João Paulo Pina Fernandes

Serviços de vocabulário para aplicações de eSaúde em Portugal

Vocabulary services for eHealth applications in Portugal

João Paulo
Pina Fernandes

Serviços de vocabulário para aplicações de eSaúde em Portugal

Vocabulary services for eHealth applications in Portugal

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática (M.I.E.C.T.), realizada sob a orientação científica do Professor Doutor João Paulo Trigueiros da Silva Cunha, Professor Associado com Agregação do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto e do Mestre Ilídio Fernando de Castro Oliveira, Assistente Convidado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri / the jury

presidente / president

Professora Doutora Maria Beatriz Alves de Sousa Santos,
Professora Associada com Agregação do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

vogais / examiners committee

Doutora Liliana da Silva Ferreira
Investigadora Sénior do Instituto Fraunhofer - Portugal

Professor Doutor João Paulo Trigueiros da Silva Cunha
Professor Associado com Agregação do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto (orientador)

Mestre Ilídio Fernando de Castro Oliveira
Assistente Convidado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro (co-orientador)

agradecimentos / acknowledgements

Quero agradecer a todas as pessoas que estiveram presentes no decorrer no meu percurso académico. Começo obviamente pela minha família especialmente os meus pais e irmãos (não esquecendo os restantes, pois foram uma constante imprescindível na minha vida).

Aos meus amigos que me acompanharam desde que tenho memória e aos que foram aparecendo e ficando.

Aos meus professores e colegas que me ajudaram a crescer como aluno e pessoa.

A quem partilhou o tempo que dediquei a este trabalho.

Ao Prof. Ilídio Oliveira um grande obrigado pelo tempo despendido e pela sua ajuda que foi um suporte essencial para a elaboração deste projecto.

Palavras-chave Interoperabilidade semântica, Terminologias, Servidor de Vocabulário, eSaúde.

Resumo

O uso seguro de eSaúde requer que as ferramentas de informação partilhem a mesma interpretação de dados mas, no actual estado das implantações, os sistemas são normalmente heterogéneos e adoptam modelos de informação locais. A falta de soluções para a comunicação entre diferentes sistemas a nível técnico e especialmente a nível semântico dificulta a capacidade de usar a informação relativa ao mesmo utente de forma continuada entre múltiplos sistemas.

Uma contribuição parcial para facilitar a integração de fontes de informação diferentes é o uso de terminologias médicas, que clarificam o uso pretendido de certos campos da informação e os respectivos valores.

Neste trabalho é proposto o uso de um servidor de vocabulário como um componente central do sistema com o objectivo de satisfazer dois casos de uso mais pertinentes: (1) criar um serviço de referência para a realidade portuguesa e (2) permitir a transformação de estruturas de informação para outros modelos clínicos (para cenários de interoperabilidade).

A ferramenta proposta, além de funcionar como um servidor de terminologias relevantes para o sistema de saúde português, é também capaz de modelar associações semânticas entre terminologias diferentes, permitindo assim a tradução e transcodificação de conceitos. As especificidades da rede de interoperabilidade do epSOS foram tomadas em consideração para o desenvolvimento das especificações.

O sistema possui a capacidade de mapear terminologias carregadas, oferece uma representação dessa informação (e.g. vista de um grafo de conceitos relacionada com uma doença específica) e permite importar essa mesma informação nos formatos RDF e JSON. Uma interface de programação de aplicações (API) foi desenvolvida para permitir a um utilizador fazer interrogações semânticas de alto nível, como por exemplo, o mapeamento entre terminologias usadas no sistema de saúde português.

Os resultados deste trabalho podem facilitar o desenvolvimento de soluções em eSaúde através da disponibilização de serviços básicos relacionados com terminologias, melhorando assim a interoperabilidade das aplicações.

Keywords Semantic interoperability, Terminology, Vocabulary server, eHealth.

Abstract

The safe use of eHealth requires that information tools share the same interpretation of the data but, in the current state of the implementations, systems are often heterogeneous and adopt local information models. The lack of interfacing solutions between different systems at the technical and, specially, semantic level, hinders the ability to use seamlessly information for the same patient, available at multiple sources.

A partial contribution to facilitate the integration of different information sources is the use of medical terminologies, which clarify the intended use of certain data fields and the possible value sets.

In this work, we propose the use of a vocabulary server as a central component to enable two motivating use cases: (1) enable a reference semantic service for the Portuguese reality and (2) enable the transformation of clinical data structures into other clinical models (for interoperability scenarios).

The proposed tool, besides serving terminologies relevant to the Portuguese health system, is also capable of modelling semantic associations between different terminology systems to enable translation and transcoding. The specific requirements of the epSOS interoperability network were used to drive the specification.

The system is able to link terminologies, offer a visual representation of that information (e.g. the viewing of a graph of concepts related to a specific disease) and allows that information extraction in RDF and JSON formats. An application programming interface was developed to enable developer to issue high-level semantic interrogations like, for example, mapping between terminology systems used in the Portuguese health system.

The results of this work can facilitate eHealth solutions developers on getting basic terminology services to extend their applications towards enhanced interoperability.

Contents

List of Figures

List of Acronyms

Acronym	Description
ADL	Archetype Definition Language
AJAX	Asynchronous JavaScript and XML
API	Application programming interface
ASL	Application Service Layer
caCORE	cancer Common Ontologic Representation Environment
CDA	Clinical Document Architecture
CTS	Common Terminology Services
DBMS	DataBase Management System
EHR	Electronic Health Record
eHealth	electronic Health
epSOS	Smart Open Services for European Patients
GUI	Graphical User Interface
HL7	Health Level Seven International
HTML	HyperText Markup Language
ICD	International Codification of diseases
ICPC	International Classification of Primary Care
JDBC	Java Database Connectivity
JIT	JavaScript InfoVis Toolkit
JDBC	Java DataBase Connectivity
JSON	JavaScript Object Notation
LOINC	Logical Observation Identifiers Names and Codes
MDE	Model-driven Engineering
MDR	Medical Dictionary for Regulatory Activities
MeSH	Medical Subject Headings
NCP	National Contact Point
OBO	Open Biomedical Ontologies
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RTS	Rede Telemática de saúde
RDFs	Resource Description Framework Schema
SDK	Software Development Kit
sIOP	Semantic interoperability
SNOMED CT	Systematized Nomenclature of Medicine Clinical Terms

Continues on the next page

Acronym	Description
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPARQL	Protocol and RDF Query Language
SQL	Structured Query Language
UMLS	Unified Medical Language System
UTS	UMLS Terminology Services
URI	Universal Resource Identifier
W3C	World Wide Web Consortium
WHO	World Health Organization
WHO-ART	World Health Organization-Adverse Drug Reaction Terminology
WSDL	Web Service Description Language
XHTML	eXtensible Hypertext Markup Language
XML	eXtensible Markup Language

Chapter 1

Introduction

1.1 Motivation and context

The sharing of Electronic Health Records (EHR) over national eHealth infrastructures is becoming a priority for many countries [1]. This exchange of data raises some issues. The safe use of eHealth requires that information tools share the same interpretation of the data but, in the current state of implementations, systems are often heterogeneous and adopt local information models. The lack of interfacing solutions between different systems at the technical and, specially, semantic level, hinders the ability to use seamlessly information for the same patient available at multiple sources. Terminologies evolved dramatically this decade and they are a crucial element in applications enrolling in data integration, decision support and knowledge management [2].

In Portugal, major health centers acquired medical software to manage and store patient data. This is one of the reasons that turned software part of problem instead of the solution. Meaning that most of the software used was not developed regarding communication with third parties. The previous fact leads to an inefficient use and sharing of information. The amount of duplicated and contradictory information increases with the amount of different and non interoperable health software. The lack of terminology use or patient unique identifiers prevents access to data, a proper use of that data and loss of information.

In this dissertation we present MedLexIEETA, a system developed to offer terminology related tools to provide means of transcoding between terminologies and translation of concepts.

There are two major scenarios we will have to take in consideration, Portugal and cross-border. In the Portuguese case and directly related to the *Rede Telemática de Saúde* [3] inner interoperability, it has to provide a set of tools to help data encoding and normalizing. Regarding Europe we will need to take into account the specifications of the Smart Open Services for European Patients (epSOS) project to demonstrate MedLexIEETA capabilities regarding the semantic problem of epSOS.

RTS is a platform for integrated access to points of the EHR, scattered in the region of Aveiro, developed by the University of Aveiro. It provides a network that enables the share of information between the major health centers and small ones. Developed under the project *AveiroDigital*, this system provides interoperability between some primary and secondary health centers of the region. It allows institutions access to unified data regarding patient episode history. The access to this data is provided to health professionals through

the site *Portal do Professional*. RTS provides an improvement in the way that clinical data is transferred between health centers, among patients and health professionals helping the improvement of patient engagement in his own health [4].

The epSOS project is the main European eHealth interoperability pilot[5]. There are now 23 European countries (figure 1.1) working on it and the ultimate goal is to offer seamless healthcare to European citizens by improving quality and safety. This improvement is related patient care while absent from its country, by providing medical data to professionals in the receiving country. epSOS aims at developing an infrastructure that supports the share of information while regarding patient safety reducing the rate of medical errors and providing faster access to data. Note that epSOS allows only the seamless retrieval of patient data stored remotely at his domestic health system, in the form of a clinical summary and a list of e-prescriptions. In some cases the previous data is not available what leads to time loss that can be the difference between an easy and fast diagnosis and a difficult and time consuming one. Portugal joined the project in 2012. The first goal of Portugal was to research and evaluate the epSOS state of the art to build a solution for the Portuguese National Contact Point (NCP). A NCP is the communication point for nations to exchange data. After achiving the first goal new challenges were determined, among those there is one (i.e. achieve semantic quality of information present in an EHR to ensure the compatibility of data between nations) that is the reason for us to approach the epSOS project[6].Being a reference project in the area epSOS cases and situations will be taken in consideration in which MedLexIEETA can help in closing the gap between the Portuguese Health network and the epSOS network [5].

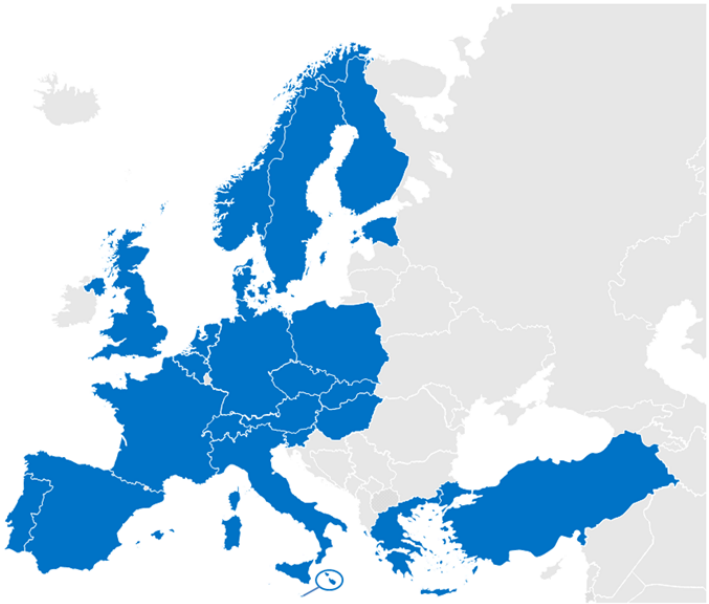


Figure 1.1: epSOS developing countries [5].

1.2 Objectives

A system developed to health scenarios can be described as being a system designed to assist in the management of overall clinical information and serve as a tool to improve the quality of care provided in those institutions. MedLexIEETA goal is to serve as a semantic tool to address two main objectives: to enable a reference semantic service for the Portuguese reality and enable the transformation of clinical data structures into other clinical models (for interoperability scenarios). More exactly the system needs to handle data from the Portuguese coding schemes and translate to the epSOS standards or transcoding between health terminologies.

We have to develop a system capable of handling terminology data and provides tools to load and retrieve information. The system must be made available by means of a web service to ensure the use of the information to future development of different applications.

1.3 Dissertation structure

This dissertation is divided in the following chapters, excluding this one:

- **Chapter 2 - State of the Art**, presents an overview of the existing literature, technologies and solutions within this dissertation scope, in an attempt to gather relevant information to design and implement MedLexIEETA knowing the strategies, assumptions and capabilities of the existing approaches to similar scenarios.
- **Chapter 3 - MedLexIEETA**, describes the functional characteristics of the system and which problems the system is going to solve.
- **Chapter 4 - System Architecture**, the MedLexIEETA proposed architecture, its main modules, their implementation and role in the system.
- **Chapter 5 - Implementation**, the implementation process and every important part related to the development of the system.
- **Chapter 6 - Results**, analyses the results, discussing if the system responds to the proposed problems and the accomplished characteristics.
- **Chapter 7 - Conclusions**, enumerates some issues found during the development of this work, the achieved accomplishments and proposes further work improvements.

Chapter 2

State of the Art

In this chapter we will address some areas that are relevant to this work, presenting concepts and previous works to understand what has been done and which results and conclusions that have been reached. We will focus on the following subjects:

- Semantic tools for interoperability and information sharing.
- Biomedical terminologies.
- Terminology servers: principles and selected examples in the biomedical domain.

2.1 Semantic tools for interoperability and information sharing

Health facilities are focused in the care of patients. Medical information records should also be focused in that care. For this information to be shared among health facilities the concepts used to describe what has been done must also be shared. The clinical information of patients must be shared and means to access patient records relevant parts must be provided. The concepts that belongs to a specialized health center can be kept separately but the basic health related concepts must be shared.

An Electronic Health Record (EHR) is a specific set of data that contains the most important information of a patient's medical record, it is stored digitally and its purpose is to improve access to medical data [7]. Nowadays, EHR data is stored in all kinds of formats spread out through the multiple medical systems available in the market. The available formats can go from relational databases or unstructured document based storage. Data storage can be structured or unstructured and may not obey to an open standard.

This need to share information comes with a number of related concepts and technologies.

2.1.1 Semantic interoperability concept

IEEE defines interoperability as "*The ability of two or more systems or components to exchange information and to use the information that has been exchanged.*"[8]

There are two major components to achieve interoperability, the ability to exchange information (i.e. syntactic interoperability) and the ability to process the information once

it has been received (i.e. semantic interoperability (sIOP)). To understand the two concepts two small examples are given: first, two people that are syntactically but not semantically interoperable. Second, semantically but not syntactically interoperable which we consider that it is not possible. Regarding the first one, there are two people with different languages, that neither one understands. Information has been exchanged but they cannot understand each other. In the second example, we take in consideration two people that one is blind and one who is deaf, they can try to exchange information with the tools available one with writing and one with talking but they will not be able to communicate (i.e. the message will not be received in the end part of the system). This would make them semantically but not syntactically interoperable [9]. Semantic Interoperability is the ability of two or more computer systems to exchange data in a format that is understandable and interpretable by the message destination system. The meaning of the message must be automatically and correctly interpreted from its content.[2]

To achieve an interoperable system we need: a clear and consistent interface to the data, a terminology whose meaning is clear and unambiguous to the ones that use the data and semantics to link it all together.

In eHealth, semantic interoperability is the study of how to use meaning between health services. This involves coding and transmission of data between health providers, patients, citizens, authorities, research and training [10].

Semantic interoperability has three major desired characteristics: consistency, understandability and reproducibility. Consistency is the system capacity to carry unambiguous information, leading to an easy recognition of what is transmitted among the system. Understandability is the bases of communication, and is needed in the system. Reproducibility consists on the reliability of data once it has been aggregated. To provide these capabilities to the system it is defended in [11] that there are four levels of interoperability, two of those that are related to semantics interoperability.

- **Level 0:** absence of interoperability.
- **Level 1:** syntactical interoperability but lack of semantic interoperability.
- **Level 2:** partial semantic interoperability.
- **Level 3:** complete semantic interoperability.

To understand these four levels an example is given: patient X moves from Ireland to Spain. Patient X falls hill and needs to go to the hospital 1 (H1) and it is transferred to hospital 2 (H2). In the hospital one of four developments can occur:

Level 0: Patient X medical record from Ireland and the result from H1 are not available in H2.

Level 1: Patient X medical record from Ireland is available but none of the doctors present in H2 can interpret the information. Also the results from H1 are available to the staff of H2 but there is not any automated system that transfers this information to H2 information system.

Level 2: Patient X medical record is available to the doctors of H2, but most of the information is based on free text although important data (such as allergies and medical history) are encoded with the international coding schemes, which are easily interpreted by the system and displayed to H2 doctors.

Level 3: Patient X medical record is available to the H2 information system after authentication, and all the data about Patient X is present in all information systems (e.g. Irish medical record and H1 results). This data is accessible and interpretable by H2 doctors.

2.1.2 Relevance for patient safety

In the previous example we can acknowledge the relation between semantic interoperability and patient safety, e.g. if Electronic Health Record (EHR) are available to physicians and health professionals, there is not the need to run exams to which those results are already available. Looking to Patient X example again and stipulating that Patient X exams which were made in H1 reduce diagnosis time in two hours and stipulating that Patient X needs surgery that should be executed in the viable time of one hour, the two hours from the H1s exams can be lifesaving.

Therefore if the information is not available to health care professionals there is a need for more tests and precious time is lost, time that can save a patients life.

sIOP is a major challenge in eHealth, since it allows the access to complete EHR of a patient that is built in every health facility used by the patient.

The European Commission states that "*Interoperability of electronic health record systems should make access easier, and enhance the quality and safety of patient care throughout the Community.*"[12]

One of sIOP [11] goals is to improve patient safety by: reducing avoidable errors; health statistics become faster and less costly; researches will advance rapidly and the improvement of administration decisions.

A significant reduction in avoidable errors and improvements in patient safety: the access to patient data will lead to a reduction of diagnosis related time resulting in a improvement of professionals information. The care of patients will be more disseminated, e.g. the treatment of patients will function more in a combination of community facilities and specialized centres. Remote areas will be more affected from this sharing of information [11].

Public Health will be facilitated by faster and less expensive collection of statistics: the recording of patient information will be done in the care process leading to a more effective and less time-consuming surveillance for the emergence of epidemic diseases, leading to less uncontained outbreaks [11].

Clinical and translational research will advance faster: as the diagnosis and research are all available in the care process, a patient from Portugal can benefit from a successful trial treatment on a patient from China. In the other hand therapies that are proved scientifically unsuccessful in Germany will not be tried out in China leading to less patient exposition. It is mentioned as well that to achieve privacy with this sharing of data it is needed a structure that handles the consent and privacy of patient data [11].

A balanced market and best administration decisions: major hospital suppliers will develop but there will be a need for specialized vendors for rare treatments that will be accessible to every hospital. The experience from one administration related to infrastructure can be shared to achieve solutions that become the improvement off all experiences instead of one [11].

2.1.3 Selected key technologies

In this section we survey important technologies that are viable options to the project.

Martinez et.al. conducted a research for a project which goal was to provide mechanisms for representing archetypes in Semantic Web-manageable manner and achieving semantic interoperability between EHR systems, combining Semantic Web and Model-driven Engineering technologies [13].

Wroe has examined the use of Semantic Web in his project and the ups and downs in the Health Care IT solutions of the technologies analysed. The results and conclusions obtained in this article are discussed through the following sub-sections [14].

In [15] the goal of the project was to "*highlight the requirements related to exchanging non-clinical EHR information and to show how this information exchange can be accomplished*".

The article written by Killic et.al. describes how it is possible to achieve mapping between two EHR standards that are based in the same reference information model using archetypes and semantic tools. The main technologies that are analysed in this article are the archetypes and the relation to the semantic web using OWL [1].

Arguello et.al conducted a a case study of visualising clinical information from EHRs, the goal of the project was to to enhance HL7 CDA documents with data visualization, where the graphical representations presented here, illustrate how data visualization can assist the user (physicians) to understand and search the contents of an EHR [7].

In the Portuguese context Ferreira et.al has developed the Medical Information eXtraction (MedInX) system. The project had the objective of processing patient discharges and from the text files extract accurate and structured representations of the data [16].

In the following subsections we will analyse these documents to discover why and what for technologies were used.

Semantic Web Technologies

Human beings are capable of using the web to perform certain tasks such as, finding information about a specific subject, reserve a book in a library or find the best price for a car. Nevertheless, a computer by itself cannot fulfil those tasks, it needs human interaction. Web pages are developed to be handled by humans, not for IT systems.

Tim Berners-Lee in [17] says " *I have a dream for the Web... and it has two parts*".

In the first part, and what is the web today, people can access masses of information immediately and intuitively. People can create information and share that information. The web has become a tool not just to browse but a means to create and share data. Tim Berners-Lee's dream first part is a reality.

The second part of his dream consists on a Web in which machines can analyse and interpret the data available on the web and withdraw the need for human interaction in every day-to-day task. Nowadays computers are not using all the resources that they have available to help humans in these tasks. Imagine a world where people just issue and order to a computer such as, *find me the best price for car X*, and the computer shows the best result. This is the Semantic Web that Tim Berners-Lee wants and it is the part of his dream that is not yet fully developed.

Although this second part is not developed there are now a number of technologies such as some of the ones mentioned in the following chapter that are key technologies to the Semantic Web development being the base XML.

The first step to achieve this, is to put information on the Web in a machine understandable way, information that machines can directly or indirectly process. Databases usually are relational databases, data it is stored in columns that are related between them, and we need these relations to be published as a semantic Web Page. Therefore computers have to be capable of represent and share data and for this to happen we need a universal language, here appears RDF. "*As long as documents are created within the same logical framework, such as RDF, partial understanding will be possible.*".[17]

From XML to OWL

The Semantic Web has in its story three main technologies that started at a more basic level, XML, evolving to RDF, and now we have OWL. In figure 2.1 we have a representation of the layers that these technologies occupy in the semantic web.

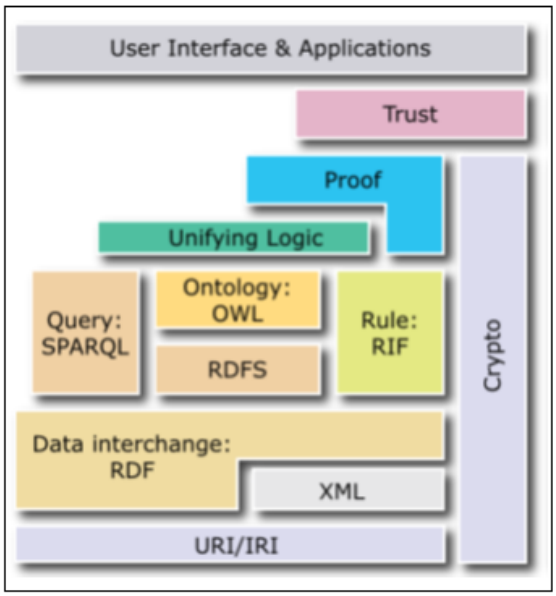


Figure 2.1: Latest Semantic Web Layers [18].

Extensible Markup Language (XML) was originally designed to be the solution for the challenges of electronic publishing [19]. The standard language for specifying web data. XML helps bring meaning to information relayed over the internet. XML adds a layer of intelligence to information [20]. Data are surrounded with tags that are readable at human level. XML tags whatever the user wants and defines and imposes no semantic constraints on the meaning of these documents [18]. XML Schema already restricts the structure of the documents and extends XML with data types [21] In [15] is defended that XML is sufficient for most applications as a messaging standard and it is used for information exchange. A structure is given to the XML messages that enables retrieval, sending and enumeration of data.

The Resource Description Framework (RDF) is a framework for representing information in the Web [22]. It is the data model of the semantic web. It describes objects and the relations between them. RDF documents come with a pointer at the top to its

RDF schema (i.e. a list of the data terms used in the document), that "*provides the tools for keeping the expressive power of an RDF document limited and its behaviour predictable*" [17]. The main advantages of RDF over alternative approaches defended in [14] are its mechanism for the identification of resources (i.e. URI, the aggregation of information regarding multiple sources) and the reification that allow statements about statements (i.e. the hierarchy that is available to translate attributes of a patient). The downsides are the exposition of RDF is limited and the domain of relational data models as repositories of health information is substantial. RDF Schema (RDFS) is a simple ontology description language. It is used for describing properties and classes of RDF resources [22]. SPARQL has the ability to express queries throughout diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware. It is the semantic query language. It is used to query RDF triples and ontologies declaratively. [18][23]

"*The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web*" [24]. OWL is a more advanced ontology language than RDFS. Adds more vocabulary for objects such as cardinality, equality, richer typing of properties [18]. One advantage of OWL defended in [7] is that the co-existence of two formats i.e. a specific XML regarding HL7 clinical document architecture and the transformation into OWL allow the benefits of a format that is standard for healthcare and at the same time, to allow standardization that can be used in the Semantic Web.

In [14] it is said that is possible to represent terminology, i.e. Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT) concepts in the OWL language as OWL classes. The logic behind the structure of SNOMED CT is Ontology based so it makes OWL language the perfect candidate to move SNOMED CT to a more machine interpretable semantics.

OWL in [1] was used to obtain class property mappings of two EHR standards through derivation. Meaning that using reference message information model of two standards and the derivations that are present in both it is possible to define archetypes and their mappings to achieve interoperability. These archetypes and mappings are then transformed into OWL instances.

Archetypes

OpenEHR [25] and iso13606 [26] architectures and standards are based on a dual model-based architecture that is divided in reference model and the archetype model. The EHR generic data is defined in the reference model. Archetypes are detailed definitions of clinical concepts in structured form and constrained combinations of the entities of the reference model [13].

In figure 2.2 is available an example of a archetype. An archetype is composed of three parts: header section, definition section, and ontology section. The header has a unique identifier and some descriptive information such as author, version and status. The definition section contains restrictions associated with the clinical concept defined by the archetype [13][1].

In [13] the archetypes are used to obtain ontologies related to the knowledge that is present in the archetype model and the reference model. A tool was also developed to translate ADL archetypes into OWL. The transformation process adopted here is present in figure 2.3 and is divided in three stages. Firstly the Archetype Definition Language (ADL) archetype is parsed and transformed into a syntactic model regarding a set of rules. Secondly the rules derived

```
OBSERVATION[at0000] matches { -- Complete blood picture
  data matches {
    HISTORY[at0001] matches { -- history
      events cardinality matches {1..*; unordered} matches {
        EVENT[at0002] occurrences matches {0..*} matches { -- any event
          data matches {
            ITEM_TREE[at0003] matches { -- structure
              items cardinality matches {1..*; ordered} matches {
                ELEMENT[at0004] occurrences matches {0..1} matches {-- Hemoglobin
                  value matches {
                    C_QUANTITY <
                      property = <[openehr::119]>
                      list = <
                        ["1"] = <
                          units = <"gm/L">
                          magnitude = <|>= 0.0|>
                    > > >
                } }
                ELEMENT[at0012] occurrences matches {0..1} matches {-- Platelet count
                  value matches {
                    C_QUANTITY <
                      property = <[openehr::119]>
                      list = <
                        ["1"] = <
                          units = <"10*9/L">
                          magnitude = <|>= 0.0|>
                    > > >
} } } } } } } } } }
```

Figure 2.2: Part of a "complete blood picture" archetype [1].

---

from the ontologies of the two standards are applied to the syntactic model from the previous step resulting in a semantic archetype model. For last the semantic model is transformed into and OWL archetype.

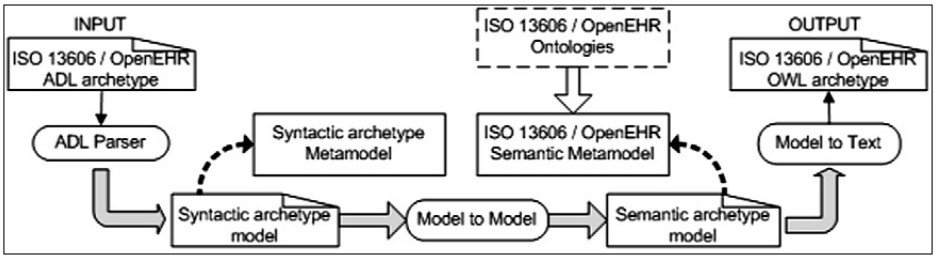


Figure 2.3: Architecture of the solution developed in [13].

Ontologies

In [27] the author refers to ontologies as "*artefacts to represent human knowledge and as critical components in knowledge management, the Semantic Web, business-to-business applications, and several other application areas*".

Ontology gives meaning to information structures that are exchanged between IT systems. It defines a set of concepts with which to model a domain of knowledge. It consists in a collection of classes and properties and the relationships between them. Ontologies provide meaning to concept maps. Classes are normally order hieratically. Properties can also be order hieratically. Figure 2.4 show an building example ontology.

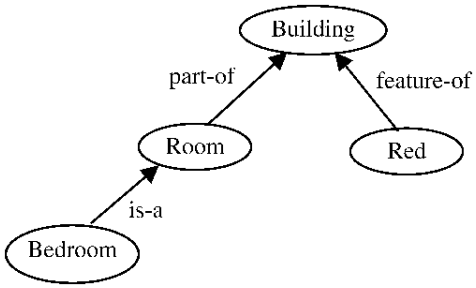


Figure 2.4: Building Ontology Example [28].

2.1.4 Related technologies

In this section we talk about the technologies that can be important for the project but are not directly connected to semantic.

AJAX

Ajax, short for Asynchronous JavaScript and XML, is not a technology [20] but a combination of several existing technologies and the way they are used together to produce dynamic content on web pages. AJAX is used in applications such as Google Maps and Youtube. It incorporates HTML and XHTML to exchange information with a server, CSS to style the data in the website, Javascript to display information and XML as the message format that are transmitted between the website and the server. In 2.5 a AJAX workflow is shown. In [7] the author regards this technology to visualize time-based events. The data to populate the timeline is withdraw from a XML file.

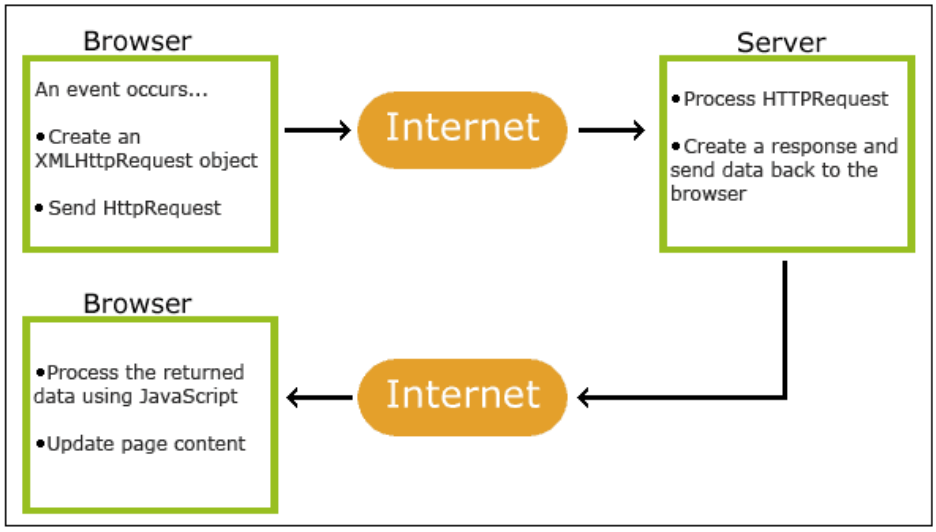


Figure 2.5: AJAX workflow[29].

Web Services

"*A Web service is a software system designed to support interoperable machine-to-machine interaction over a network*" [30].

Web Service (WS) are one communication paradigm that defines the mechanism of interoperability among applications. Nevertheless, this concept is independent of the technology used in its implementation. One of these technologies is the Simple Object Access Protocol (SOAP).

SOAP

SOAP is a technology originally developed to fill the void between disparate RPC-based communication platforms [31]. By using XML, SOAP is a simple and highly flexible protocol. However, the use of XML carries some disadvantages, due to the messages format that can make the processing of messages resource demanding, and the weight those messages have on the network.

JSON

The JavaScript Object Notation (JSON) was not invented but discovered. In 2001 received a name, a description and also the values of this technology as data interchange format were pointed out. The main advantage of JSON is that it will never change because it has not a version number and there is not a mechanism to revise the technology [32].

Even if it was born as being a JavaScript resource it was developed to be an language independent format. The data structures represented are the same data structures of a programming language. The codification of data is so simple that it can be interpreted by almost every programming language [32]. Compared to Java it uses two main structures, objects and arrays. The object is composed of a "*string : value*". The array is enclosed with rectangular brackets hosting the values of the array separated by commas [33]. In figure 5.11 a concept is highlighted and it is possible to see these two structures.

2.2 Biomedical terminologies

A terminology can be defined as a "*an organisation of entities into classes for a specific purpose such as international reporting or remuneration.*".[11]

In the global world that we leave today we cannot demand that health professionals talk the same language, and even if they do, semantic problems could still occur. To avoid this problem, were developed concept coding systems. At this pairs of concepts and codes is given the name of terminology. The origin of this concept regarding health terminologies began with a statistical study in the *XVII* century when a man called John Graunt tried to estimate the percentage of newborns that died before reaching the age of six. The first real terminology however appears in the *XIX* century with a classification of causes of death, The Bertillon Classification of Causes of Death, this classification served as a base for the International Classification of Diseases (e.g. ICD).[34]

Terminologies are an essential key to solve redundancy in health related concepts. Disease "Cholera" can have more than 100 representations spread throughout the world, like

"cholera", "Unspecified cholera", "Cólera" and so on, this leads to redundancy. If we assume that the disease with the code "A00" it is all the previous descriptions, we do not have redundancy and all the possible descriptions for instant translation are there. A terminology in different languages can now be linked. Code "A00" is the "Cholera" in ICD10 English version, and "Cólera" in ICD10 Portuguese version.

With the combination of terminologies and bioinformatics in health it was created the concept of "health smart" Semantic Web. If we join this concept with a health aware user we can develop tools to improve consumer health and well being. This will help users to be more health aware therefore improving their knowledge but also their responsibility.[35]

There are a lot of health related terminologies and among them we can highlight the following ones:

- **International Classification of Diseases (ICD):** developed to promote international comparability in the collection of statistics from death certificates and health records. Created in the XIX century [34] .
- **International Classification of Primary Care (ICPC):**is an epidemiological tool used to classify data about three elements of the health care encounter: reasons for encounter, diagnosis or problem, and care process.[36]
- **LOINC:** is a terminology developed to facilitate the interoperability between laboratories, clinical services and care providers. It attempts to be a standard for the communication of results related to clinical care.[37]
- **Systematized Nomenclature of Medicine-Clinical Terms (SNOMED CT):** provides the core general terminology for the EHR. The concepts have unique meanings and formal logic-based definitions distributed hierarchically. It consists in the terminology core (concepts, descriptions, and relationships), that works to support the implementation and use of SNOMED CT, including subsets, cross maps to existing classifications and coding schemes.[38]

There is some confusion in the definition of terminology in eHealth since the the terminologies mentioned above are very different (i.e. The SNOMED CT is more complex than ICD 10 and both are terminologies). In [11] is defined an eHealth terminology as a representation of knowledge that includes a set of entities, their terms and relations. They may also include ontologies, thesaurus and more sources of knowledge.

Normalization problems are solved with terminologies, nevertheless data transmission and data access are not. The need of systems capable of storage and provide terminologies services exists and systems must provide interaction between a server and a client. Terminology servers are the response to these requests, they provide access to terminologies and a way to develop terminology related services.

2.3 Terminology servers: principles and selected examples in the biomedical domain

Terminology servers are systems that manage terminologies for clinical applications. They must serve as a repository during system development and in the perfect scenario publish the services developed to other systems [35]. It allows software access to terminologies.

The first approach to the concept of terminology server [39] was also analysed in [40] these services as essentials:

- *"Managing external references.*
- *Managing internal representations.*
- *Mapping natural language to concepts.*
- *Mapping concepts to classification schemes.*
- *Management of extrinsic information."*

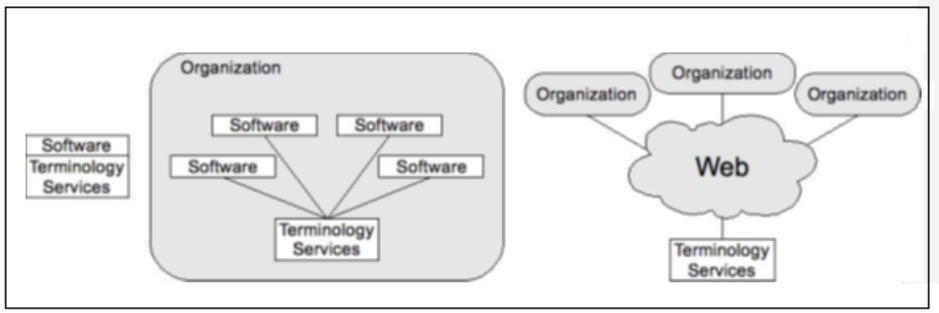


Figure 2.6: Impact of sharing terminology related services as Web-Services. [35].

In figure 2.6 we can see the impact that a terminology server can make in combination with a web service, it leads to the reuse of terminology data to improve other applications. In [35] is carried out an analyses of publications related to terminology serves, in an attempt to answer the question "what requirements are relevant to terminology server functionality?" The answer is divided in two main sections, terminology access (e.g. services for semantic computation) and management (e.g. services required to keep the terminology server properly running). Management services include:

- Addition / Deletion (mass): addition and deletion of data to the terminology;
- Deprecation and replacement: transition from previous versions to new ones.
- Development (collaborative): development of the terminologies.
- Verification: Mechanism to check the terminology state.
- Security and privacy: If the server has private information, server access should require restricted access.
- Extensibility: the server must be able to reshape is software modules.
- Usage and resource monitoring: Keep track of server's usage.
- Resource allocation, distribution and scheduling: The managing of resources must be properly developed.

- Inventory: Provide information about the state of the server.

Access services include:

- Querying: the server must respond to all basic requests of data and be capable of response to more ambitious ones (e.g. retrieve attribute values that support translation to other coding schemes).
- Knowledge generation or inference: Generate inexistent knowledge.
- Natural language bridging: Translating natural language to terminology concepts.
- Terminology interoperability: Mapping between terminologies from inside the server and outside the server.

In [41] a solution for a similar problem to the one we are trying to solve is elaborated based in two main components the Unified Medical Language System (UMLS) and a terminology server lexEVS. They used the first to deal with the most common and spread terminologies and the second to load and use terminologies that were not available in the UMLS.

2.3.1 lexEVS

The Cancer Common Ontologic Representation Environment (caCORE) was created by the National Cancer Institute Center for Bioinformatics (NCICB) to solve the requirement of systems to communicate and understand each other (i.e. to be interoperable) in the context of biomedical research systems. It is an infrastructure that was made to provide a mechanism to create interoperable biomedical information systems. As previously said there are two main requirements that must be fulfilled to achieve interoperability, syntactic interoperability and semantic interoperability. To achieve this caCORE was developed using the Model Driven Architecture (MDA) paradigm to provide a data repository with a clear and consistent access, controlled terminologies available at runtime to give meaning to those who use the data repository. The last piece of the puzzle is the link between the controlled terminologies and model driven data, a source of knowledge that gives meaning and consistency to the classes and attributes retrieve from the data system [9]. LexEVS is a caCORE Software Development Kit (SDK) generated system.

LexEVS provides a common terminology model, open access to loaded terminologies and retrieve information from that data [42]. Not only can users use the terminologies that are available through the Enterprise Vocabulary Services (EVS), but also create their own terminologies.

It gives the ability to control content of a terminology, retrieve set or sub-sets of terminologies and mappings. It Implements the draft specification for HL7 Common Terminology Services 2 (CTS 2). There is a support for mapping associations between terminologies. It is possible to load terminologies in three basic formats: text, XML and OWL formats. However these formats are used to load for example the Open Biomedical Ontologies (OBO) text format and the metadata loader that is provided in XML format. LexEVS provides the following components: Terminology Server; JAVA API; Rest/HTTP Interface; SOAP/Web Services Interface; Distributed API; LexEVS Grid Services; Developers Gui. [42]

LexEVS architecture has the following layers:

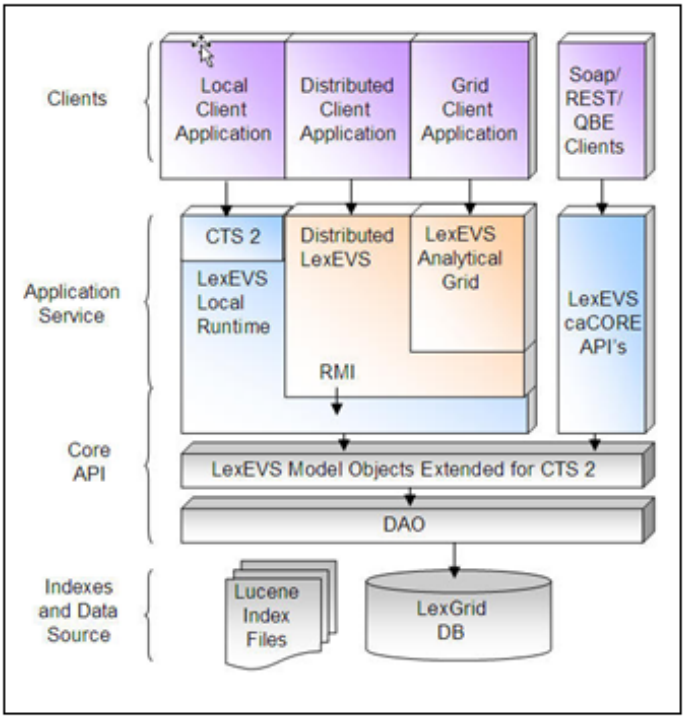


Figure 2.7: LexEVS architecture [42].

- **Aplication Service Layer (ASL):** The application layer is the link between clients and the Core Application programming interface (API). Distributed Clients access the Core API after authentication. LexEVS caCORE functions access the same metadata but do not use the LexEVS Local Runtime functions, they have direct access. The Application Service Layer is the gateway to the information present in LexEVS.

- **Core API Layer:** Carries all LexEVS API requests from the ASL. It provides access to the database. Access is done using Lucene index files to provide the cheapest way to get the information from the database. Returns objects in response to the queried information with the information requested.

- **Data Source Layer:** data repository, responsible for access the data and fetch the information required to represent the requested objects.

2.3.2 Unified Medical Language System UMLS

The Unified Medical Language System is a repository of terminologies that was created by the US National library with the purpose of overcome two main problems, redundancy (i.e. the amount names for the same concept) and the lack of uniformity (i.e. the absence of a standard format for distributing terminologies)[43].

UMLS includes over one hundred terminologies, one million concepts and 4 million names for those concepts [44]. In figure 2.8 it is shown some of the terminologies present in this terminology server are, SNOMED CT, ICD-10-CM, ICD-9-CM, Medical Subject Headings

(MeSH) and more. UMLS is composed of three knowledge sources: The Metathesaurus consists in Concepts derived from the source terminologies and the relations and attributes of those concepts; The Semantic Network that consist in 135 semantic types (e.g. Disease or Syndrome), 6.864 relations among them and the Lexical resources that are the relations between the actual concepts. [44]

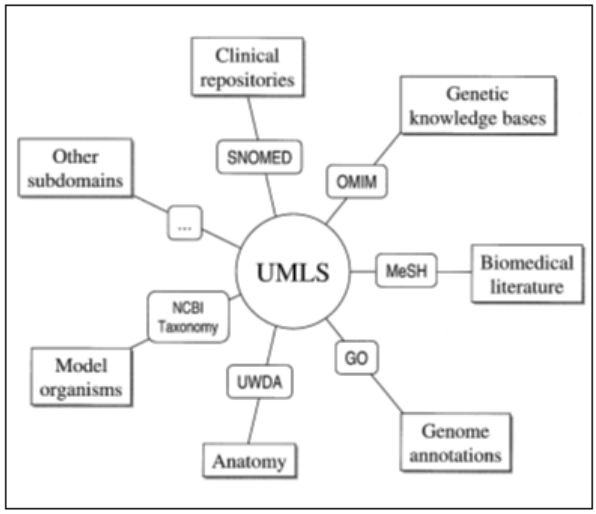


Figure 2.8: The various subdomains integrated in the UMLS [43].

UMLS Metathesaurus organizes terminologies elements by concepts(e.g. SNOMED element "Headache" is one of the elements of the concept "Headache" from UMLS concept). Concepts can have different names, and every name is kept. Essentially every concept comes from one or more sources, and the Methatesaurus tries to storage every data (i.e. two terminologies used the same name for different concepts, the Thesaurus add this name and source to each concept accordingly). For each Concept there are Terms that are different ways to represent the same concept. For each Term, there are Strings and Atoms. Strings are similar names with some changes like one being the plural of the other. For each name that comes from one terminology there is a Atom. To see an example in figure 2.9 we have the Concept Headache, for this concept there are three terms "headache", "cranial pain" and "cephalgia head pain". The Term L00118681 has two Strings "headache" and "headaches". The String S0046854 has two Atoms that came from two different terminologies (i.e. SNOMED and MeSH).

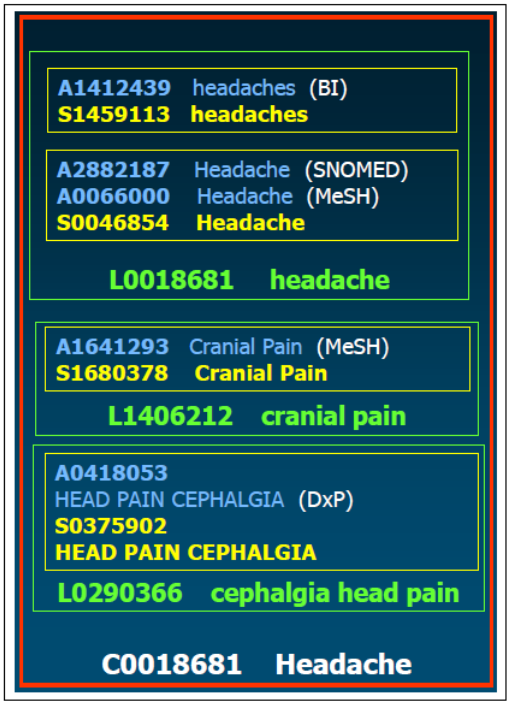


Figure 2.9: UMLS Concept.

Chapter 3

MedLexIEETA requirements

We have realized that one of the problems that the eHealth reality has to surpass is the interoperability between systems. The approach proposed here is a system that addresses the problem from the viewpoint of the terminology tools required to facilitate eHealth solutions development. It is a partial contribution departing from the Portuguese context. Therefore we need a system that can work with multiple terminologies and give response to a number of use cases related to those same terminologies.

3.1 Functional requirements

Core functions have to be developed to provide services related to the data that is available in the terminology server and the operations that result from that data. We have to be defined the system requirements and the use cases that it needs to satisfy.

3.1.1 Terminology management scenarios

The system main function will be using and retrieve terminology data. The system must have the capability to improve the amount of data that is available. It must be possible to add terminologies and update those same terminologies.

There are use cases that the system needs to satisfy related to life cycle of the terminologies. Figure 3.1 is a use case diagram of the terminology related use cases.

- **Use Case 1: Load Terminology.** A terminology has to be obtained from a official source and to be available in a format that is compatible with the terminology server. If the terminology is not available in a readable format, the developer has to transform the data. Then the user loads a terminology that has to pass a validation process to check if the terminology is structurally correct (e.g. if there are not repeated codes for concepts). After being validated is loaded in the terminology server.

- **Use Case 2: Validate Terminology.** It is a crucial step before loading or updating a new terminology into the system. The system has to incorporate a validation method to check the terminology state (i.e. valid or not valid), that preferably gives feedback about the error and the location of the error since some terminologies have more than 10000 concepts. Lets imagine that a terminology is available only in excel format and the terminology server does not handle that kind of data. The user has to transform

the data into a readable format. The transformation process can input errors in the terminology such as duplicated information or incorrect structures that have to be found and transmitted to the user.

- **Use Case 3: Update Terminology.** A terminology is not a static tool, it evolves and it is updated. For example the ICD 10 terminology is updated annually. Therefore a solution has to be made available to update a terminology. The user updates a terminology present in the terminology server, if the terminology passes the use case validate terminology is updated.
- **Use Case 4: Delete Terminology.** The user has the ability to delete terminologies from the system.

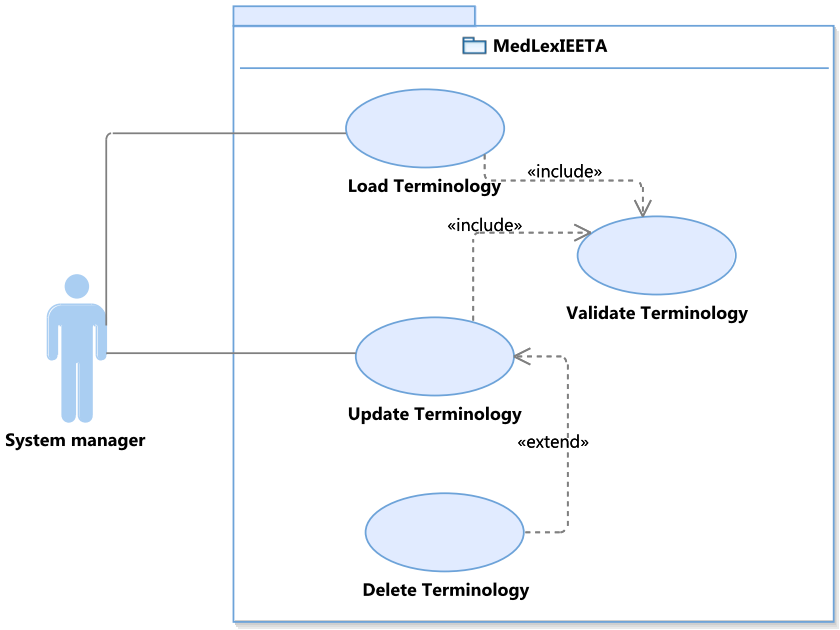


Figure 3.1: Terminology lifecycle use cases.

Based on the use cases that were previously mentioned, the terminology is loaded into the system if it passes a validation test. The system must give feedback about the terminology state and if it is not valid the user must know why and proceed to the proper changes. After inserted, the user must have the option to update and delete this kind of data. The update function must obey to the same rule of validation. Regarding the delete process the system must be able to delete the entire terminology.

3.1.2 Terminology data retrieval scenarios.

After the terminology loading process we have to extract information from the terminologies. We will have to develop services to retrieve valuable information. In figure 3.2 is present the use cases related to the extraction of terminology data from the system.

- **Use case 5: Retrieve concept by description.** The User provides the concept description and the system searches the terminologies present for that same description. The system has to find exact matches and similar matches for the provided description. For each match the concept is retrieved to the user. The probability of multiple matches for a description is high so a mechanism to process queries has to be developed. When finding a given concept by a provided description the system will search the terminologies for that same description and has to prompt the user what is the concept that he wants.

- **Use case 6: Retrieve concept by code.** The User provides the concept code and the system searches the terminologies available for that same code. If the code exists the concept is retrieved to the user. In the case of duplicated codes among different terminologies the system must have a function that allows the user to choose the wanted concept.

- **Use case 7: Retrieve description by code.** The User provides the concept code and the system searches the terminologies available for that same code. If the code exists the description of the concept is retrieved to the user. In the case of duplicated code among different terminologies the system must have a function that allows the user to choose the wanted concept.

- **Use case 8: Retrieve relations of given concept.** The user requests the related codes of a given concept. The system searches the terminologies for that concept. If the concept exists, it is gathered and retrieved all the related concepts: the concepts inside the concept terminology (i.e. the "parent" concept and "children") and mapped concepts of other terminologies.

- **Use case 9: Mapping between terminologies.** The user provides mapping between some terminologies to the system. The system has to provide support to map this data between terminologies. Being this the most important use case it is crucial that the proper mapping can be achieved between the terminologies.

3.1.3 Terminology server system attributes

For all the previous use cases the system will need a terminology server to store and handle the terminologies that the project will use. The terminology server must provide all the characteristics approached in the previous use cases. It has to be defined what terminologies to use, what queries can be prompted and the kind of responses.

The terminology server is the main tool of MedLexIEETA. We will need to develop functions related to terminology data, means to extract and share that information. Also some decisions related to the understandability of the information had to be made. This reasoning led to the following requirements.

- **Translate information into readable and/or usable formats.**

  There are some use cases that a simple *String* can be a response to a use case (e.g. use case 6), however there are use cases that need more than a *String* or a *List* of *Strings* to provide the requested information (e.g. use case 8). The System must provide interfaces

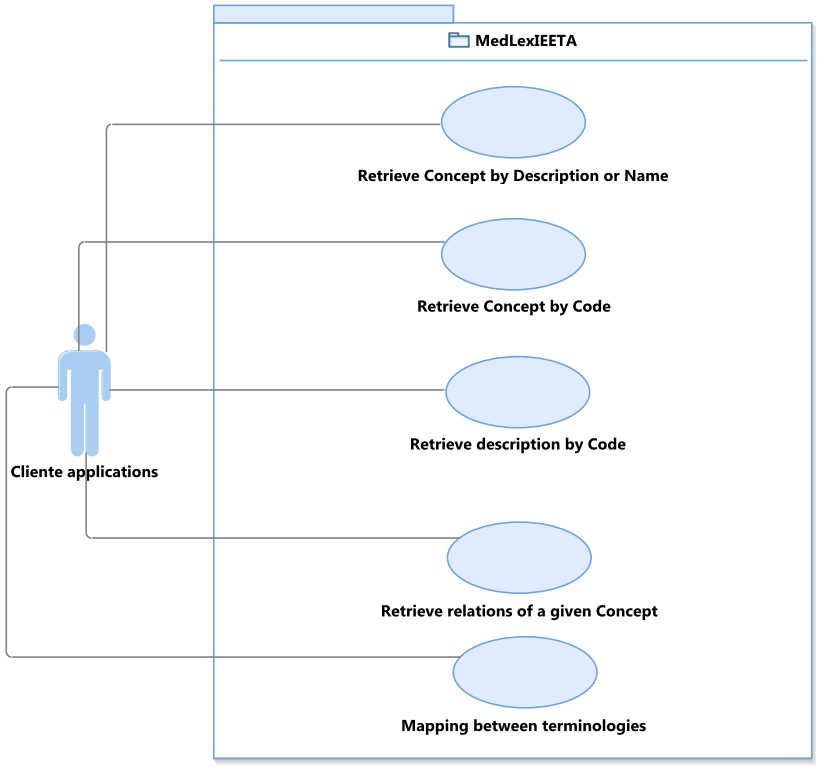


Figure 3.2: Retrieve and handling terminology data use cases.

to extract data in formats that are readable to a user or a information source to other system.

- **Offer service interfaces to enable extensibility of the system.**

  A web service is going to be available to provide access from other developers to system services. All important and relevant methods must be publicly available and documented to full access and use of the system data. The services that are used by the system have to be published to let other systems developers improve the quality of their systems.

- **Provide graphical representation of large data results.**

  With a vast number of terminologies and mapping done between each other the result of a complete response to a query (i.e. full disease information) can be difficult to human recognition [10]. There will be a need to graphically represent information in graphs.

Chapter 4

System Architecture

In this chapter we will describe the basic architecture and technical options made in MedLexIEETA. In the end of this chapter a complete architecture is displayed in figure 4.2.

MedLexIEETA is a software application to support terminology encoding and provides mapping between terminologies. The MedLexIEETA (figure 4.1) architecture is divided in three modules:

- Terminology Module

  The Terminology module is the core of the MedLexIEETA architecture and it is the crucial component of this project. This module is responsible to aggregate the terminologies that will be used and to compute that data into usable information. It will have to deploy one or more terminology servers. This module must be able to work with multiple languages in order to support the development of eHealth systems (e.g. multilingual clinical systems). The requests and the corresponding response to the system are processed here. The response has to be provided in different formats to offer different ways in which it can be processed.

- Web Service Module

  The Web Service Module is the gateway to the system. Here the calls to the system will take place. The requested calls have to be translated to methods of the terminology model, to be processed a reply. It works as a bridge between the terminology module and the visualisation module that acts here as the third party system.

- Visualization Module

  There are some cases that the requested information will be vast and not always clear, the visualization module will be the answer for easy human recognition and data understanding. The visualization module is also a client representation that uses the services provided by the system.

4.1 Design of the terminology module

In this section the decisions behind the terminologies and the terminology server are discussed.

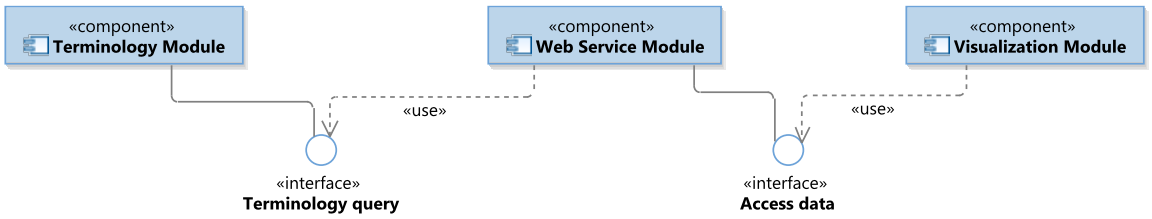


Figure 4.1: Basic architecture of MedLexIEETA.

Regarding terminology related decisions.

The terminologies that will be used must have at least one of these characteristics: be relevant to Portuguese eHealth reality or be available in multiple languages to demonstrate the multi-languages ability of the system. The terminologies loaded into the terminology server will be the ICD-9-CM in Portuguese, ICD-10-CM in Portuguese, "*Tabela dos Convencionados da Administração Central do Sistema de Saúde (ACSS)* (Table that contains prices, taxes, description and codes for health cares in Portugal) and the allergies terminology developed by the Portuguese health system authorities. From the UMLS Metathesaurus are the International Classification of Primary Care (ICPC), the Medical Dictionary for Regulatory Activities (MDR), the WHO Adverse Drug Reaction Terminology (WHO - ART) and the SNOMED-CT terminology.

Regarding terminology servers related decisions.

We will use the same components used in [41], their approach to solve the same problem but for the British reality consists in using the UMLS to deal with the commonly used terminologies and one LexEVS server to deal with the terminologies that were related to the national eHealth reality. We will then use one terminology server (LexEVS) and the UMLS Metathesaurus. The UMLS Metathesaurus has over of one hundred terminologies and an API that gives access to all that information. The thought therefore was to use the terminologies that are already available in the UMLS Metathesaurus in both languages (i.e. Portuguese and English). The terminologies that are present in UMLS Metathesaurus are already mapped, so we will use this mapping. LexEVS will serve as a repository for the terminologies that are not available in the UMLS Metathesaurus. With these decisions we will met the requirements that were proposed in chapter 3. We will have to deploy an instance of the LexEVS server, load the terminologies into it and implement the services that will be required. The UMLS Metathesaurus will be integrated in the solution, and work as a source of knowledge for MedLexIEETA. We will have to implement services to request and interpret data from the Metathesaurus. In figure 4.2 the components in blue are the ones that we will have to implement, and in red the one that we have to integrate in the solution.

4.2 Web service module

For this module we had two major classes of web services that are used in SOA, SOAP and REST (Representational State Transfer). REST web services are based on the following principles, resource names are referenced by URLs and the REST interfaces are limited to

HTTP. The methods available to are the POST (i.e. create resource), GET (i.e. retrieve a resource), PUT (i.e. alter a resource) and DELETE (i.e. eliminate a resource).

REST WS are considered simple, effective and for most applications it is considered an adequate solution. The simplicity of HTTP brings more advantages then to add another network transport layer. REST WS design is advised when the producer and the consumer previously agreed and know the type of request and the type of response. The absence of headers and other layers of XML make REST a good choice to limited bandwidth cases. [45]

We do not need most of the non functional requirements that justifies the use of SOAP services (e.g. Security, addressing and trust) and REST is more efficient and a simples choice for the web service.

There are a number of services that must be developed and next we present the generalization of these services.

- **Get Single Concept**: the service receives a input that can be either a code or a description and it has to retrieve the concept that matches the input. In the case that the description is the input a method to solve the ambiguity of the input has to be developed. This service has to be developed to enable access to every terminology used.
- **Get Full disease**: this service will be the most complex service. The service has to return all the information present in all the terminologies used that is related to a single disease.

4.3 Visualization module

The visualization module has the purposes of serving as a possible client for the web service and to give a clear representation of the data that is extracted trough the services developed.

The decisions behind this module were the following:

- Implementation of the web-site

  We use JQuery and HTML to do the design of the web page related to the visualization module.

- Libraries to represent graphed information in the web

  The solution that we found was to represent the data in JSON format using the library JavaScript InfoVis Toolkit (JIT). This library has the ability to represent tree like data and it can be a tool to present this data in a manner that facilitates a faster understanding of a large data set. The amount of information that is retrieved from the terminology server justifies this ability.

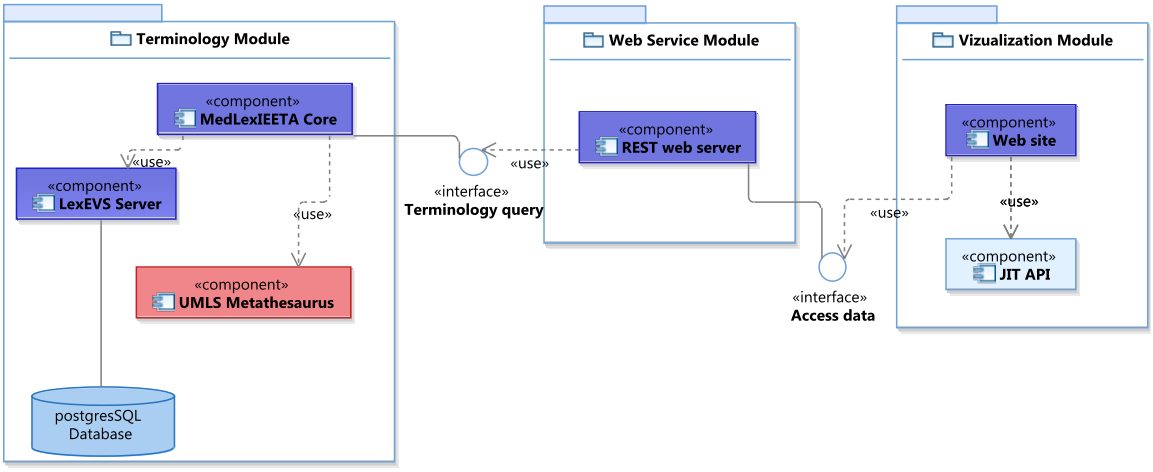


Figure 4.2: Architecture of MedLexIEETA.

Chapter 5

Implementation

This chapter covers the implementation process of the system. To achieve all the requirements discussed in chapter 3 we proposed an architecture to tackle them. Here we discuss problems of data acquiring and parsing, the developed structures and the created services.

This chapter explores the foundations of the application, the problems found while building it and the engineered solutions along with the path that took to them.

The use cases proposed in chapter 3 change according to the terminology that are related. There are cases that some of the functions related to one terminology do not apply to another terminology.

5.1 Deployment and selection of components

In this section we explain the deployment of the two main components of MedLexIEETA. First we explain the deployment of the LexEVS server and the related choices. Then the steps followed to use the UMLS Metathesaurus.

5.1.1 Implementation of the LexEVS server

The deployment of the LexEVS server as the main repository for terminologies is divided into steps. The first step in the project was to load one instance of the terminology server LexEVS. The version installed (i.e. LexEVS Local Runtime) can be configured to use different databases, being the recommended ones MySQL and PostgreSQL. Since this project is being developed under the RTS project and the DBMS of RTS is PostgreSQL, we chose this DBMS to create the terminology server database. After deploying a database, the JDBC for that database must be available to the instance of the server installed. LexEVS requires a specific configuration that is available in the file *lbconfig.props* [42]. This file contains the properties controlling the behaviour of the server. A small portion of the file is in figure 5.1. The properties configurable are database related, log files properties and other configurations. We will analyse the most relevant.

- **Supporting DataBase :**
- DB PRIMARY KEY STRATEGY: Defines the primary key strategy of the database tables. It has two possible values GUID and Sequential Integer. In the first case the index values are inserted randomly in an index three that is updated after each insert for

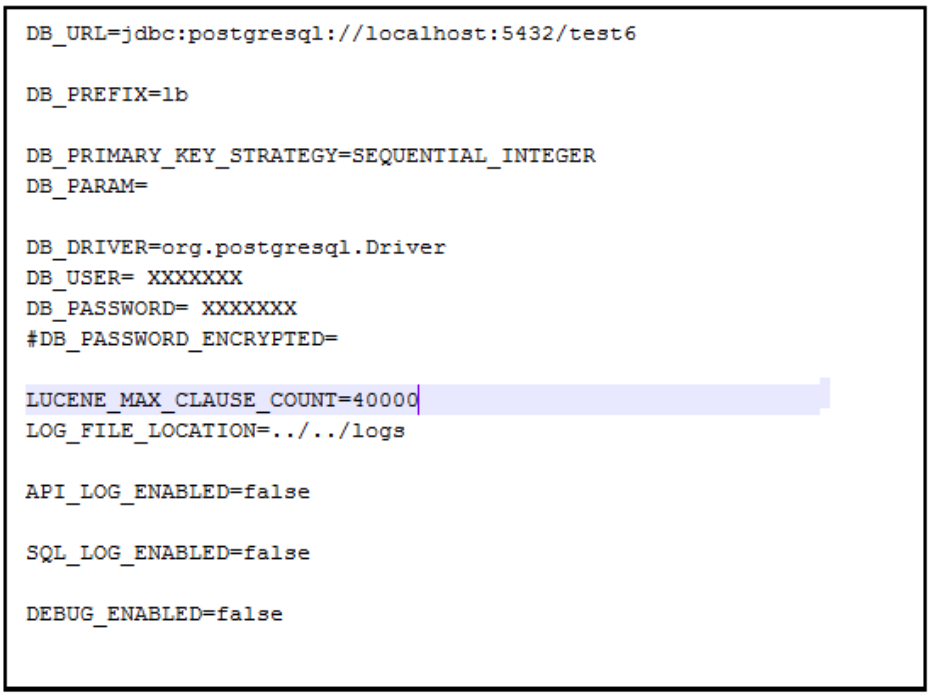


Figure 5.1: Configuration file of our LexEVS server.

---

balance. In the second case, as the name says, the index values are inserted sequentially and it is the best case for large terminologies. We set the value for the Sequential Integer value because of the terminologies size.

- SINGLE TABLE MODE: determines if all the terminologies are loaded into a single set of tables or each terminology per set. Set to false since the mode true can result in long periods of time during the removal of a terminology.

- Log files: The variables related to the log files were set to false to prevent performance delays.

- **Other:**

- MAX RESULT SIZE: the maximum number of results that a user can retrieve from a terminology NodeGraph. Default value is 1000, however in some cases it was not enough, therefore it was set to 10000.

- LUCENE MAX CLAUSE COUNT: Number of clauses that are created due to a query. Large terminologies demand a high number. The value set is the default, 40000, since it was enough.

The strategy chosen was to use the LexEVS server as the main repository of terminologies that are not available in the UMS Metathesaurus. We had to transform the terminologies chosen because most of them are not available in a format compatible with the server. When

the terminologies were correctly loaded into into the system, we had to develop services to access the information. Two main inputs were taken in consideration, the code of the concept and the description. The description as input led to some troubles because of the ambiguity created by words, so we had to crate a solution to this problem that it will be discuss later. We also developed functions to search concepts linked hierarchically to take advantage of all the data available in the server. Each instance of the server comes with a GUI that has all the basic features of the server. This was the main tool to load and validate the terminologies loaded. The loading process works by providing a terminology in the right format. If the terminology passes the validation process of LexEVS is loaded into the system.

The terminologies that were loaded into LexEVS had to be processed to a format accepted in the server. The format used was the text format. The layout of a formal txt file for LexEVS is given at figure 5.2. Each line has a concept that is formed by a code, a description and optionally another value. If a concept has sub-concepts, they have to appear with a tab in the next line. This is an example of a terminology hierarchy in this file format:

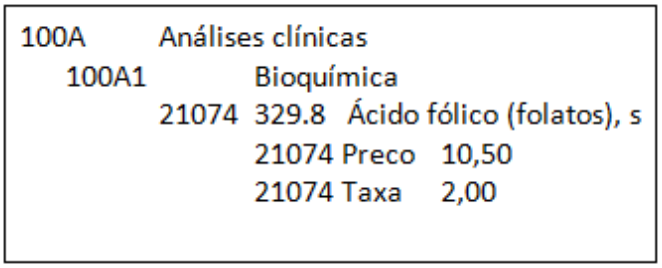


Figure 5.2: Terminology format example.

The previous extract presents five concepts although the concepts regarding terminology data are tree. The line with the tag *Preço* and *Taxa* are attributes of the concept *Ácido fólico (folatos)*. Every concept can have more sub-concepts or attributes. In this representation a line can represent a concept if he has attributes hierarchically beneath them, or a attribute if he is the lowest in the hierarchy. LexEVS GUI has the function to graphically represent terminology data. Data previously showed is present in the system with the format of figure 5.3.

The ICD-9-CM in Portuguese was obtained through the *Administração Central do Sistema de Saúde* [46] in a Excel file. Firstly the information needed was extracted to a text file unformatted. Secondly a file reader was used to process each concept that was parsed to eliminate all the tabs and spaces from each line; finally the data was rearrange in the right format. The steps for this terminology are showed in figure 5.4. The loaded terminology has 17582 concepts.

The ICD-10-CM is not available in a Portuguese source, therefore we used one from an official source of Brasil more specifically from the *Departamento de Informtica do Sistema nico de Sade (DATSUS)*[47]. It was already in text format therefore we used the same process of ICD 9-CM without the translation from excel to text. The source file had to be parsed

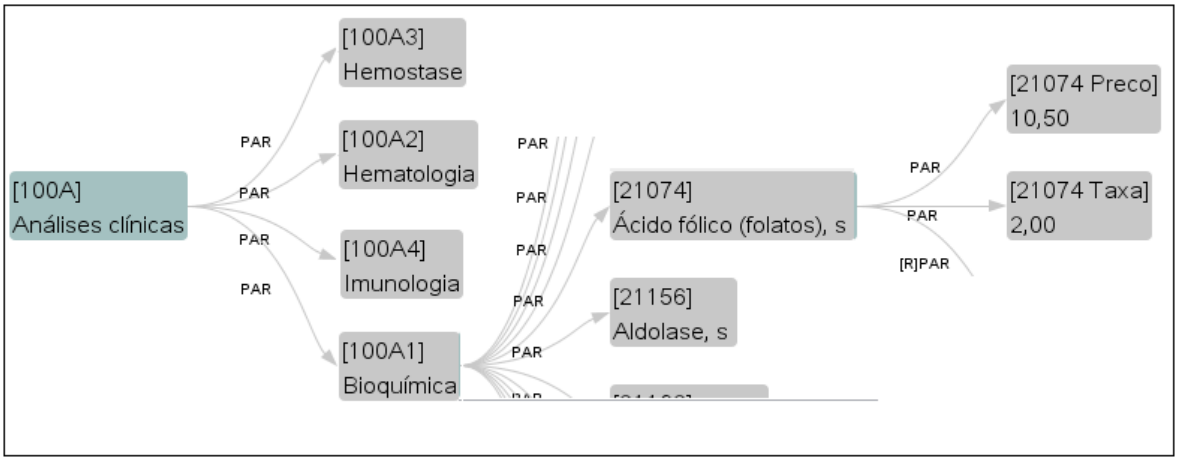
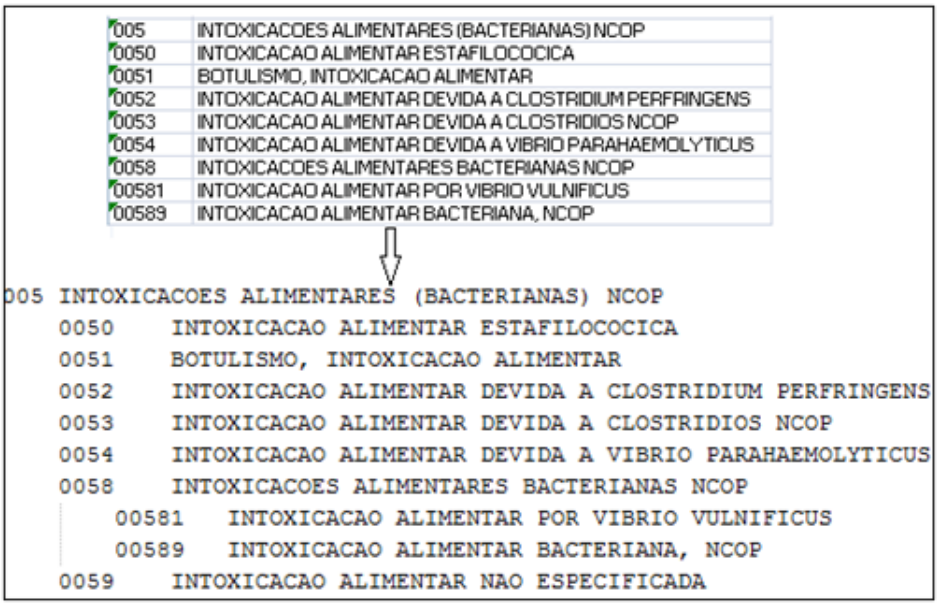


Figure 5.3: LexEVS graphical terminology representation.



005	INTOXICACOES ALIMENTARES (BACTERIANAS) NCOP
0050	INTOXICACAO ALIMENTAR ESTAFILOCOCICA
0051	BOTULISMO, INTOXICACAO ALIMENTAR
0052	INTOXICACAO ALIMENTAR DEVIDA A CLOSTRIDIUM PERFRINGENS
0053	INTOXICACAO ALIMENTAR DEVIDA A CLOSTRIDIOS NCOP
0054	INTOXICACAO ALIMENTAR DEVIDA A VIBRIO PARAHAEMOLYTICUS
0058	INTOXICACOES ALIMENTARES BACTERIANAS NCOP
00581	INTOXICACAO ALIMENTAR POR VIBRIO VULNIFICUS
00589	INTOXICACAO ALIMENTAR BACTERIANA, NCOP

```
005 INTOXICACOES ALIMENTARES (BACTERIANAS) NCOP
    0050    INTOXICACAO ALIMENTAR ESTAFILOCOCICA
    0051    BOTULISMO, INTOXICACAO ALIMENTAR
    0052    INTOXICACAO ALIMENTAR DEVIDA A CLOSTRIDIUM PERFRINGENS
    0053    INTOXICACAO ALIMENTAR DEVIDA A CLOSTRIDIOS NCOP
    0054    INTOXICACAO ALIMENTAR DEVIDA A VIBRIO PARAHAEMOLYTICUS
    0058    INTOXICACOES ALIMENTARES BACTERIANAS NCOP
        00581   INTOXICACAO ALIMENTAR POR VIBRIO VULNIFICUS
        00589   INTOXICACAO ALIMENTAR BACTERIANA, NCOP
    0059    INTOXICACAO ALIMENTAR NAO ESPECIFICADA
```

Figure 5.4: ICD 9 from table to text file with the correct format.

and reformed in a valid format. The loaded terminology has 14126 concepts.

The allergy terminology (*Catlogo Portugus de Alergias e Reaes Adversas*) has the allergies and adverse reactions in Portuguese. It was easy to transform since it had ninety one concepts. The concepts were taken from tables of a pdf file, and written in the right format in a text file. The steps are the one exemplified in figure 5.5.

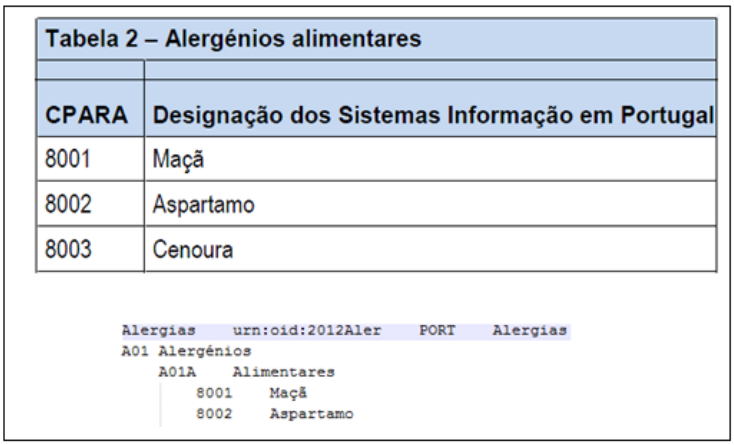


Tabela 2 – Alergénios alimentares

CPARA	Designação dos Sistemas Informação em Portugal
8001	Maçã
8002	Aspartamo
8003	Cenoura

```
Alergias    urn:oid:2012Aler    PORT    Alergias
A01 Alergénios
    A01A    Alimentares
        8001    Maçã
        8002    Aspartamo
```

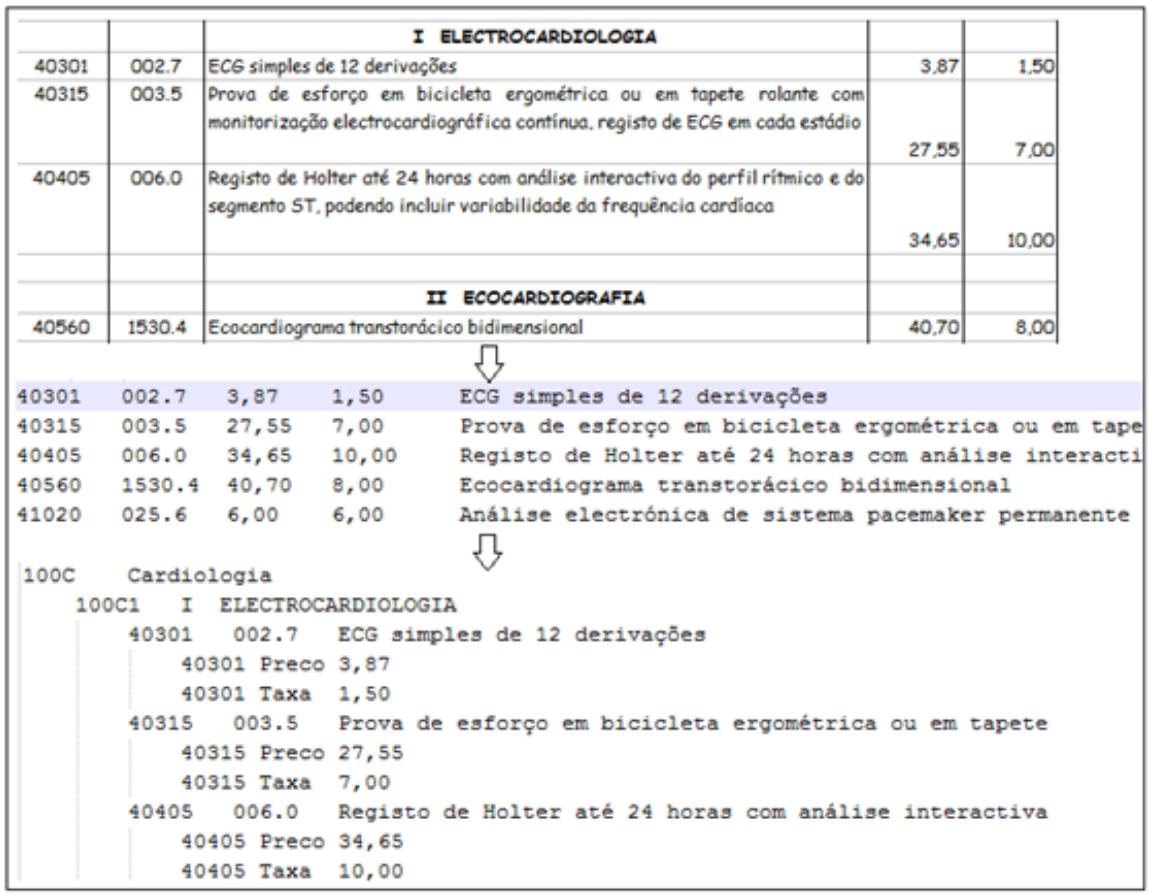
Figure 5.5: Allergies from pdf file to text with correct format.

In the case of the "*Tabela dos convencionados*", there were two codes for each concept in the terminology. Additional information was also present such as price and the taxes. So it was decided to use the format in 5.6. The file was built from the excel file by copying the information into a text file, and then using regular expressions to parse each line and write the file in the right format. The file contains 2035 lines of concepts.

5.1.2 UMLS Metathesaurus

"*The UMLS Terminology Services (UTS) API 2.0 is intended for application developers to perform Web service calls and retrieve UMLS data within their own applications.*'"[48] To use the UTS API 2.0, there are a number of steps needed to install it. It is required to create a directory in which the WSDLs are going to be compiled. Fetch the files from the UTS repository using the wsimport command. Each folder crated has the classes that are needed to use the API [48].

To call the API services the policy of access is "ticket granted". To have access to the API you need to request UMLS for a username and password to access this data. Regarding the development the first step and the most important one is to create a proxy grant ticket, that is valid for 8 hours and is necessary each time a request to the API is made to request a single-use ticket. One single-use ticket grants access to one API call and each time a call is made the ticket used expires. One example of both tickets is given at figure 5.7.



		I ELECTROCARDIOLOGIA		
40301	002.7	ECG simples de 12 derivações	3,87	1,50
40315	003.5	Prova de esforço em bicicleta ergométrica ou em tapete rolante com monitorização electrocardiográfica contínua, registo de ECG em cada estádio	27,55	7,00
40405	006.0	Registo de Holter até 24 horas com análise interactiva do perfil rítmico e do segmento ST, podendo incluir variabilidade da frequência cardíaca	34,65	10,00
		II ECOCARDIOGRAFIA		
40560	1530.4	Ecocardiograma transtorácico bidimensional	40,70	8,00

```
40301   002.7   3,87    1,50      ECG simples de 12 derivações
40315   003.5   27,55   7,00      Prova de esforço em bicicleta ergométrica ou em tape
40405   006.0   34,65   10,00     Registo de Holter até 24 horas com análise interacti
40560   1530.4  40,70   8,00      Ecocardiograma transtorácico bidimensional
41020   025.6   6,00    6,00      Análise electrónica de sistema pacemaker permanente
```

```
100C    Cardiologia
    100C1   I  ELECTROCARDIOLOGIA
        40301   002.7   ECG simples de 12 derivações
            40301 Preco 3,87
            40301 Taxa  1,50
        40315   003.5   Prova de esforço em bicicleta ergométrica ou em tapete
            40315 Preco 27,55
            40315 Taxa  7,00
        40405   006.0   Registo de Holter até 24 horas com análise interactiva
            40405 Preco 34,65
            40405 Taxa  10,00
```

Figure 5.6: From excel to terminology format.

```
private String username = "FernandJ";
private String password = "Mae        ";
private UtsWsSecurityController securityService;
private String ticketGrantingTicket;
private String serviceName = "http://umlsks.nlm.nih.gov";

public UTSTeste() throws UtsSecurity.UtsFault_Exception {
    //initialize the security service
    securityService = (new UtsWsSecurityControllerImplService())
                        .getUtsWsSecurityControllerImplPort();
    ticketGrantingTicket = securityService.getProxyGrantTicket(username, password);
    //use the ticketGrantingTicket to generate a Single Use Ticket with each call
}
private String getProxyTicket() {
    try {
        return securityService.getProxyTicket(ticketGrantingTicket, serviceName);
    } catch (Exception e) {
        return "";
    }
}
```

Figure 5.7: UMLS API ticket granting service.

5.2 MedLexIEETA usage work flow

To exemplify the workflow of MedLexIEETA we will now discuss the most complex scenario that the system supports: the sequence of events to a full disease.

In order to get all the data related to a disease present in the system there are a number of steps that have to be followed. First, decide the input format for the query, being the code of the disease and the source terminology, or, the name of the disease. In figure 5.9, we give an example with the sequence of events when the description is the input format of a query. The description as input brings ambiguity to the request so we had to use a method (i.e. *soundsLike*) to know which disease he wants. What happens is that the system searches the terminologies in the repository calling the function *soundsLike(description)*. This function searches for possible concepts that match, it forms a list with those concepts as *simpleDiseases* (figure 5.8) and return them to the user. The user, according to the result list, chooses the disease concept, the code and codename of the concept are used as input to get the full disease by code (i.e. code of the disease) and sourceCode(i.e. terminology to use). There two data sources for this structure, the LexEVS data and the UMLS Metathesaurus data.

To get the LexEVS data, the system uses a function getRelatedDiseases() that, through a number of steps, searches the concept from which the disease originated (*relatedFrom* of figure 5.8), and the concepts that are related to the disease (*relatedTo* of figure 5.8).

The Metathesaurus data is the data from the terminologies that are related to the disease. The system searches from the concept related to the code that was provided. Concepts in the UMLS Metathesaurus as previously mentioned are the link between related terminology codes of a disease. Subsequently we form a filter to prevent the return of extra information, define the number of results, the languages that are relevant and the terminologies that we use. With this filter the system searches only the atoms of the concept related to the terminologies in use. From the list of atoms that we receive, comes the data related to the terminologies and

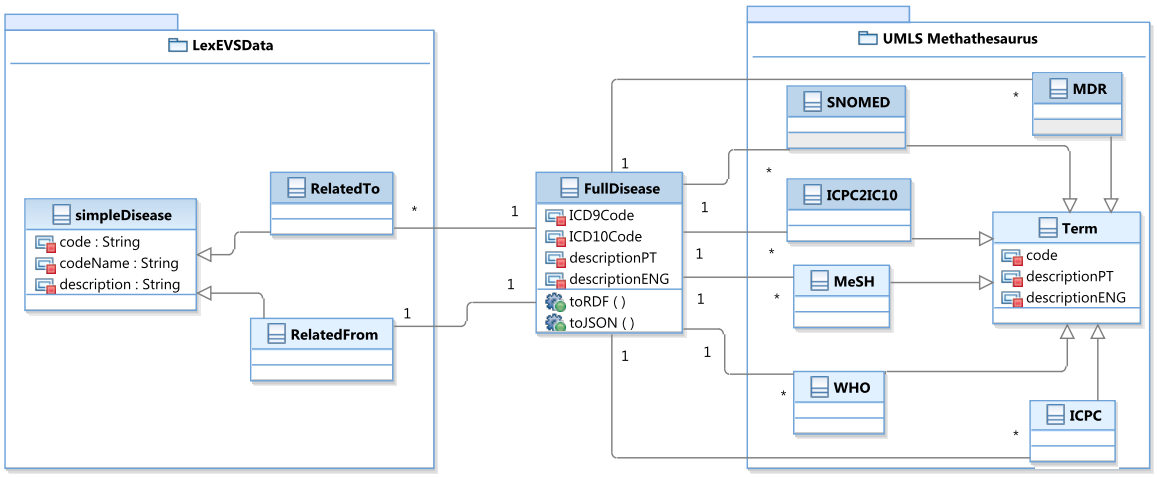


Figure 5.8: Class diagram of a full disease.

---

we store the information in the respective classes. This is how we develop the mapping in MedLexIEETA is done. The mapping between ICD-9-CM and ICD-10-CM is given with the atoms from ICD-9-CM and ICD-10-CM that come from the same concept. This works also for every terminology linked to the concept. The descriptions of the main class *FullDisease* is the description that is available for the ICD terminology that is searched.

For every terminology that is available in both languages filtered (i.e. Portuguese and English) is stored the description in those languages. This is how we developed the translation capability of the system. The amount of data gathered in some cases has more than 40 concepts related to one disease. This gave birth to the need of representing semantic information in a way that was easier for humans to process. RDF is a tool for represent semantic data but it can be difficult to understand if the file dimension is too big. The W3C has a tool [49] to parse and validate RDF documents and although it provides a visual representation of the data as we can see in figure 5.10, RDF representation for human understanding is far from perfect. We decided to extract the information in a format that could be appropriate for a more easier representation. The extraction of this data was done in two formats: RDF and JSON.

5.3 Building of the response

The extraction of major chunks of data was provided in RDF and JSON. In this section it is explained how we developed the transformation from Java classes to the proper output. The data extracted aims to provide terminology dat to application developers by serving as a source of information that can add value to data that already exists.

RDF

The feature used from JENA framework[33] was the API for writing and processing RDF data in XML. Nevertheless the framework also provides an API to handle OWL and RDFs

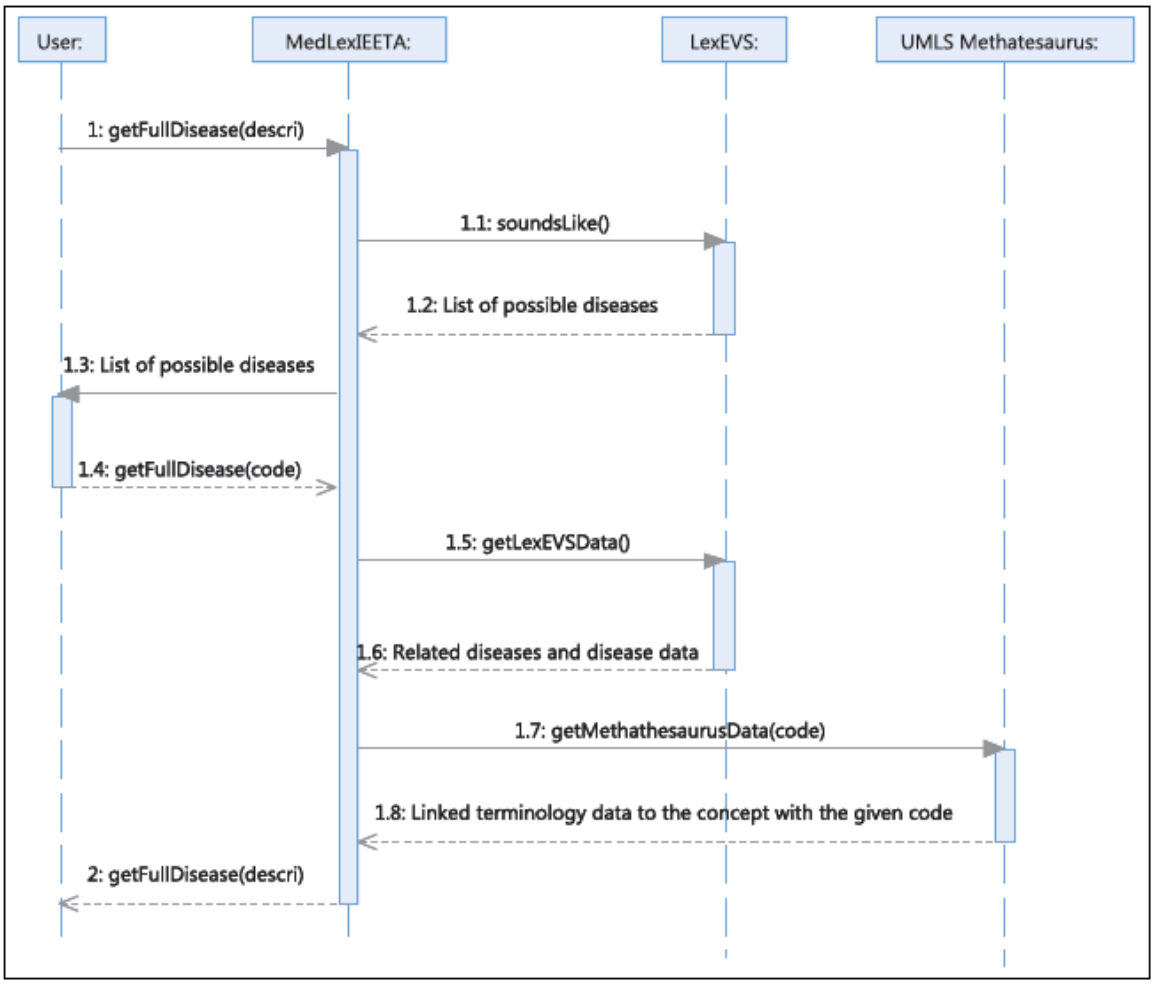


Figure 5.9: Sequence Diagram of a typical system call to get a full disease.

ontologies, SPARQL querying and servers that allows RDF data to be published [33]. The RDF file resulting from a query is extracted after the gathering of data. It is used similar as a toString method of class. The properties (i.e. the link between nodes of a RDF file) are declared and used according to the information available. Then an iterator covers each concept of a terminology and for each concepts creates a node in the RDF file. In figure 5.10 it is possible to see an example of the RDF file resultant of a query to the system and the graphical representation of the data.

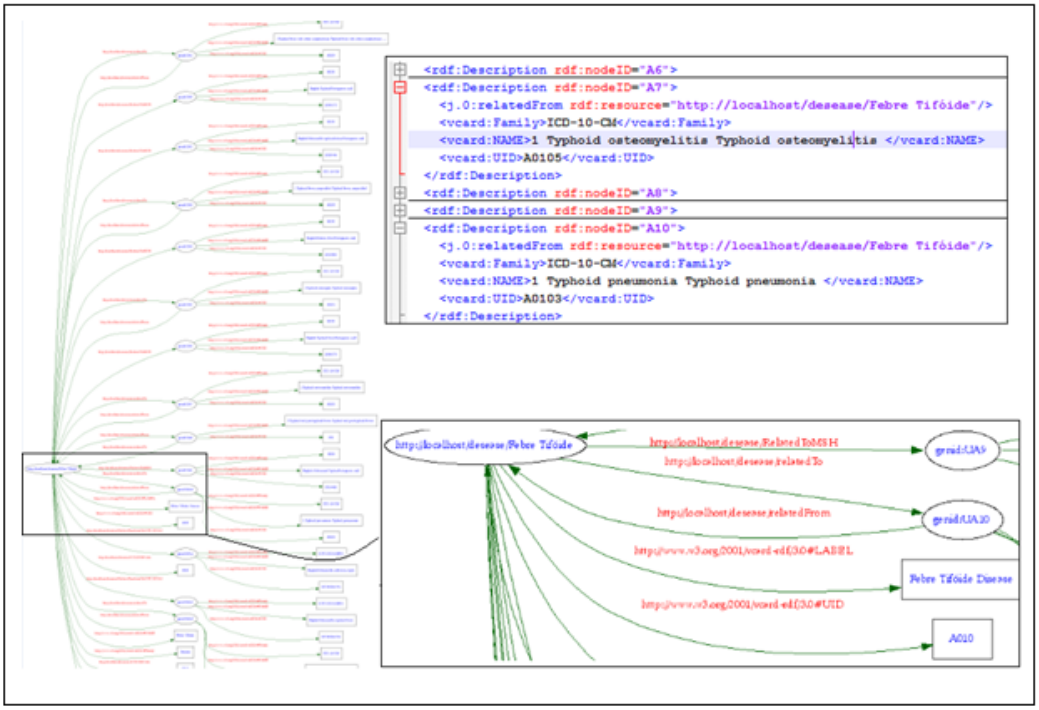


Figure 5.10: RDF response in different formats to represent the amount of information and justifies a graphical representation.

JSON

The JSON structure resulting from a query is extracted after the gathering of data. Resorting to a toString method the concepts are added to a string builder that builds a graph with the information. An iterator covers each concept of a terminology and for each relation or concept we create a node in the JSON response. In [32] says that the main disadvantage of JSON is to represent graphed information. By the experience of doing it, comparing the complexity of representing this structure to simpler ones the difference the work that was evolved was significantly because of the structure of a nodes inside nodes. In figure 5.11 it is possible to see the response to a query in JSON. We develop the response in JSON to user queries. Although the main advantage for the JSON response is the graphical representation of the data since the RDF has the same data. If needed the system could be easily transformed to creat a JSON with all the data present in system. We had to take all the codes from one terminology and create an iterator to run queries for each code. Each resulting concept would

be stored in a List of diseases and then we had to concatenate the respective JSONs.

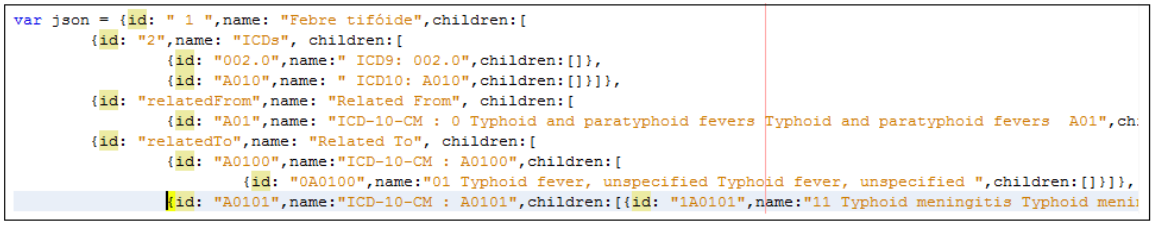


Figure 5.11: First nodes of a response in JSON format.

5.4 Web Service Module

The Web Service Module is the layer of MedLexIEETA regarding the publication of services. A diagram of the services is available in the figure 5.12.

- **GetFullDisease:** This example case was described in section 5.2.
- **GetCodebyDescription:** This is the previous step of the example case described in 5.2, in case of the user does not have the code related to a disease. The system was developed to retrieve the possible matches and their codes. This service uses LexEVS infrastructure and the Portuguese ICD-9-CM or ICD-10-CM.
- **GetRelatedDiseases:** LexEVS provides functionalities to find the concepts related to a provided concept. From this list of related concepts it was necessary to filter the concept from which the concept originated if it exists, and the concepts that are originated from. The arguments are the code of the disease.
- **GetConvencionados:** The Convencionados terminology has the peculiarity of, for each term it has two codes instead of one. This service searches the terminology for one of the codes and retrieves the concept related with the code provided. The input argument is one concept code. The output is the concept in JSON, RDF or literal formats.
- **GetAllergies:** Service that grants access to the allergy terminology loaded on LexEVS. It is the basic service of retrieving a description based in the code provided by the user. The argument provided is the code of the allergy. The output is the related description if available.
- **GetICD9description:** Service that grants access to the ICD-9-CM terminology in Portuguese loaded in LexEVS. It is the basic service of retrieving a description based on the code provided by the user. The arguments provided are the code of the ICD-9-CM and the language of the terminology . The output is the related description if available.
- **GetICD10description:** Service that grants access to the ICD-10-CM terminology in Portuguese loaded in LexEVS. It is the basic service of retrieving a description based on

the code provided by the user. The arguments provided are the code of the ICD-10-CM and the language of the terminology. The output is the related description if available.

- **GetICD10byICD9/GetICD9byICD10/ GetICPC2ICD10:** Services that provide the transcoding functionality between the ICD-9-CM, ICD-10-CM and ICPC2ICD10. The services return the relative code that matches the other terminology. The input argument is the code of the disease. The output is the mapped code.

- **GetTermData:** Get data from the other terminologies present in the UMLS Metathesaurus. This service uses the same structures of the case discussed in 5.2 but returns only terminology data from UMLS. The input argument are the code of ICD-9-CM and a flag to signals that the code is from ICD-9-CM, or, the code of ICD-10-CM and the flag is set to ICD-10-CM. The outputs are JSON or RDF.

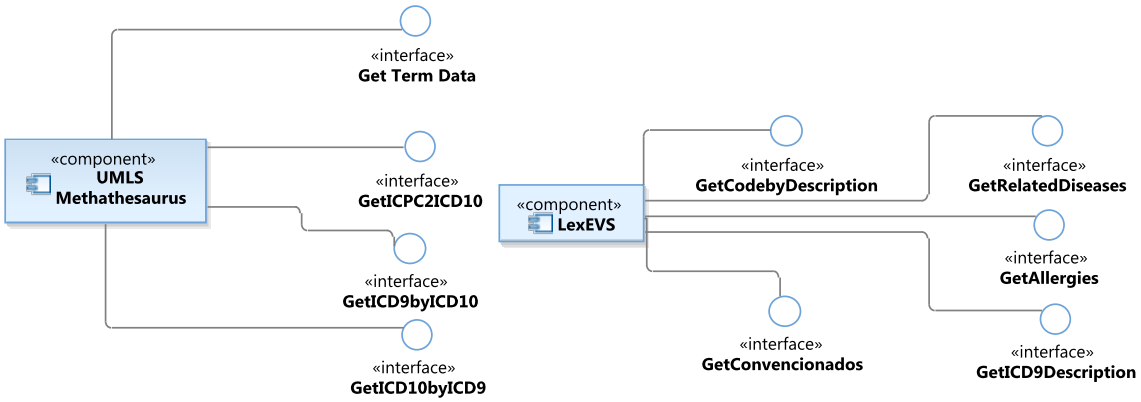


Figure 5.12: Services provided by MedLexIEETA.

5.5 Visualization module

This module consists of a web-site that graphically represents the data. It serves also as a client representing a web service client using the services provided by MedLexIEETA. It was developed using jQuery, javascript for the web pages and for the graphical representation based on JSON it was used the Jit library.

In figures 5.13 and 5.14 are the pages represented for the search engine. The main functionality of this web-site is to search the full diseases mentioned. The main page provides a search field, that is the input used to prompt the system. Through the web service the service *GetFullDisease* is invoked. The result page displays the first match returned and the possibility to change to other concept. The *Jit* representation is focused in the main description of the disease and it has 3 levels of hierarchy. The user reads the graph from the inside out being the graph center the disease description. The first level corresponds to the terminology that is related to the main concept. The second level has the related code to the terminology that it is linked. The third level has the English and Portuguese descriptions related to the code of the related terminology.

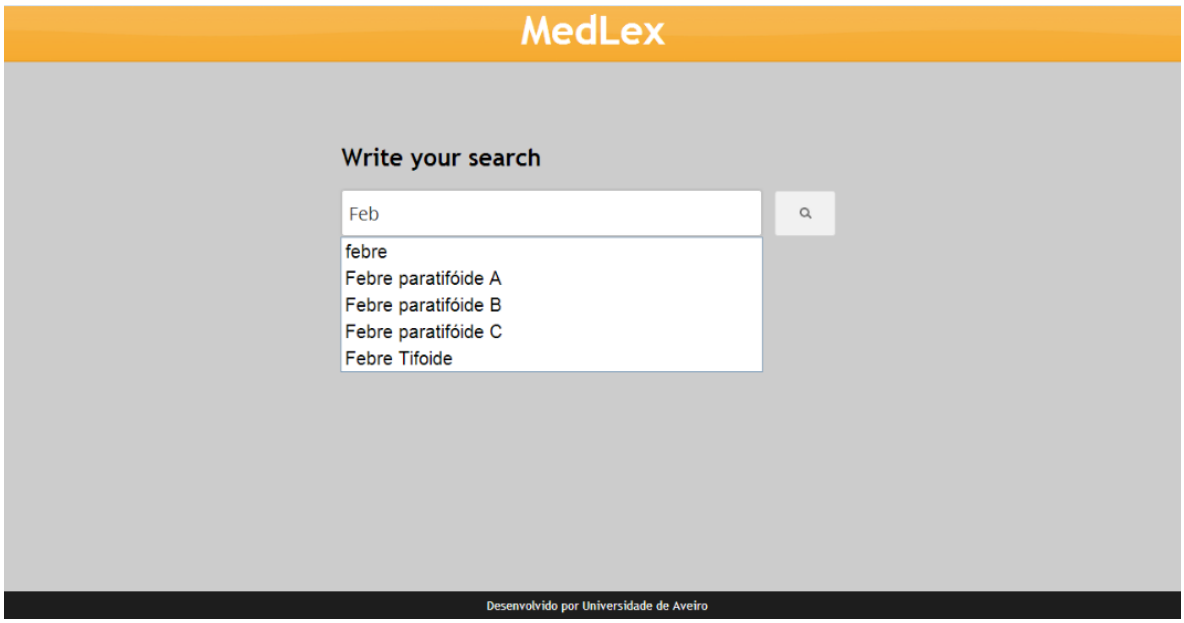


Figure 5.13: Index of MedLexIEETA website.

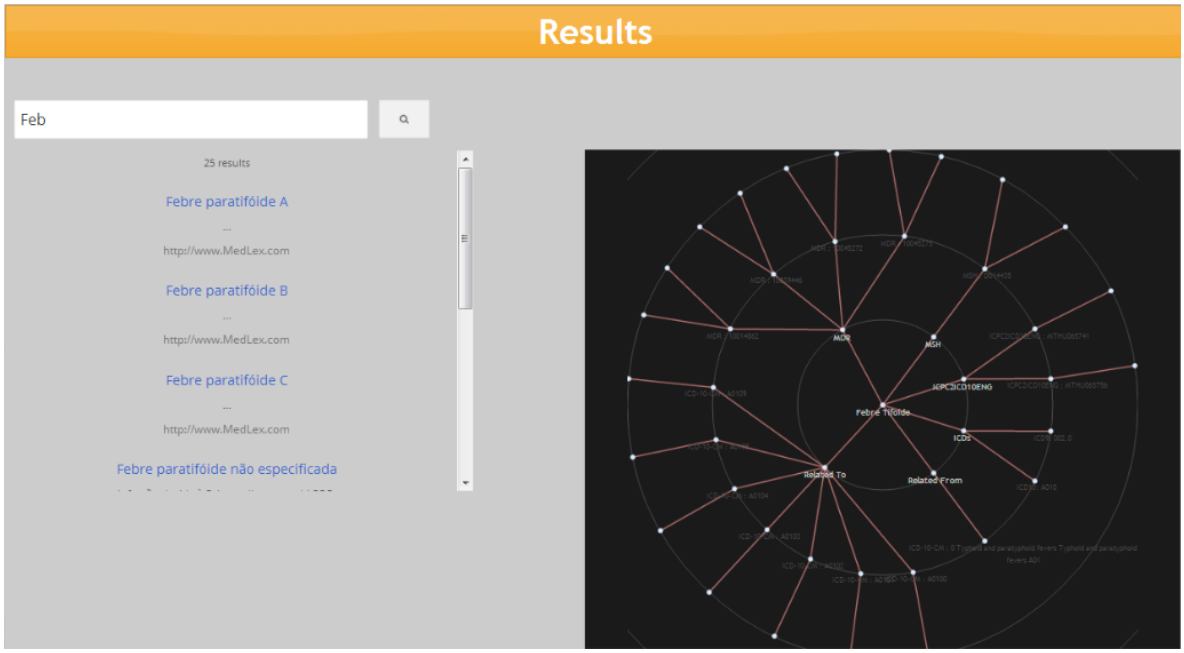


Figure 5.14: Graphicall representation and results from the service.

5.6 MedLexIEETA for applications

One of the goals of MedLexIEETA is to serve as a support for application development. The services provided offer an added value to eHealth data.

An example in which the system could prove useful would be to assist in preparing the epSOS documents from an existing EHR system in Portugal. There are two feasible scenarios in which MedLexIEETA can enter: to adapt previous records of patients episodes by analysing the text document and finding key data (e.g. disease, treatments and medicine), this could be accomplished if firstly the documents went through a solution identical to the work developed in [16] passing subsequently through the MedLexIEETA services to insert the right codes that match the text reference. The second case would be in the record of a patient episode, the input of information must be done according to the terminologies that would be used (e.g. epSOS codification for blood group is the codification of SNOMED-CT). For this to happen a significant improvement had to be done on MedLexIEETA to increase the amount of Portuguese terminologies and most importantly the mapping between these terminologies and the ones that are used by epSOS.

Chapter 6

Results

In this chapter we present the results of MedLexIEETA, considering the services and the functionalities that the system provides.

In section 2.3 we defined five essentials characteristics for a terminology server. Let us see if those services are available in MedLexIEETA. Firstly the management of external references is available through the web services that enables support for the development of clinical applications. Secondly the management of internal references is granted by the capacity of the system to provide a extensible basis to all the terminologies present in it. The mapping of natural language to concepts is available trough the functionality of the system give possible matches for the required input and display the options concerning concepts and give the relative concept code. Every concept has attached to it the corresponding coding schemes and codes making available the fourth characteristic. Finally the management of extrinsic information is provided by the capacity MedLexIEETA to load and use new terminologies.

The core of the system is the LexEVS instance, for the Portuguese terminologies and translations of terminologies, and the UMLS API that resolves the mapping and handles the Terminologies that are available and are relevant to the project.

Available Code Systems

Code System Name	Code System Vers...	URI	Tag	Status	Last Update Time
ICD-9	CM	urn:oid:2.16.840.1.113883.6.1		active	6:32:33 PM on 09/06/2012
ICD-10CM	Volume	urn:oid:2.16.840.1.113883.6.2		active	6:32:47 PM on 09/06/2012
Alergias	1.0	urn:oid:2012		active	6:32:16 PM on 09/06/2012
Convencionados8	1.0##PT	urn:oid:20128		active	2:38:48 PM on 09/11/2012

Figure 6.1: LexEVS available terminologies.

The LexEVS instance was loaded with the Portuguese ICD 9 and ICD 10, the allergies codification and the "*Tabela dos Convencionados*". Part of the GUI available in our LexEVS instance is available in figure 6.1. A representation of the data available through the GUI is showed in 6.2.

The main goals achieved with the terminology server of the system were the use of Portuguese terminologies and the translation that come with the Portuguese version of ICD 9 CM

and ICD 10 CM. The first goal achieved helps to demonstrate the capability of the system to handle local and national problems related to codifications. The "*Tabela dos Convencionados*" problem that was resolved was the existence of two codes for the same concept. As highlighted in figure 6.2 the codes are easily available and mapped to each other. The second goal demonstrates a solution to the translation of concepts between different languages that use the same terminologies. The concept the we proved was that if the system has access to twenty or more translations of the same terminology the visualization of that data can be automatically represented in a specific language.

We developed two methods to translate concepts. The first was the mapping derived from the UMLS API in combination with the Portuguese versions of the ICD 9 and ICD 10. In 6.3a and 6.3b is a demonstration of the system capability to translate concept descriptions trough the first method.

The second method developed was to use terminologies in which both versions (i.e. the Portuguese and English versions) were available in the Metathesaurus and for a specific concept, retrieve both definitions. For the second method it was only used the Portuguese and English languages for the available terminologies in the Metathesaurus of UMLS. In 6.3c is present a translation of the same concept of the MDR terminology and in 6.3d a translation of a concept of the MSH terminology.

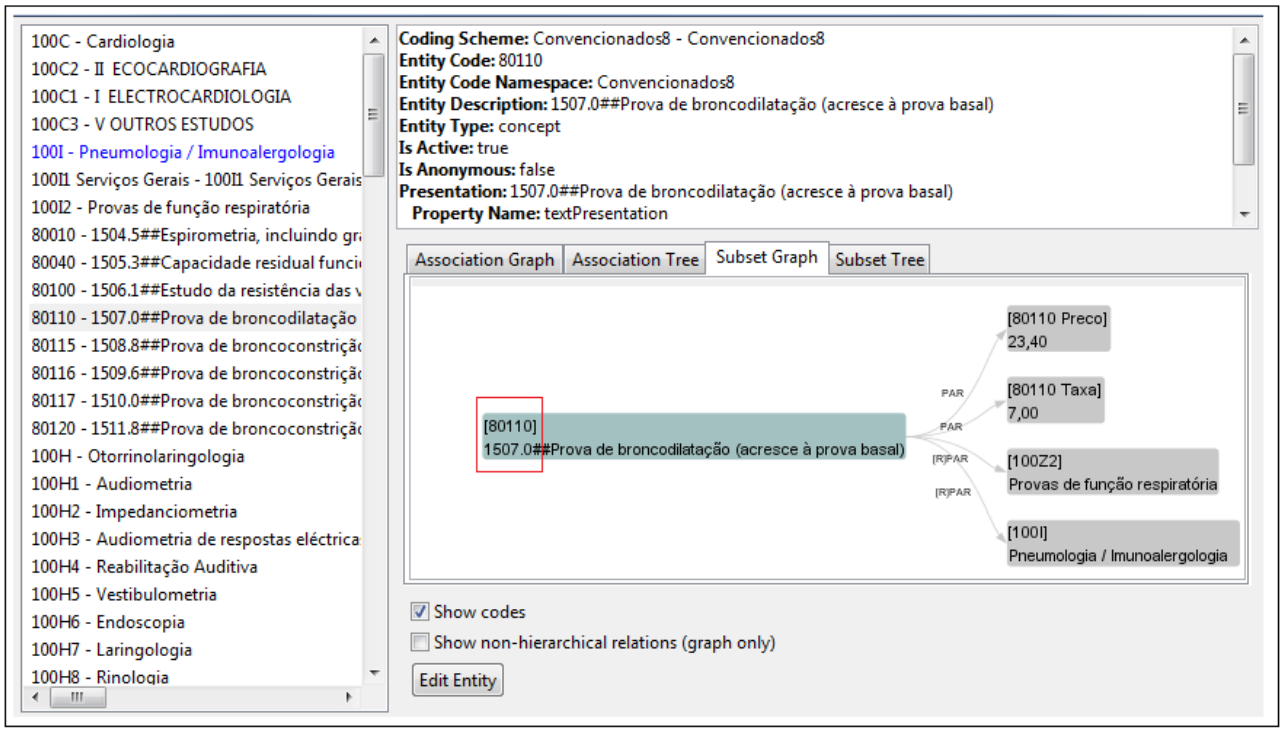
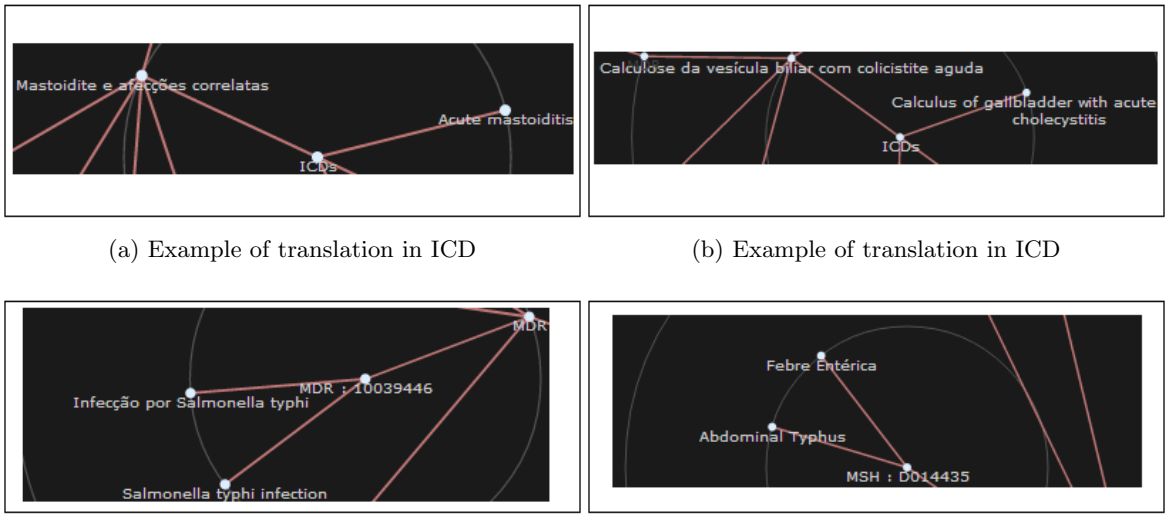


Figure 6.2: LexEVS GUI representing the "*Convencionados*" terminology.

Through the analyses of a result of a query it is possible to see that two characteristics are provided by the system, translation and transcoding. In figure 6.3c it is possible to see the translation languages regarding the same concept of the terminology MDR.

The translation problem was resolved with the LexEVS Server and the UMLS API. How-



(a) Example of translation in ICD

(b) Example of translation in ICD

(c) Example of translation in MDR

(d) Example of translation in MSH

Figure 6.3: Examples of translation in MedLexIEETA.

---

ever the transcoding problem was resolved using only the second one. For each concept requested, the system retrieves the data related to each terminology used. This capability represents a solution for the transcoding problems of Portugal regrading the transition from ICD 9 to ICD 10. Therefore it can be applied to the semantic problems of the project epSOS. SNOMED-CT is one of the most "*comprehensive, multilingual clinical terminology in the world*"[50] so it made sense to involve this terminology in the solution of the system.

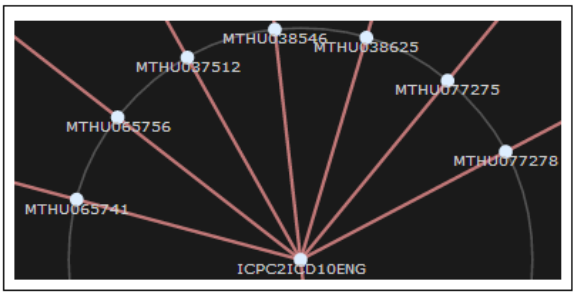
In figure 6.4 is possible to see the mapping that is available for a specific disease, 6.4a shows the concepts of ICPC2ICD10, the figure 6.4b has the mapping between ICD 9 and ICD 10 and in 6.4c are the concepts related to the terminology SNOMED-CT.

In the figure 6.5 it is displayed all the data related to a disease retrieved from MedLEXIIETA.

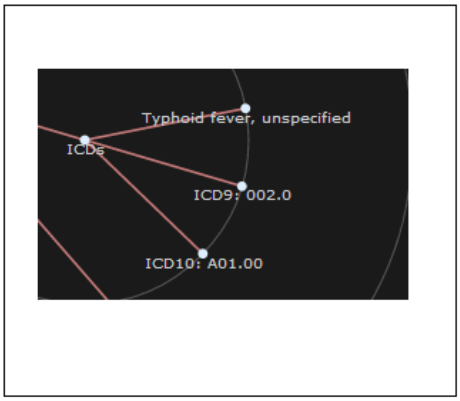
From the figures 5.10, and 6.5 we can see the advantage of a visual representation to support data extraction. Although the JSON format was more difficult to implement, the RDF format was not enough to offer easy knowledge to users without access to tools that provide the ability to process RDF data.

The web-site as a client

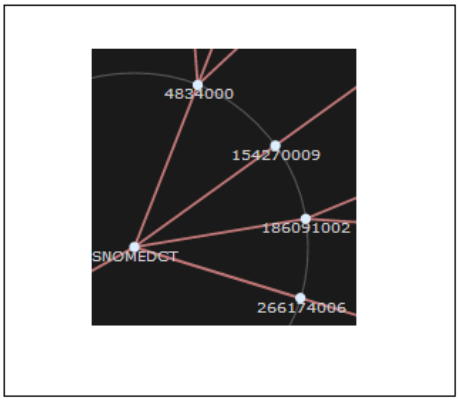
The web site was developed as the main interface to users of MedLexIEETA. It uses the services provided by the terminology module. The communication between client and server was developed using a REST server and worked correctly, and no major communication problems were encountered.



(a) ICPC2ICD10 concepts



(b) ICD 9 and ICD 10 codes



(c) SNOMED concepts

Figure 6.4: Transcoding between SNOMED-CT, ICD 10, ICD 9 and ICPC2ICD10.

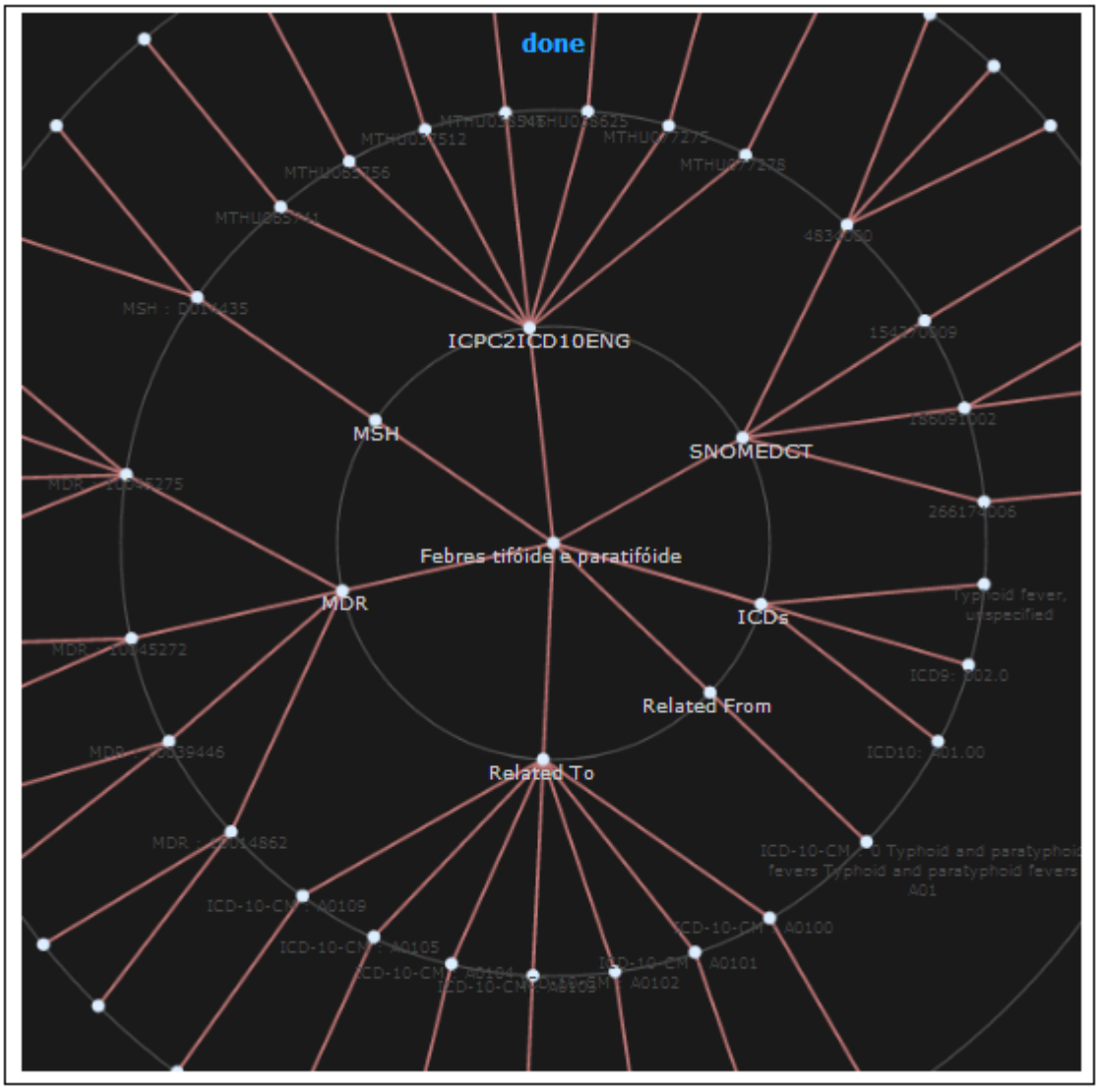


Figure 6.5: Final graphical representation of MedLexIEETA data.

Chapter 7

Conclusions

7.1 Accomplishments

Departing from the need of a tool that enables the use of terminology related information to develop services that extract more functionality from that same data we have successfully built a system that contains terminologies from the Portuguese health reality and provides transcoding of concepts between terminologies and translation from concepts in different languages. The solution builds on the LexEVS module, used in large biomedical projects, which was customized and loaded with terminologies in Portuguese that were not available at UMLS Metathesaurus. This module holds the ICD-9-CM in Portuguese, ICD-10-CM in Portuguese, "*Tabela dos Convencionados* and the allergies terminologies used and is the system core since it holds the terminologies used to filter and process Portuguese terminology related functions. It was developed to match the needs of the Portuguese reality.

The remote UMLS Metathesaurus enables the use of terminology data and the most important feature it allows access to mapping between terminologies. The transcoding and translate function of the system comes from the mapping between terminologies (e.g. transcoding) and the different descriptions for the same concept (e.g. translation).

The main objective of the project discussed in this dissertation was the use of a vocabulary server as a central component to enable two motivating use cases: (1) enable a reference semantic service for the Portuguese reality and (2) enable the transformation of clinical data structures into other clinical models (for interoperability scenarios). Regarding the first case the LexEVS instance of MedLexIEETA works as the repository of Portuguese specific terminologies and provides semantic services related to these terminologies. The second case was in part resolved through the use of the mapping system of UMLS Metathesaurus. The system can transform or give correspondences between terminologies and provide translation of concepts between different languages.

As a proof of concept, we have developed the website MedLexIEETA to enable the users to browse terminology data and the graph of information retrieved from a query to the system core data. The system also provides services for application developers, allowing the high-level querying of terminology related services to further reutilize the functionalities provided by the system.

7.2 Issues

The main issues regarding the development of this project were related to the difficulty to understand the semantic tools, the number of concepts that were studied and the lack of proper terminology data.

When taking the first steps through both semantic tools (i.e. the LexEVS instance and the UTS API), the learning process was slow and the adaptation to this area of knowledge was hard, mainly because of the lack of semantic studies in my university years.

Regarding the terminologies, there are so many formats to publish terminology data that most of them are completely useless to software developers. The time for reading terminologies is over, so there is no point in publishing one in pdf format (e.g. the allergy terminology present in the system). The spending of time treating a terminology to fill the standards of LexEVS was a variable that I was not counting.

The difficulties felt from the lack of Portuguese terminologies that could actually be mapped to most relevant international terminologies and that gave a significant improvement in eHealth in Portugal prevented me from develop a case in which a mapping between a global terminology and a Portuguese one was done.

7.3 Future Work

For future work in this project I advise to take MedLexIEETA to the next level. Meaning that the system has to take a patient EHR from RTS and translate words to codes what could lead to a solution that could even be used in the epSOS project if needed. Another tool that could be developed using the system is the mapping between diseases and drugs/medicines. I think that the uses that could be given to such a service such as mobile applications and diagnosis tools could be a great improvement in the eHealth reality.

To do this the system must use the mapping between the terminologies used to encode the information in RTS and translate the epSOS terminologies. This will not always be easy because most of the codifications used are not in a format that can be easily transformed to a terminology. Clinical related projects are emerging and it is creating a problem between the systems developed since the every project wants to lead and not to work together. It is like a bridge being built from the two sides of the river but the connection is not correctly done.

The solution developed to update terminologies can be improved, since the mechanism developed in the case of small changes in a terminology does not justify the load of a new file. Therefore a update tool that can interpret a file or some kind of input with the necessary changes.

More terminologies can be loaded into the terminology server. The more terminologies and controlled coding schemes that are used in eHealth, more improved is the quality of the EHR records, meaning, more terminologies less ambiguity.

References

[1] O. Kilic and A. Dogac. Achieving clinical statement interoperability using r-MIM and archetype-based semantic transformations. *IEEE Transactions on Information Technology in Biomedicine*, 13(4):467–477, July 2009.

[2] F. Amato, A.R. Fasolino, A. Mazzeo, V. Moscato, A. Picariello, S. Romano, and P. Tramontana. Ensuring semantic interoperability for e-health applications. In *2011 International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, pages 315–320, July 2011.

[3] Rede Telematica de Saude. RTSaude. http://www.rtsaude.pt/. Accessed in 2012-10-03.

[4] I. C. Oliveira and J. P. S. Cunha. "integration services to enable regional shared electronic health records". User Centred Networked Health Care - Proceedings of MIE 2011, pages 310–314, Oslo, Norway, 2011.

[5] Smart Open Services for European Patients. epSOS: home. http://www.epsos.eu/home.html. Accessed in 2012-10-31.

[6] Portal da Saude. Portal da saude - epSOS - portugal. http://www.portaldasaude.pt/portal. Accessed in 2012-11-18.

[7] M. Argello, J. Des, R. Perez, M.J. Fernandez-Prieto, and H. Paniagua. Electronic health records (EHRs) standards and the semantic edge: A case study of visualising clinical information from EHRs. pages 485–490. IEEE, 2009.

[8] Anne Geraci. *IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries*. IEEE Press, Piscataway, NJ, USA, 1991.

[9] George A Komatsoulis, Denise B Warzel, Francis W Hartel, Krishnakant Shanbhag, Ram Chilukuri, Gilberto Fragoso, Sherri de Coronado, Dianne M Reeves, Jillaine B Hadfield, Christophe Ludet, and Peter A Covitz. caCORE version 3: Implementation of a model driven, service-oriented architecture for semantic interoperability. *Journal of biomedical informatics*, 41(1):106–123, February 2008.

[10] I. Berges, J. Bermudez, and A. Illarramendi. Toward semantic interoperability of electronic health records. *IEEE Transactions on Information Technology in Biomedicine*, 16(3):424–431, May 2012.

[11] Vell N. et al. Stroetmann. Semantic interoperability for better health and safer healthcare. Technical report, Office for Official Publications of the European Communities, Luxembourg, 2009.

[12] The Comission of the European Communities. Commission recommendation of 2 july 2008 on cross-border interoperability of electronic health record systems, July 2008.

[13] Catalina Martnez-Costa, Marcos Menrguez-Tortosa, and Jesualdo Toms Fernndez-Breis. An approach for the semantic interoperability of ISO EN 13606 and OpenEHR archetypes. *J. of Biomedical Informatics*, 43(5):736–746, October 2010.

[14] Chris Wroe. Is semantic web technology ready for healthcare? *Paper presented at the 3rd Eur. Semant.Web Conf., Budva, Montenegro*, June 2006.

[15] J. Lahteenmaki, J. Leppanen, and H. Kaijanranta. Interoperability of personal health records. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009. EMBC 2009*, pages 1726–1729, September 2009.

[16] Liliana da Silva Ferreira. *Medical information extraction in European Portuguese*. PhD thesis, Universidade de Aveiro, July 2011.

[17] Tim Berners-Lee and Mark Fischetti. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. DIANE Publishing Company, 2001.

[18] Erdogan Dogdu. Semantic web in eHealth. In *Proceedings of the 47th Annual Southeast Regional Conference*, ACM-SE 47, pages 73:1–73:4, New York, NY, USA, 2009. ACM.

[19] World Wide Web Consortium. Extensible markup language (XML). http://www.w3.org/XML/. Accessed in 2012-10-24.

[20] Jesse James Garret. Ajax: A new approach to web applications - adaptive path. http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications. Accessed in 2012-10-25.

[21] World Wide Web Consortium. OWL web ontology language overview. http://www.w3.org/TR/owl-features/. Accessed in 2012-10-25.

[22] World Wide Web Consortium. Resource description framework (RDF): concepts and abstract syntax. http://www.w3.org/TR/rdf-concepts/. Accessed in 2012-10-24.

[23] World Wide Web Consortium. SPARQL query language for RDF. http://www.w3.org/TR/rdf-sparql-query/. Accessed in 2012-10-25.

[24] World Wide Web Consortium. OWL web ontology language reference. http://www.w3.org/TR/owl-ref/. Accessed in 2012-10-24.

[25] openEHR Foundation. openEHR - what is openEHR? http://www.openehr.org/what_is_openehr. Accessed in 2013-01-06.

[26] EN 13606 Association. The CEN/ISO 13606 association site. http://www.en13606.org/. Accessed in 2013-01-06.

[27] C. Brewster and K. O'Hara. Knowledge representation with ontologies: the present and future. *IEEE Intelligent Systems*, 19(1):72–81, February 2004.

[28] James Z. Wang, Farha Ali, and Pradip K. Srimani. An efficient method to measure the semantic similarity of ontologies. *International Journal of Pervasive Computing and Communications*, 6(1):88–103, April 2010.

[29] World Wide Web Consortium. OWL concepts. http://docs.oracle.com/cd/E18283_01/appdev.112/e11828/owl_concepts.htm. Accessed in 2012-10-25.

[30] World Wide Web Consortium. Web services glossary. http://www.w3.org/TR/ws-gloss/. Accessed in 2012-10-25.

[31] Thomas Erl. *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[32] C. Severance. Discovering JavaScript object notation. *Computer*, 45(4):6–8, April 2012.

[33] Apache. Apache jena - apache jena. http://jena.apache.org/. Accessed in 2012-10-25.

[34] World Health Organization. WHO | international classification of diseases (ICD). http://www.who.int/classifications/icd/en/. Accessed in 2012-10-25.

[35] Neil Barrett and Jens H. Weber-Jahnke. eHealth interoperability with web-based medical terminology services - a study of service requirements and maturity. *Journal of Emerging Technologies in Web Intelligence*, 1(2), November 2009.

[36] Mohammad Nazir Ahmad and Robert M. Colomb. Managing ontologies: a comparative study of ontology servers. In *Proceedings of the eighteenth conference on Australasian database - Volume 63*, ADC '07, page 1322, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc.

[37] Regenstrief Institute. Loinc background. http://loinc.org/background. Accessed in 2012-11-19.

[38] National Library of Medecine. 2012AA SNOMED CT source information. http://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/SNOMEDCT/. Accessed in 2012-11-06.

[39] A. L. Rector, W. D. Solomon, Wd Solomon, T. W. Rush, W. A. Nowlan, and Tw Rush. *A Terminology Server For Medical Language and Medical Information Systems*. 1994.

[40] C. G. Chute, P. L. Elkin, D. D. Sherertz, and M. S. Tuttle. Desiderata for a clinical terminology server. *Proceedings of the AMIA Symposium*, pages 42–46, 1999.

[41] Lim Choi Keung, Sarah N, Lei Zhao, Edward Tyler, F. D. Richard Hobbs, and Theodoros N. Arvanitis. Integrated vocabulary service for health data interoperability. pages 124–127. eTELEMED 2012 Conference, January 2012.

[42] National Cancer Institute. LexEVS 6.0 local runtime configuration file settings - EVS - LexEVS - national cancer institute - confluence wiki. https://wiki.nci.nih.gov/display/LexEVS/LexEVS+6.0+Local. Accessed in 2012-10-25.

[43] Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(Database issue):D267–D270, January 2004.

[44] J J Cimino and X Zhu. The practical impact of ontologies on biomedical informatics. *Yearbook of medical informatics*, pages 124–135, 2006.

[45] Oracle. RESTful web services. http://www.oracle.com/technetwork/articles/javase/index-137171.html. Accessed in 2012-10-25.

[46] Administracao Central do Sistema de Saude. Codificacao clinica hospitalar. http://www.acss.min-saude.pt/ Accessed in 2013-01-06.

[47] DATASUS. CID-10. http://www.datasus.gov.br/cid10/V2008/cid10.htm. Accessed in 2013-01-06.

[48] Unified Medical Language System. UMLS terminology services – home. https://uts.nlm.nih.gov/home.html#apidocumentation. Accessed in 2012-10-25.

[49] World Wide Web Consortium. W3C RDF validation service. http://www.w3.org/RDF/Validator/. Accessed in 2012-10-25.

[50] International Health Terminology Standard Development Organisation. Why use SNOMED CT? http://www.ihtsdo.org/snomed-ct/whysnomedct/. Accessed in 2012-11-07.