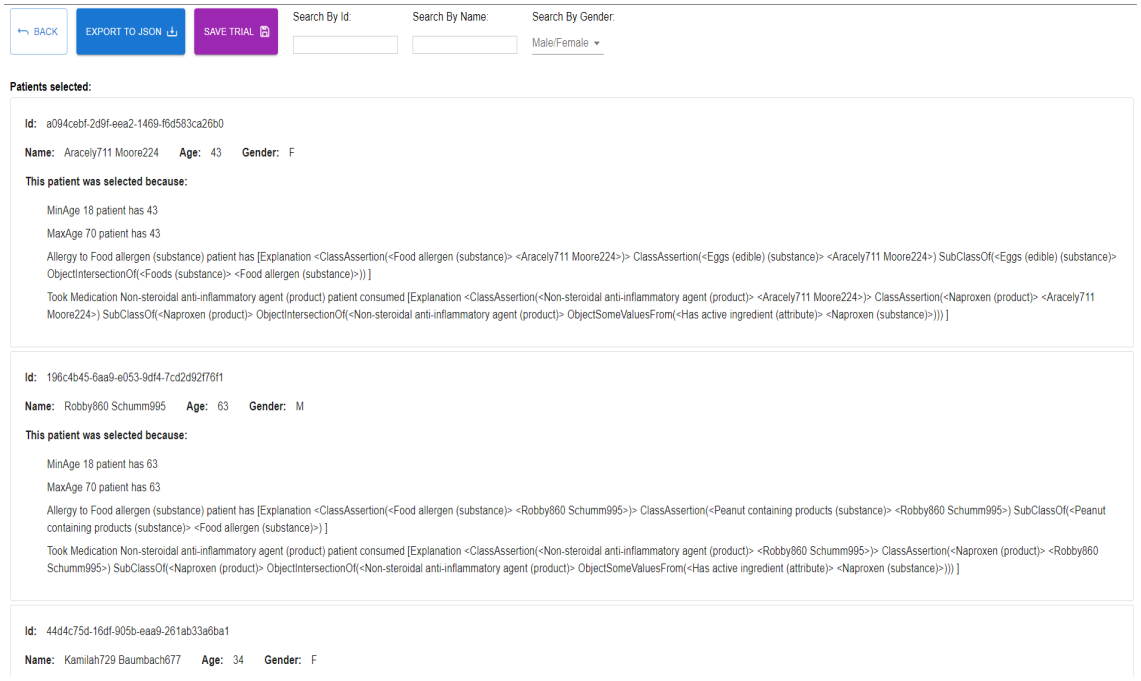

Figure 6.6: Selected Patients - Complex Justifications

After each trial, when redirected to this screen, the patient has some additional features in addition to observing the selected patients: i) the user can filter the selected patients by id, by name or by gender to search for a specific patient; it is possible to save the current trial which will later appear on the user's dashboard with information regarding the criteria of this trial and which patients were selected; iii) and he can also simply return to the dashboard screen by clicking the back button.

Some examples of filtering patients by Id, name, and gender are shown in attachment II.

7

Evaluation

In this chapter, we present time comparisons for various tests with SNOMED-CT ontology as input. We continue to compare the times for this ontology and one with fewer classes to examine how the number of ontology classes affected the ontology loading.

7.1 Validation

At an earlier stage, everything was evaluated on the command line or with an Integrated Development Environment (IDE) and the input criteria (rule file) were simulated. In the final stage, it was tested using the web application described in chapter 6.

During the entire implementation, the Protégé tool was fairly helpful in this subject of carrying out a second validation of all the results that were achieved throughout the development of the application. It served as a means of verifying that the outcomes were accurate and consistent. This means that, in order to ensure that the patients selected were correct and that the application was working as intended, we also tested and compared what was the results by using the NoHR plugin on the Protégé for the same query, inserting the rules, mappings and SNOMED-CT ontology.

Another method used to validate certain tests was to check the database to see if certain selected patients actually met the criterion. One example would be to determine whether or not the patients who were selected had a specific allergy. This was accomplished by running some MySQL queries to verify that the selected patients actually had that information.

In the following sections, the results regarding the performance of the application are presented using the original SNOMED-CT ontology and later a reduced version of this ontology.

We performed the tests on Intel Core i5-10500H processor @ 2.5GHz and 16GB 3200MHz DDR4 of memory running Windows 11. Our Java processes were given a maximum heap size of 8GB and Java version 8 was used. For these tests, we used the database developed in MySQL. The initial database used for the first tests contained 63

patients, 34 allergies, 3961 medications. The remaining tables created, which we already mentioned, do not have a great impact on these tests, taking into account that criteria that would resort to these tables were not used. However this database contained 11 tables, 1 view and occupies approximately 8.5 MB. Later, in section 7.4 we used a much larger database that contains 5492 patients, 3745 allergies, 254659 medications and occupies about 110MB of memory.

The results shown are based on five samples and all were obtained using the web application with requests to the back-end.

7.2 Performance evaluation with SNOMED-CT ontology

In this section, the times of five tests performed using the SNOMED-CT ontology were compared. This version of the SNOMED-CT ontology has 299239 classes and 1355009 axioms. The file occupies about 106MB.

Table 7.1 presents the times for each of run. The second row presents the times when the application was run for the first time. The third column depicts the times when the application had already been launched and the ontology had already been loaded, thus it was not necessary to reload the ontology, just reloaded the new criteria.

We can observe that when we run the application for the first time and we need to load the SNOMED-CT ontology, the average time is around 1716 seconds which is close to 29 minutes. This is a very acceptable time considering that currently, the manual process takes days and this time only happens the first time the ontology is loaded. In the last column, the average time is about 73 seconds, and this value corresponds to just over 1 minute. This means that after the ontology has been loaded, all trials will take about 1 to 2 minutes.

Sample nº	**Time (s) - First run**	**Time (s) - After first run**
Sample 1	1722	64
Sample 2	1739	69
Sample 3	1669	84
Sample 4	1647	83
Sample 5	1803	67

Table 7.1: Application run times using the SNOMED-CT ontology

To achieve these outcomes, we evaluated each sample according to the following criteria:

- **Samples 1 and 2:**
  - Min Age: 14
  - Max Age: 70
  - Gender: Male

  - Not have an allergy to Soya bean (substance)
  - Not have an allergy to Latex (substance)
  - Not have an allergy to Pollen allergen (substance)
  - Not have taken the medication Lisinopril (product)
  - Have taken the medication Ibuprofen (product)
- **Samples 3 and 4:**
  - Min Age: 18
  - Max Age: 75
  - Have an allergy to Food allergen (substance)
  - Have taken the medication Non-steroidal anti-inflammatory agent (product)
  - Have taken the medication Lisinopril (product)
- **Sample 5:**
  - Min Age: 2
  - Max Age: 100
  - Gender: Male
  - Not have an allergy to Eggs (edible) (substance)
  - Not have an allergy to Food allergen (substance)
  - Not having taken the medication Lisinopril (product)
  - Have taken the medication Non-steroidal anti-inflammatory agent (product)

Of course these times can increase if the number of trial criteria increases and the number of patients, number of allergies, medications and so on increase in the database. However, we do not expect huge time increases as these times are quite good when comparing minutes with days which is what trials normally take. In section 7.4 we present the results of testing with a much larger database, in which it would be practically impossible, if not impossible, for a human to find the right patients among so many patients, allergies, and medications in the database.

7.3 Performance between ontologies of different dimensions

We presented the times of five tests performed using the original SNOMED-CT ontology in the previous section. We reduced this ontology for the following tests, removing classes that were never used based on the criteria implemented. The SNOMED-CT ontology now contains only 90769 classes and 425387 axioms, and the file is approximately 47MB in size. One can conclude from these values that there was a significant reduction in the number of classes and axioms, as well as a reduction in file size by more than half.

Table 7.2 presents the times for each of run similar to the previous table. We can see that there is a significant time improvement when compared to the values shown in the previous section. This is because the original ontology had many classes that were never used but ended up greatly increasing the algorithm's temporal complexity, either when loading the ontology or later when searching for justifications that validate the patient's selection. Despite a reduction in ontology of approximately 60%, the times demonstrate an improvement of approximately 98%. This is primarily due to the fact that the original ontology significantly overloaded the memory of the machine on which the tests were conducted, resulting in a relatively slow process. Because the machine's memory was not overloaded after the ontology was reduced, the algorithm that loaded the ontology was quite fast.

Sample nº	**Time (s) - First run**	**Time (s) - After first run**
Sample 1	22	31
Sample 2	23	31
Sample 3	25	36
Sample 4	25	34
Sample 5	24	33

Table 7.2: Application run times using a reduced version of SNOMED-CT ontology

Additionally, we were able to observe an occurrence that may appear unusual. Contrary to expectations, the performance of the first run is better compared with the subsequent runs. As from the second run onwards it is necessary to reload the new criteria and update the program with the correct rules, it is expected that this update will be performed; therefore, there will be a small delay that can be observed due to the short times. Without the waiting time, the program would continue and, would not always update all of the criteria correctly, producing incorrect results.

We previously described the criteria used for each sample; for these results, we applied the same criteria.

Samples 1 and 2 were performed based on eight inserted criteria, none of which generated complex justifications, which means that it was not necessary to resort to ontology to present valid justifications.

Samples 3 and 4 were derived from trials that used five criteria. Although these times do not differ significantly when comparing with samples 1 and 2, the last column shows that it took a few seconds longer than samples 1 and 2, despite using fewer criteria. This is because the inserted criteria for samples 3 and 4 generate more complex justifications, requiring the use of ontology to obtain them. The algorithm that generates the justifications, as explained in a previous chapter, when it is necessary to resort to the ontology, it is necessary to consult the database, in order to later associate the individuals to the respective classes in the ontology so that the justifications can then be searched. Thus, the time is slightly higher.

Similarly, sample 5 was performed based on seven criteria. The time of the last column for this sample is also slightly higher comparing with samples 1 and 2 for the same reason, we also had to resort to ontology to generate justifications for a criterion.

As a result, it is possible to observe that the times when this reduced version of the SNOMED-T ontology was used were really good. It is a significant improvement that was not expected to be so good. When running the application for the first time, the average time is approximately 24 seconds, and originally it was 1716 seconds. After the first time, the average time is 33 seconds when reloading only the new criteria and generating the justifications, and originally it was 73 seconds. The times for the third column have been reduced by more than half, although the ontology is no longer reloaded, it is always necessary to use the ontology either for the selection of patients or to generate justifications. Thus, with the reduction of the existing classes, the search times are lower, having thus obtained this excellent performance.

Overall, even without the ontology reduction, the times were quite promising. With these times, it would already be much better to opt for this application to select patients for the different clinical trials rather than manually searching for candidates. Not only because the application takes much less time than the manual process to select patients, but also because all selected patients have a justification that facilitates validation.

7.4 Performance using a larger database

Previously, we addressed the performance using a very small database, comparing the original SNOMED-CT version to a reduced version of that ontology. Now, in Table 7.3, we present the performance results obtained when using a database so large that manually searching for patients in a short period of time would be practically impossible. The shortened version of the SNOMED-CT ontology was still used in these tests.

Sample nº	**T(s) Patient Sel.**	**T(s) Justi.**	**T(s) Total**	**Nº Selected**
Sample 1 - First Run	8715	82	8797	344
Sample 1 - Second+[26]	8728	83	8811	344
Sample 2 - First Run	8690	81	8771	344
Sample 2 - Second +	8679	82	8761	344
Sample 3 - First Run	3049	162	3211	130
Sample 3 - Second +	3047	162	3209	130
Sample 4 - First Run	3043	151	3194	130
Sample 4 - Second +	3041	152	3193	130
Sample 5 - First Run	11474	24	11498	20
Sample 5 - Second +	11482	23	11505	20

Table 7.3: Application run times using a larger database

[26]This means that the results were obtained after running the program at least once. The ontology is no longer reloaded in this case.

The tests carried out, as shown in Table 7.3, are similar to those performed previously presented in Table 7.2. The big difference is the size of the database used. Now, the times are in fact much higher because the search for patients is not so simple due to the existing number of patients and the number of information related to each patient being much higher than the previous database, which was quite simple.

This table was built in a more detailed manner. We can see how long it took to select patients based on the criteria, how long it took to find all the justifications, the total time that is the sum of these two times, and the number of patients selected for this trial. The used samples are the same to those described in detail previously.

We can observe for each sample that whether the application is run for the first time or not, the time does not vary significantly because we are using the reduced version of the SNOMED-CT ontology. As displayed in Table 7.2, the difference in times between the first and subsequent runs is due to reloading the rules (new criteria) and justifications rather than loading the ontology. The time for the first run would be slightly longer with the initial version of SNOMED-CT ontology, as shown in Table 7.1.

Samples 1 and 2 were obtained using eight criteria that produced no complex justification. It took slightly less than 2.5 hours to select patients for these tests. This time is longer than those obtained previously, but the results are favorable to our application. This is because, for a database where it would take a long time to manually find all of the patients who met all of the criteria, this time is beneficial, as it would take several days manually. The algorithm took just over a minute to generate the justifications, which is quite fast. The number of responses (selected patients) was 344, which makes it possible to verify that even for a reasonable number of valid patients, the time it took to generate the justifications was relatively small, because none of the criteria required resorting to ontology. This number of selected patients complements the time ratio it took the algorithm to find all patients among thousands.

Five criteria were used to obtain the results of samples 3 and 4, two of which generated more complex justifications, and the algorithm had to traverse the ontology classes to generate the respective explanations. For these two tests, patient selection took less than an hour, which is excellent. The generation of justifications took less than 3 minutes, which is also great. We obtained 130 patients who met the criteria for these samples. Despite obtaining fewer patients than in samples 1 and 2, there were two criteria that ended up resorting to the ontology to generate justifications. It was necessary to proceed to the second step of the algorithm that generates the justifications and to search for relationships between the ontology classes and the information for each selected patient. These criteria were: i) Have an allergy to Food allergen (substance); ii) Have taken the medication Non-steroidal anti-inflammatory agent (product). As a result, it is evident that with fewer criteria, the time has decreased overall, but as the generation of justifications was already a little more complex, it took slightly longer to generate justifications.

In the case of sample 5, we have already passed 3 hours. Seven criteria were used for this sample, one of which generated complex justifications. The times increased

when compared to samples 1 and 2, which used eight simple criteria, and despite having reduced one criterion, we had a more complex criterion that consulted not only the database but also the ontology to find the right patients. Nonetheless, the time it took to generate the justifications was surprisingly short. This is because the number of patients selected for this sample was much smaller than the other samples. Thus, the algorithm did not have to find as many justifications

7.4.1 In-depth evaluation of a sample

Previously, the results of five samples had been presented. In this section, we will only consider samples 3 and 4, in which we will reduce and increase the number of criteria while maintaining the same number of criteria as a baseline when adding a new criterion. Remembering that samples 3 and 4 contained the following criteria:

- Min Age: 18
- Max Age: 75
- Have an allergy to Food allergen (substance)
- Have taken the medication Non-steroidal anti-inflammatory agent (product)
- Have taken the medication Lisinopril (product)

These criteria led to the selection of 130 patients, which took approximately 3045 seconds (approximately 51 minutes), and less than 3 minutes to generate all justifications.

Criteria used	**T(s) Patient Selection**	**T(s) Justifications**	**Nº Selected**
Original sample	3045	157	130
Four criteria	1639	204	384
Six criteria	3175	80	66
Eight criteria	3419	74	64

Table 7.4: Performance evaluation as the number of criteria changes

Table 7.4 enables us to verify the validity of our arguments. Starting with an analysis of the reduction to four criteria after the elimination of the *'Have taken the medication Non-steroidal anti-inflammatory agent (product)'* criterion, we were able to observe a significant reduction in the time required to select patients. The eliminated criterion was complex, always relying on ontology to identify possible selected patients and generate justifications. After removing that criterion, the number of patients selected increased from 130 to 384, which increased the time required to generate all justifications.

Adding criterion *'Gender: Male'* to the original sample, as shown in the next row of the table, resulted in a slight increase in the time required to select patients. In the previous case, removing one of the criteria significantly reduced the time required to select patients. In this instance, the addition of a criterion resulted in only a slight increase as the added criterion is relatively simple and only requires searching the database for male patients; consequently, the time required to generate justifications was reduced by nearly half, as was the number of selected patients.

To obtain the values of the final row, the following criteria were added to the original sample: i) *'Gender: Male'*; ii) *'Have an allergy to Mold (organism)'*; iii) *'Have an allergy to Pollen (substance)'*. In fact, the times obtained with this sample of eight criteria were quite good; this test did not exceed one hour. It was expected that the times would be higher than what was actually obtained. The conclusion that can be drawn is that there are certain types of criteria that make the patient selection considerably slower than others. This is due to the fact that either the criterion is more complex and need to resort to ontology, or the criterion contains negation symbol, which slows down the algorithm, or because there are more patients related to that criterion.

We can conclude that, despite the possibility of increasing the number of criteria implemented in the future and the size of the tests, using a reduced version of the ontology that excludes classes that may not be of interest speeds up the entire process. We believe that, regardless of the number of criteria used, the application will always return the results of patient selection and justifications in less than a day. And, to the best of our knowledge, this does not exist in the market.

Our results show that, while the time for the larger database has increased significantly, it is still quite acceptable because it would be practically impossible to find all the information regarding the criteria in such a way that patients who cross all the information intended are selected manually in the midst of thousands of patients and thousands of allergies and medications.

In the real world, there are several types of criteria that we did not implement and even larger databases, which would undoubtedly increase the times given the dimensions. However, if the goal is to actually extend the application, as previously stated, it is not expected that times will worsen to the point where it is no longer viable. This is due to the fact that, in that case, a much better machine than the one we used to carry out these tests would be used, which in itself would improve these times even more.

8

Conclusion

In this chapter, we will summarize all of the work done so far in this dissertation and how it was accomplished, as well as leave some suggestions for future improvements.

Clinical trials are common procedures before new drugs or new medical devices are released. Thus, making this process more and more efficient is in fact vital, not only to obtain faster results, but also to guarantee success by minimizing errors. As a result, the more automated this process is, the less effort the human must expend in finding the right candidates for each trial.

Nowadays, this process is usually manual, where someone in particular looks for patients who meet the necessary criteria to be fulfilled. So far as we know there is no tool, application or framework that is simple to use, regardless of the user's experience and knowledge, such that the patient selection process could be automated. Therefore, one of the main limitations related to this procedure is usability. This emphasizes the need for the development of a simple and intuitive application with these capabilities.

These limitations have been overcome during this dissertation following the development of the back-end application integrating NoHR, as well as the development of the web application. At this moment the user does not even need to do any set-up. The web application already provides everything the user needs to start new clinical trials in a simple and very intuitive way. In this regard, the first web application was created, which allows performing clinical trials and obtaining patients automatically, as well as obtaining the proper justifications for each selection, without any kind of experience and prior knowledge.

The outcome was quite encouraging. Based on the criteria (rules) entered, it was really possible to quickly obtain the appropriate patients for each test conducted to simulate a clinical trial. As we mentioned, after the improvements using the original SNOMED-CT ontology, it only took around 28 minutes to load the full ontology the first time, and then it usually takes no more than 2 minutes to select the patients and present the appropriate justifications for each patient in the following trials. This is assuming that the server runs once and then is always active. Subsequently, it was demonstrated that when the ontology

was reduced, the times significantly decreased, and this was with a very small database. When simulating more realistic trials using a larger database, the time required to obtain the correct patients and present the justifications increased significantly, surpassing 3 hours. However, as we also mentioned, these are quite good performances given that the process is lengthy and requires several days to complete manually.

Several challenges had to be overcome during the development of this application, such as the generation of justifications and adaptations to improve the question of their readability. This, in our opinion, has been accomplished successfully. The explanations generated are frequently perceptible and aid in validating the selection of each patient. As we previously demonstrated in Figure 6.6, justifications for certain types of criteria may at times be somewhat more extensive. They end up requiring a more detailed reading and attempting to comprehend the relationship between the criterion and the patient's information. However, this is not a significant issue, as we believe they are still perceptible despite demanding some extra attention.

Moreover, we had to ensure that the front-end criteria entered were correctly passed to the back-end in the correct manner so that it could be converted into a file of rules as input. This was also accomplished by analyzing the corresponding SNOMED-CT ontology terms and the criteria we wanted to make available. Each criterion's name had to be the exactly the same as the corresponding one in the ontology and the user cannot enter criteria invented by him. Only the criteria allowed by the application through the drop-down menus are allowed, to ensure that there are no errors. Otherwise, the back-end could not select the correct patients. This is because we could be creating criteria that seemed to be correct but would actually be wrong according to what would be intended, as they would not be in accordance with the terms of the SNOMED-CT ontology. A deeper analysis of the ontology, revealed that there were indeed some terms that partly contain the same name as other terms but then differ in some ways. So it was really necessary to be careful about the full name of each criterion entered.

All things considered, it is clear that although there is still a lot of work to be done to make it stable and used globally, the point in favor of this entire application is that it has truly demonstrated that automating the entire patient selection process is indeed possible with valid justifications in a relatively short period of time compared to the manual procedure. This could have a significant impact in real life because it would not only reduce human errors, but it would also ensure that deadlines are met, thereby increasing the success of the various studies.

8.1 Future Work

We believe that this work opens the door to the potential of automating clinical trial processes and it provides a huge ground to explore and improve.

To begin with, expanding the amount of existing criteria in the application would benefit the users who are working on clinical trials studies. This expansion would not require such a high complexity, it would only be necessary to analyze the criteria that would make sense to add, and verify the correct terms according to the SNOMED-CT ontology and the database. It would also be required to create the corresponding mappings for each criterion and if necessary extend the database by creating new tables.

In addition to this expansion, although not extremely critical, it would be interesting to analyze the temporal complexity of the original SNOMED-CT ontology implies and try to improve in this aspect. This only has a greater impact the first time the application runs the first trial, but whenever the need for maintenance arises, it is in turn necessary to restart the entire application. We got good results when we reduced the size of this ontology, but we believe that further optimizations can be made without modifying the original ontology. Furthermore, we believe that there may be some optimizations regarding the use of a higher-dimensional database, as previously presented, where better performances would undoubtedly be achieved.

Also, the algorithm could be updated so that each patient's justifications can be computed in parallel. This would be a step towards optimizing computing efficiency. Given that the algorithm's performance was already good enough, the primary focus was on producing readable justifications.

The justifications' algorithm occasionally yields explanations that are more difficult to comprehend and read. Thinking about solutions to produce them easier to read without creating major difficulty with terms that may be unfamiliar would be quite interesting. In this regard, it would be advantageous to develop an algorithm or update the existing one so that the generated justifications attempt to be more linear. This means that instead of, for example, using various names of ontology classes and sub-relations to justify something, one could use a sentence that is more concise and considerably more readable, such as: *The patient was selected because he had taken medication A, whereas all volunteers were required to have taken medication Z. This patient meets this criterion because medication A is related to medication Z.* Thus, it would be possible to simplify certain ontology-related justifications without the use of unfamiliar terms.

Finally, the web application design could be greatly improved to a more modern design. As we mentioned earlier, the focus has always been on the simplicity of the application without great difficulties in using it, regardless of the user's previous knowledge. However, the current design is not modern and elegant as it could be. Obviously, it is not essential or a top priority. It is far more important to have the application work correctly for new criteria, achieve good performance, and ensure that there are no usability issues than to have a modern and robust design. Nevertheless, there is still room for improvement.

Bibliography

[1] G. Antoniou et al. *A Semantic Web Primer, 3rd Edition*. MIT Press, 2012. ISBN: 978-0-262-01828-9 (cit. on p. 14).

[2] G. Araújo, A. Mourão, and J. Magalhães. "Learning to rank clinical trials with rule-based criteria". In: *Proceedings of the Twenty-Seventh Text REtrieval Conference, TREC 2018, Gaithersburg, Maryland, USA, November 14-16, 2018*. Ed. by E. M. Voorhees and A. Ellis. Vol. 500-331. NIST Special Publication. National Institute of Standards and Technology (NIST), 2018. URL: https://trec.nist.gov/pubs/trec27/papers/NOVASearch-PM.pdf (cit. on p. 23).

[3] F. Baader, S. Borgwardt, and W. Forkel. "Patient Selection for Clinical Trials Using Temporalized Ontology-Mediated Query Answering". In: *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*. Ed. by P.-A. Champin et al. ACM, 2018, pp. 1069–1074. DOI: 10.1145/3184558.3191538. URL: https://doi.org/10.1145/3184558.3191538 (cit. on p. 22).

[4] F. Baader et al., eds. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003. ISBN: 0-521-78176-0 (cit. on p. 12).

[5] J. de Bruijn et al. "OWL DL vs. OWL flight: conceptual modeling and reasoning for the semantic Web". In: *Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005*. Ed. by A. Ellis and T. Hagino. ACM, 2005, pp. 623–632. DOI: 10.1145/1060745.1060836. URL: https://doi.org/10.1145/1060745.1060836 (cit. on p. 15).

[6] N. Costa, M. Knorr, and J. Leite. "Next Step for NoHR: OWL 2 QL". In: *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*. Ed. by M. Arenas et al. Vol. 9366. Lecture Notes in Computer Science. Springer, 2015, pp. 569–586. DOI: 10.1007/978-3-319-25007-6_33. URL: https://doi.org/10.1007/978-3-319-25007-6_33 (cit. on p. 18).

[7] C. V. Damásio, A. Analyti, and G. Antoniou. "Justifications for Logic Programming". In: *Logic Programming and Nonmonotonic Reasoning, 12th International Conference, LPNMR 2013, Corunna, Spain, September 15-19, 2013. Proceedings*. Ed. by P. Cabalar and T. C. Son. Vol. 8148. Lecture Notes in Computer Science. Springer, 2013, pp. 530–542. DOI: 10.1007/978-3-642-40564-8_53. URL: https://doi.org/10.1007/978-3-642-40564-8_53 (cit. on pp. 26, 27).

[8] M. Desai. "Recruitment and retention of participants in clinical studies: Critical issues and challenges". In: *Perspectives in Clinical Research* 11.2 (2020), pp. 51–53. DOI: 10.4103/picr.PICR_6_20. URL: https://www.picronline.org/article.asp?issn=2229-3485;year=2020;volume=11;issue=2;spage=51;epage=53;aulast=Desai;t=6 (cit. on pp. 1, 21).

[9] W. Drabent, J. Henriksson, and J. Maluszynski. "HD-rules: A Hybrid System Interfacing Prolog with DL-reasoners". In: *Proceedings of the ICLP'07 Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services, ALPSWS 2007, Porto, Portugal, September 13th, 2007*. Ed. by A. Polleres et al. Vol. 287. CEUR Workshop Proceedings. CEUR-WS.org, 2007. URL: http://ceur-ws.org/Vol-287/paper%5C_9.pdf (cit. on p. 4).

[10] T. Eiter et al. "Rules and Ontologies for the Semantic Web". In: *Reasoning Web, 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*. Ed. by C. Baroglio et al. Vol. 5224. Lecture Notes in Computer Science. Springer, 2008, pp. 1–53. DOI: 10.1007/978-3-540-85658-0_1. URL: https://doi.org/10.1007/978-3-540-85658-0_1 (cit. on pp. 8, 13).

[11] J. Fandinno and C. Schulz. "Answering the "why"in Answer Set Programming - A Survey of Explanation Approaches". In: *CoRR* abs/1809.08034 (2018). arXiv: 1809.08034. URL: http://arxiv.org/abs/1809.08034 (cit. on p. 24).

[12] E. Fink et al. "Selection of patients for clinical trials: an interactive web-based system". In: *Artif. Intell. Medicine* 31.3 (2004), pp. 241–254. DOI: 10.1016/j.artmed.2004.01.017. URL: https://doi.org/10.1016/j.artmed.2004.01.017 (cit. on pp. 1, 2).

[13] J. H. Gennari et al. "The evolution of Protégé: an environment for knowledge-based systems development". In: *Int. J. Hum. Comput. Stud.* 58.1 (2003), pp. 89–123. DOI: 10.1016/S1071-5819(02)00127-1. URL: https://doi.org/10.1016/S1071-5819(02)00127-1 (cit. on p. 16).

[14] C. Golbreich and E. K. Wallace. *OWL 2 Web Ontology Language New Features and Rationale*. 2nd ed. 2012. URL: https://www.w3.org/TR/owl2-new-features (cit. on p. 11).

[15] P. Hayes. *RDF Semantics*. 2004. URL: https://www.w3.org/TR/rdf-mt/ (cit. on p. 9).

[16] I. Horrocks, O. Kutz, and U. Sattler. "The even more irresistible $\mathcal{SROIQ}$". In: 2006-01, pp. 57–67 (cit. on p. 13).

[17] V. Ivanov, M. Knorr, and J. Leite. "A Query Tool for *EL* with Non-monotonic Rules". In: *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*. Ed. by H. Alani et al. Vol. 8218. Lecture Notes in Computer Science. Springer, 2013, pp. 216–231. DOI: 10.1007/978-3-642-41335-3_14. URL: https://doi.org/10.1007/978-3-642-41335-3_14 (cit. on pp. 4, 17, 18).

[18] A. Kalyanpur. "Debugging and Repair of OWL Ontologies". PhD thesis. University of Maryland, College Park, MD, USA, 2006. URL: http://hdl.handle.net/1903/3820 (cit. on p. 25).

[19] A. Kalyanpur et al. "Finding All Justifications of OWL DL Entailments". In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*. Ed. by K. Aberer et al. Vol. 4825. Lecture Notes in Computer Science. Springer, 2007, pp. 267–280. DOI: 10.1007/978-3-540-76298-0_20. URL: https://doi.org/10.1007/978-3-540-76298-0_20 (cit. on pp. 4, 24, 25, 41).

[20] V. Kasalica et al. "NoHR: An Overview". In: *Künstliche Intell.* 34.4 (2020), pp. 509–515. DOI: 10.1007/s13218-020-00650-1. URL: https://doi.org/10.1007/s13218-020-00650-1 (cit. on p. 4).

[21] V. Kasalica et al. "Telco Network Inventory Validation with NoHR". In: *Logic Programming and Nonmonotonic Reasoning - 15th International Conference, LPNMR 2019, Philadelphia, PA, USA, June 3-7, 2019, Proceedings*. Ed. by M. Balduccini, Y. Lierler, and S. Woltran. Vol. 11481. Lecture Notes in Computer Science. Springer, 2019, pp. 18–31. DOI: 10.1007/978-3-030-20528-7_2. URL: https://doi.org/10.1007/978-3-030-20528-7_2 (cit. on pp. 18, 19).

[22] Y. Kazakov, P. Klinov, and A. Stupnikov. "Towards Reusable Explanation Services in Protege". In: *Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017*. Ed. by A. Artale, B. Glimm, and R. Kontchakov. Vol. 1879. CEUR Workshop Proceedings. CEUR-WS.org, 2017. URL: http://ceur-ws.org/Vol-1879/paper31.pdf (cit. on pp. 25, 26, 28).

[23] M. Kifer. "Rule Interchange Format: The Framework". In: *Web Reasoning and Rule Systems, Second International Conference, RR 2008, Karlsruhe, Germany, October 31-November 1, 2008. Proceedings*. Ed. by D. Calvanese and G. Lausen. Vol. 5341. Lecture Notes in Computer Science. Springer, 2008, pp. 1–11. DOI: 10.1007/978-3-540-88737-9_1. URL: https://doi.org/10.1007/978-3-540-88737-9_1 (cit. on p. 15).

[24] D. Kless and L. Jansen. "How fit is OWL to represent realist ontologies? The semantics of representational units in realist ontologies and the Web Ontology Language". In: *43. Jahrestagung der Gesellschaft für Informatik, Informatik angepasst an Mensch, Organisation und Umwelt, INFORMATIK 2013, Koblenz, Germany, September 16-20, 2013*. Ed. by M. Horbach. Vol. P-220. LNI. GI, 2013, pp. 1851–1880. URL: https://dl.gi.de/20.500.12116/20617 (cit. on p. 9).

[25] M. Knorr. "On Combining Ontologies and Rules". In: *Reasoning Web*. Vol. 13100. Lecture Notes in Computer Science. Springer, 2021, pp. 22–58 (cit. on pp. 2, 3, 12–15, 17, 19).

[26] J. Kola, W. K. Wong, and B. Buddala. "Making Clinical Trials available at the Point of Care - connecting Clinical Trials to Electronic Health Records using SNOMED CT and HL7 InfoButton Standards". In: *12th International Conference on Semantic Web Applications and Tools for Health Care and Life Sciences*. Ed. by R. Cornet and A. Waagmeester. Vol. 2849. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 54–63. URL: http://ceur-ws.org/Vol-2849/paper-07.pdf (cit. on p. 22).

[27] A. Krisnadhi, F. Maier, and P. Hitzler. "OWL and Rules". In: 2011-08, pp. 382–415. ISBN: 978-3-642-23031-8. DOI: 10.1007/978-3-642-23032-5_7 (cit. on pp. 13, 17).

[28] M. Krötzsch. "OWL 2 Profiles: An Introduction to Lightweight Ontology Languages". In: *Reasoning Web. Semantic Technologies for Advanced Query Answering - 8th International Summer School 2012, Vienna, Austria, September 3-8, 2012. Proceedings*. Ed. by T. Eiter and T. Krennwallner. Vol. 7487. Lecture Notes in Computer Science. Springer, 2012, pp. 112–183. DOI: 10.1007/978-3-642-33158-9_4. URL: https://doi.org/10.1007/978-3-642-33158-9_4 (cit. on p. 11).

[29] J. Lee and R. Palla. "Integrating Rules and Ontologies in the First-Order Stable Model Semantics (Preliminary Report)". In: *Logic Programming and Nonmonotonic Reasoning - 11th International Conference, LPNMR 2011, Vancouver, Canada, May 16-19, 2011. Proceedings*. Ed. by J. P. Delgrande and W. Faber. Vol. 6645. Lecture Notes in Computer Science. Springer, 2011, pp. 248–253. DOI: 10.1007/978-3-642-20895-9_27. URL: https://doi.org/10.1007/978-3-642-20895-9_27 (cit. on p. 3).

[30] J. Leveling. "Patient selection for clinical trials based on concept-based retrieval and result filtering and ranking". In: *Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017, Gaithersburg, Maryland, USA, November 15-17, 2017*. Ed. by E. M. Voorhees and A. Ellis. Vol. 500-324. NIST Special Publication. National Institute of Standards and Technology (NIST), 2017. URL: https://trec.nist.gov/pubs/trec26/papers/teckro-PM.pdf (cit. on p. 23).

[31] C. Lopes, M. Knorr, and J. Leite. "NoHR: Integrating XSB Prolog with the OWL 2 Profiles and Beyond". In: *Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings*. Ed. by M. Balduccini and T. Janhunen. Vol. 10377. Lecture Notes in Computer

Science. Springer, 2017, pp. 236–249. DOI: 10.1007/978-3-319-61660-5_22. URL: https://doi.org/10.1007/978-3-319-61660-5_22 (cit. on p. 18).

[32] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/master/template.pdf (cit. on p. ii).

[33] B. Motik et al. *OWL 2 Web Ontology Language Profiles*. 2nd ed. 2012. URL: https://www.w3.org/TR/2012/REC-owl2-profiles-20121211/ (cit. on p. 11).

[34] D. Nardi and R. J. Brachman. "An Introduction to Description Logics". In: *The Description Logic Handbook: Theory, Implementation, and Applications*. Ed. by F. Baader et al. Cambridge University Press, 2003, pp. 1–40 (cit. on p. 2).

[35] E. N. Ngayua, J. He, and K. Agyei-Boahene. "Applying advanced technologies to improve clinical trials: a systematic mapping study". In: *Scientometrics* 126.2 (2021), pp. 1217–1238. DOI: 10.1007/s11192-020-03774-1. URL: https://doi.org/10.1007/s11192-020-03774-1 (cit. on pp. 2, 21).

[36] C. Patel et al. "Matching Patient Records to Clinical Trials Using Ontologies". In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*. Ed. by K. Aberer et al. Vol. 4825. Lecture Notes in Computer Science. Springer, 2007, pp. 816–829. DOI: 10.1007/978-3-540-76298-0_59. URL: https://doi.org/10.1007/978-3-540-76298-0_59 (cit. on pp. 3, 21).

[37] E. Pontelli, T. C. Son, and O. El-Khatib. "Justifications for Logic Programs under Answer Set Semantics". In: *CoRR* abs/0812.0790 (2008). arXiv: 0812.0790. URL: http://arxiv.org/abs/0812.0790 (cit. on pp. 26, 27).

[38] D. L. Poole and A. K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. 2nd ed. Cambridge University Press, 2017. URL: http://artint.info/2e/html/ArtInt2e.html (cit. on p. 14).

[39] R. Reiter. "A Theory of Diagnosis from First Principles". In: *Artif. Intell.* 32.1 (1987), pp. 57–95. DOI: 10.1016/0004-3702(87)90062-2. URL: https://doi.org/10.1016/0004-3702(87)90062-2 (cit. on p. 25).

[40] R. Rosati. "Integrating Ontologies and Rules: Semantic and Computational Issues". In: *Reasoning Web, Second International Summer School 2006, Lisbon, Portugal, September 4-8, 2006, Tutorial Lectures*. Ed. by P. Barahona et al. Vol. 4126. Lecture Notes in Computer Science. Springer, 2006, pp. 128–151. DOI: 10.1007/11837787_5. URL: https://doi.org/10.1007/11837787_5 (cit. on p. 15).

[41] G. Schreiber and Y. Raimond. *RDF 1.1 Primer*. 2014-06. URL: https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/ (cit. on pp. 9, 10).

[42] H. Uesaka. "Clinical Trials: An Overview". In: *International Encyclopedia of Statistical Science*. Ed. by M. Lovric. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 256–259. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_11. URL: https://doi.org/10.1007/978-3-642-04898-2_11 (cit. on p. 1).

[43] C. N. Vorisek et al. "Evaluating Suitability of SNOMED CT in Structured Searches for COVID-19 Studies". In: *Public Health and Informatics - Proceedings of MIE 2021, Medical Informatics Europe, Virtual Event, May 29-31, 2021*. Ed. by J. Mantas et al. Vol. 281. Studies in Health Technology and Informatics. IOS Press, 2021, pp. 88–92. DOI: 10.3233/SHTI210126. URL: https://doi.org/10.3233/SHTI210126 (cit. on p. 24).

[44] J. Walonoski et al. "Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record". In: *Journal of the American Medical Informatics Association* 25.3 (2017-08), pp. 230–238. ISSN: 1527-974X. DOI: 10.1093/jamia/ocx079. eprint: https://academic.oup.com/jamia/article-pdf/25/3/230/34150150/ocx079.pdf. URL: https://doi.org/10.1093/jamia/ocx079 (cit. on p. 31).

[45] *What Are Clinical Trials and Studies?* https://www.nia.nih.gov/health/what-are-clinical-trials-and-studies (cit. on p. 2).

[46] J. Wiedekopf et al. "Desiderata for a Synthetic Clinical Data Generator". In: vol. 281. 2021-05. ISBN: 9781643681849. DOI: 10.3233/SHTI210122 (cit. on p. 31).

[47] G. Xiao, T. Eiter, and S. Heymans. "The DReW System for Nonmonotonic DL-Programs". In: *Semantic Web and Web Science - 6th Chinese Semantic Web Symposium and 1st Chinese Web Science Conference, CSWS 2012, Shenzhen, China, November 28-30, 2012*. Ed. by J. Li et al. Springer, 2012, pp. 383–390. DOI: 10.1007/978-1-4614-6880-6_33. URL: https://doi.org/10.1007/978-1-4614-6880-6%5C_33 (cit. on p. 4).

[48] S. Zillner and D. Sonntag. "Aligning medical ontologies by axiomatic models, corpus linguistic syntactic rules and context information". In: *2011 24th International Symposium on Computer-Based Medical Systems (CBMS)*. 2011, pp. 1–6. DOI: 10.1109/CBMS.2011.5999162 (cit. on p. 3).

I

Annex 1 - Login Images

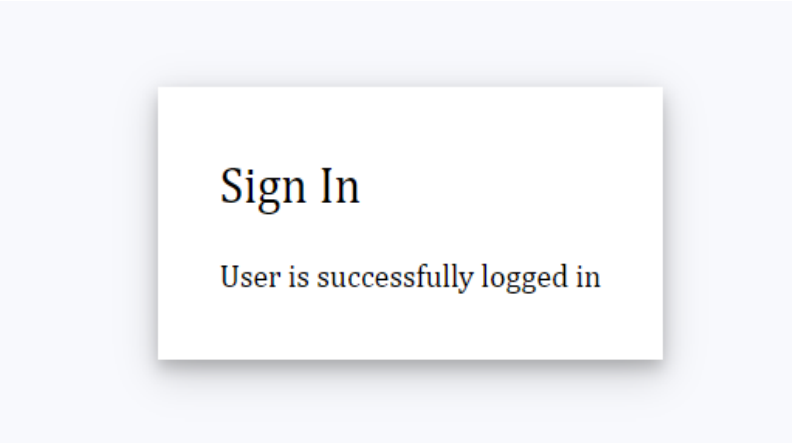

Figure I.1: Login with sucess

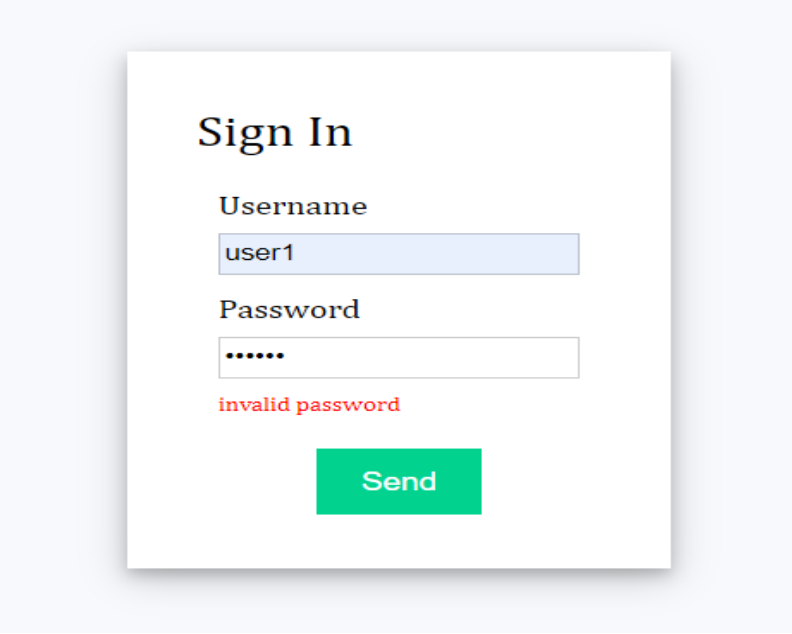

Figure I.2: User or password wrong on Login

II

Annex 2 - Selected Patients Filters

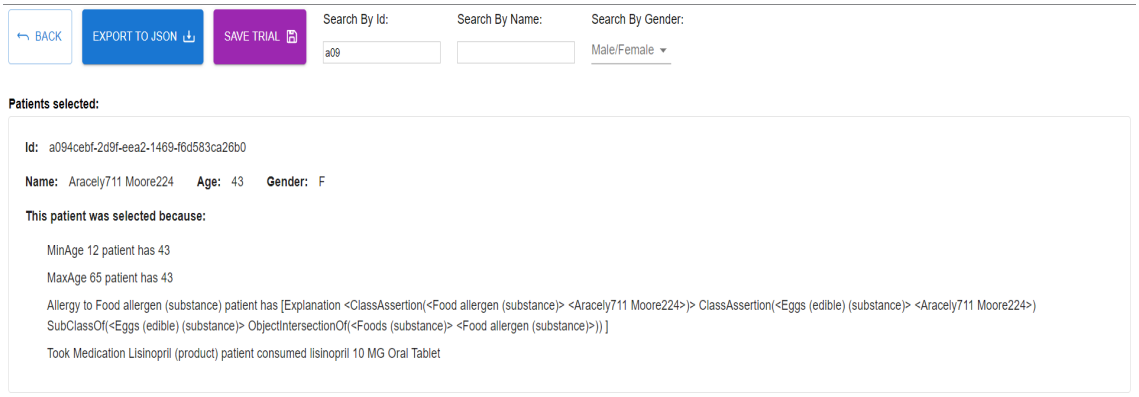

Figure II.1: Selected Patients with filter by id

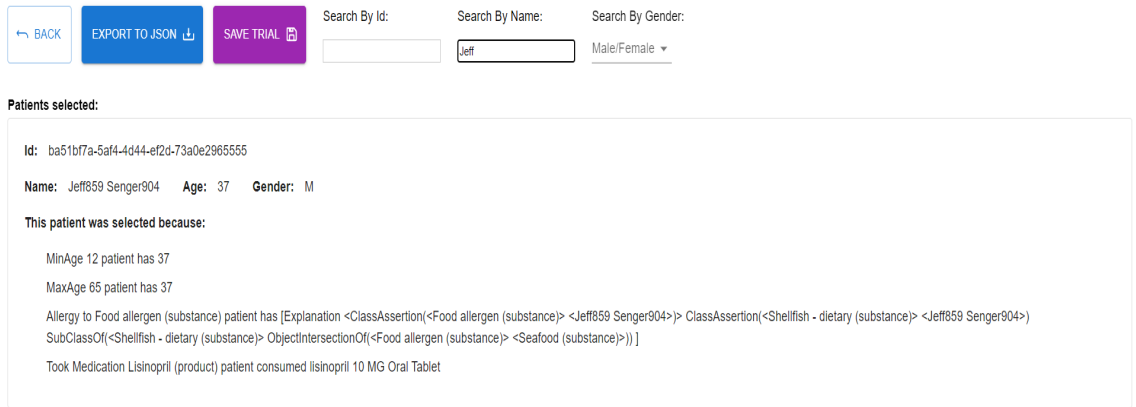

Figure II.2: Selected Patients with filter by name

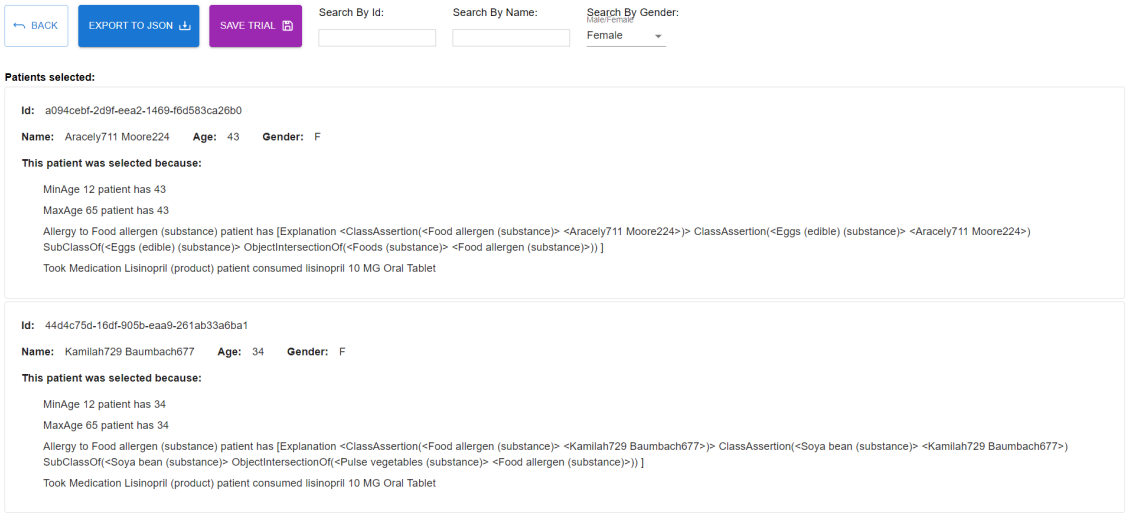

Figure II.3: Selected Patients with filter by gender