Universidad Politécnica
de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

AUTOMATED MAPPING PROCESS FOR CREATING SKOS THESAURI

Author:

Mario De Lucas García

Director:

Víctor Rodríguez Doncel
Departamento de Inteligencia Artificial

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Madrid, Mayo 2022

Trabajo Fin de Grado
Grado en Ingeniería Informática
Título: Automated mapping process for creating SKOS thesauri

Autor: Mario De Lucas García
Tutor: Víctor Rodríguez Doncel
Departamento de Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

If you want to encourage someone,
to do something, make it easy.
Richard Thaler

Agradecimientos

A Víctor, por creer en esta idea. Sin tu gran ayuda este trabajo no sería posible.
A mis padres, por hacerme ver que el trabajo es lo único que abre una puerta.
A mis abuelos, por todo eso que las palabras no pueden explicar.
A David, por ser luz y guía en los buenos y malos momentos.
A Lucas, Sergio y Damián, amigos y referentes, por ser siempre los primeros y los más importantes.
A Beñat, por aparecer y quedarte. Eres hogar.
A María, por dar el doble de lo que recibe.
A Jessica, lo prometido es deuda.
En definitiva, a todos los que me acompañaron en este viaje de cincos años, gracias.

Contents

List of Figures

List of Tables

Abstract

The field of Semantic Web was born with the aim of changing the way the data was processed. This new paradigm does not focus on the data itself, but rather on the *knowledge* contained, where the main objective is to link the data to create an open linked data knowledge base. Throughout this process, the elements that take on importance are the language used to represent these data (Resource Description Framework, RDF), the standardised vocabularies (e.g., FOAF, DCTERMS, SKOS, etc.) and SPARQL, the query language for consulting the knowledge.

Nonetheless, the data that can be published on the Web must be built according to the above elements. Therefore, traditional data that already exist may be transformed into data that follow the semantic web paradigm. In this stage of the process is where the focus of the project is fixed.

Throughout the document, we will study the state-of-the-art of the mapping languages, especially RML and R2RML. Besides, we will review the specification of the SKOS vocabulary. Both elements are the fundamentals of the development part of this project. In that part, we will show **easySKOS**, a web service that transforms semi-structured data in CSV or XLS format into SKOS thesaurus, following the principles of the semantic web. This tool provides templates covering the most popular SKOS thesauri. Then, the user just *drags&drops* the template and **easySKOS** converts it into SKOS thesaurus.

This project is intended to help the expansion of the web semantic by offering an easy-to-use and suitable-for-all-audiences tool for creating SKOS thesauri.

Keywords Semantic Web, Linked data, Mapping languages, RML, SKOS, Thesauri

Resumen

El área de la Web Semántica nació de la necesidad de cambiar la manera en la que se trataban los datos. En este nuevo paradigma no se habla de datos, sino de 'conocimiento' (*knowledge*), donde el objetivo último es conectar los datos que hasta ahora eran atómicos, creando una gran red de datos enlazados (*linked data*) para facilitar su aprovechamiento. En este proceso, los elementos que toman importancia son el lenguaje utilizado para representar estos datos (Resource Description Framework, RDF), los vocabularios estandarizados (e.g., FOAF, DCTERMS, SKOS, etc.) para enriquecer los datos y el lenguaje para la consulta de información en RDF conocido como SPARQL.

Sin embargo, para que ese proceso se pueda poner en funcionamiento es necesario transformar los datos tradicionales en datos que utilicen esos elementos mencionados para que puedan ser aprovechables en el contexto de la web semántica. Es en esta etapa inicial de transformación de datos donde sitúa el foco este proyecto.

A lo largo de este documento se estudia el estado del arte de los lenguajes de mapeos existentes, especialmente RML y R2RML, así como el vocabulario SKOS. Estos dos elementos serán los actores que intervendrán en la parte práctica, que consiste en un servicio web donde se permita transformar datos semi estructurados en formato CSV o XLS en tesauros SKOS, siguiendo los principios de la web semántica. Esta herramienta llamada **easySKOS** ofrece diferentes plantillas para que los usuarios puedan rellenarlas con su información. Posteriormente, se habilita un *drag&drop* que transformará esa plantilla en un tesauro SKOS, obteniendo datos que pueden ser publicados en la web.

Con este proyecto se pretende ayudar a la expansión global de la web semántica y se actúa ofreciendo una herramienta fácil de usar y para todos los públicos que genera tesauros SKOS a partir de información semi estructurada y no ontológica.

Palabras clave Web semántica, Datos enlazados, Lenguajes de mapeos, RML, SKOS, Tesauros

I

Automated mapping process for creating SKOS thesauri

PART I. INTRODUCTION

Chapter 1

Introduction

This section outlines the meaning, inspiration and basic goals to be accomplished by this final degree project and the layout of the rest of the paper.

1.1 Context

During the last decades, we have witnessed a substantial evolution of information technology, now IT. Internet has been a fundamental part of this process since J.C.R. Licklider, a researcher at the Massachusetts Institute of Technology (MIT), coined the concept of *"Galactic Network"*[1] in a way similar to how the Internet is understood. The growing expansion and dominance of the Internet has led to an incredible number of more than 4.500 million users around the world, which represents around 60% of the entire world population[2]. To put the magnitude of this figure in context, 74% of the world's population have access to a safe drinking water source at home, or only the 54% has a minimum of safe managed sanitation services[3].

Internet is here to stay, offering a wide range of undiscovered opportunities and challenges to be faced by our generation and the future ones. Today, the world is inconceivable without the advantages that the Internet has brought to mankind.

As a result of this evolution, we have noticed how much data has been created using new IT. Only during 2018 were we responsible for the creation of more than 33 zetabytes (ZB) [1] (i.e., 1 ZB is equivalent to 10^{21} bytes). This massive creation of data (see figure 1.1) has opened up a world of new opportunities, but at the same time, it also creates structural shortcomings that limit its exploitation.

[1] *For more information: MIT and the galactic network*
[2] *For more information: World population*
[3] *For more information: UNICEF*

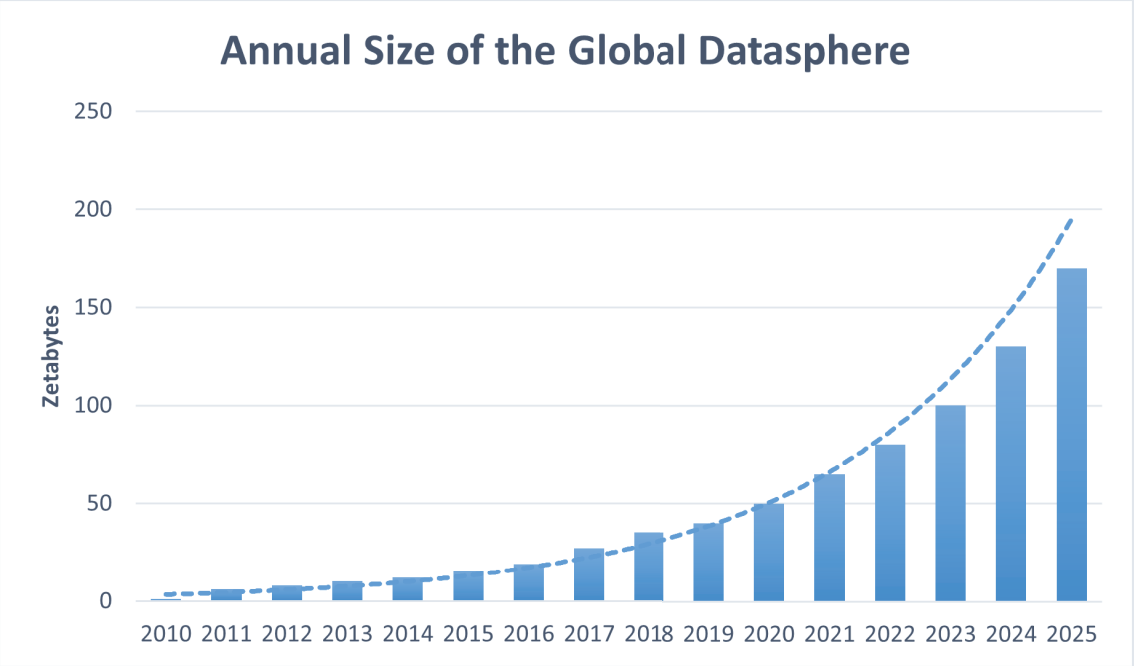

Figure 1.1: Evolution of the volume of data creation.

With the aim of facing some of the limitations that have appeared, new hardware (e.g., quantum computers[2] or cloud computing[3]) and software improvements(e.g., artificial intelligence[4] or data visualisation[5]) have been developed.

In the same direction, our attention will be on the Artificial Intelligence subdomain, which studies and develops new techniques related *to boost the processing capacity of data to obtain useful information*[6]. This subdomain is known as the Semantic Web, and some of the most important advantages in the field are:

- Creation of the concept of **linked data**, knowledge graphs[4].
- **Specific vocabularies** to enrich the gross data (not yet linked) with meaning.
- Creation of the concept **ontology** consisting of a collection of information that defines verbose relationships between concepts.
- New specifications for knowledge representation languages are used to express these data, such as the Resource Description Framework (RDF[8]) or the Ontology Web Language (OWL[9]), which are responsible for representing the data on the World Wide Web.

Throughout this document, we will discuss each of the fundamentals of the Semantic Web, and in particular, the SKOS vocabulary and how it enriches the data, because

[4] *"A knowledge graph (i) mainly describes real world entities and their interrelations, organised in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains."* [7]

it is adopted by many institutions around the world and is one of the standards proposed by the *World Wide Web Consortium* [5] for bringing many documents (e.g., thesauri) to the Web ecosystem and therefore increasing the contribution to the open and globally connected knowledge stored on the Web.

1.2 Objectives and motivation

In recent years, many excellent projects have been born (e.g., DBPedia[10], Gene Ontology Resource[11], etc.) with the aim of expanding the popularity and usage of the Semantic Web and its implications. Although they are formal and serious projects, their popularity among the general population has not reached the expected level[12]. In addition, several studies have argued about the limits of the expansion of the Semantic Web in the general public and institutions. Some of them [12] have identified several deficiencies, such as:

- The need to develop tools for the transformation of linked data and the lack of experts in different areas for these transformations.
- There is a lack of applications that consume Linked Data.

This project was born with the objective of reducing the negative impact of these shortcomings. In particular, its main objective is to reduce entry barriers related to the creation of linked data for non-semantic web experts. In this way, the community will be able to grow and more information will be shared. Therefore, the primary idea of Tim Berners Lee, which is that '*a well-designed and information-rich semantic web can helped to evolve human knowledge as a whole*' could be achieved[13].

How will it be done? Until now, traditional data transformations into ontological resources were performed by experts who have a wide experience in this field. First, because they did not know whether it was possible to make those transformations and if they would be able to transform the traditional data with guarantees. However, in chapter 4, we will propose the initial version of *easySKOS*, which is an open source tool that transforms traditional CSV or Excel spreadsheets into RDF-based documents, using the SKOS vocabulary.

SKOS is used because it provides a framework to express the basic structure and content of controlled vocabularies such as thesaurus or taxonomies and facilitates the development and publication of concepts on the Web, as well as the linking of data on the Web and the integration of concepts into other concept schemes[14].

[5] *For more information:* W3C

What need does this project meet? The idea of developing this converter stemmed from another project that consists of building a taxonomy for a group of biologists. We had to clean up the gross data and build the mapping rules for it. After that project, we started to think about an automated way to create that type of taxonomies, because we thought that this group of biologists with no experience in the field of semantic web were not the only ones with this need. As a consequence, we started working on the converter and adding more interesting SKOS templates.

In pursuit of this main goal, the following milestones were identified:

- Specification of requirements.
- Design the architecture of the software.
- Review the state-of-the-art of mapping languages to be able to define the best option, regarding the necessities of this project.
- Review existing vocabularies (LOV[6]) in order to choose one that is more versatile and useful in more different scenarios, as well as easy to use and lightweight to not hinder the transformation process.
- Design and create the CSV and Excel templates that will be utilised by users.
- Design and implement a web service with a friendly UI/UX (front-end development) that allows the user to upload files and download the SKOS-based thesauri.
- Design and implement the functionality of the web service (back-end development)
- Design and implement the HTTP REST API interfaces for communication between the front-end and back-end.
- Build a complete documentation that collects all the information on how to use it, troubleshooting problems, and the change log.
- Measurement and evaluation of the tool with real users.

Finally, the success of Linked Data[7] [8] in any domain will depend on the ability to generate and connect data. Moreover, everyone should be able to retrieve the most complete information from the common knowledge shared.

[6] *For more information: Linked Open Vocabularies*

[7] *For example, the fascinating initiative called DRUGS4COVID that collects and processes the corpus of more than 60,000 scientific articles indexes to enrich them with links to external resources.*

[8] For example, the World Health Organisation is using Semantic Web principles in ICD-11[15]

This is the direction this project is following, that is, to offer an easy method for experts and non-semantic web experts to generate rich and well-structured data, in line with linked data principles. These principles are [16]:

- Use URIs as identifiers for "things".
- Use HTTP URIs so that resources are available to the general public.
- Provide standardised information (RDF, SPARQL) when someone searches for a URI.

1.3 Methodology

The elements that have been used to develop this document and to achieve the objectives set out in this study are the following:

- This document has been done partly on the basis of the guidelines proposed by Petersen et al. [17] for the main stages of a systematic review of the literature in the State-of-the-Art chapter. The information has been searched in the most popular scientific databases, such as Google Scholar, Thomson Reuters Web of Knowledge, and EBSCO Business Source Premier. The sources with the highest number of citations and the most current publication dates were selected from all sources.
- The development section has been completed following an adaptation of the *waterfall* methodology, with special attention to the design and analysis phases.

1.4 Structure

The document is organised as follows: This first introduction has contextualised the problems we have to face and the motivation that guides our objectives. Second, we will present the state-of-the-art of mapping languages and SKOS terminology, which are the foundations on which the whole project will be built. Later, we will introduce **easySKOS** which is the development part of this project and consists of a web service that provides an easy way to transform CSV and Excel spreadsheets into SKOS-based files. Finally, we will share and analyse the results obtained and describe what future work should be done.

II

Automated mapping process for creating SKOS thesauri

PART II. STATE OF THE ART

Chapter 2

The Mapping Process to create RDF-based documents

Since the term Linked Data was first coined in 2006 by Tim Berners-Lee, where he discussed the Semantic Web project[1], we have been studying new methods to unleash its enormous potential [16]. Similarly, we have experienced an exponential increase in publications that follow the open data plan[2], such as The Linked Open Data (see appendix A) Cloud[3], DBpedia[4], the European Data Portal [5] or the Spanish Data Portal[6]. Those examples had to transform their original data (i.e., physical and digital but not RDF-based data) into data aligned with the linked data principles.

Nevertheless, to be able to perform those transformations, a mapping process needs to be done. In this project, we are going to give a brief description of the approaches developed and discuss their advantages and disadvantages. At the end of the chapter, we will compare the solutions and decide about what approach fits the needs of our project.

2.1 Mapping languages

Most of the data that has been created does not follow the principles of the semantic web and linked data. Besides, the nature of the data may be digital or physical. Then, digital data may have different structures (e.g., hierarchical, plain or tabular) or appears in different formats (e.g., XLSX, JSON or CSV). As a consequence, the

[1]For further details see: Principles of Linked Data
[2]Tim's 5-star Open Data plan
[3]The Linked Open Data Cloud
[4]For further details see: Dbpedia.com
[5]More information on: data.europe.eu
[6]More information on: https://datos.gob.es/es/

mapping languages are of particular importance in the mapping process where the input data is mapped and transformed into data that follows the RDF data model [18].

Because of this very diverse nature of the data, distinct mapping approaches have been developed, ranging from *custom implementations* [19], which were the first to appear on the scene, to more *generic approaches* [20].

Initially, generic approaches focused on data with different formats, and the most mature solutions were for the generation of knowledge graphs from sources of relational databases. There were divided into two main groups: (i) *direct mapping* and (ii) *detached rules* (R2RML) [21].

- *Direct mapping* defines simple transformations, providing the basis to be used later to define more complex transformations.
- *Detached rules* consist of separating the definition and the execution, for example D2RQ, that lead to the W3C recommended R2RML.

In *generic approaches*, data owners must learn several tools for each data format [22]. Therefore, the researchers came up with different solutions for heterogeneous data sources. Most of the solutions focused on *detached rules* because in *direct mapping* each type of data source requires its implementation (e.g., case of Datalift [23]) [24]. As a result, the solutions following the *detached rules* diverged into two streams

- *Dedicated mapping languages* such as RML [22] or xR2RML [25] that enhances both RML and R2RML.
- *Repurposed mapping languages* that broaden current languages for other assignments, for example, SPARQL-Generate [26] that repurposes SPARQL. [27]

RML, the first language that extended R2RML and currently, the main *dedicated mapping languages* will be analysed and compared with R2RML to identify their advantages and disadvantages.

2.1.1 RML and R2RML comparison

RDF Mapping Language (RML) is the RDF-based language designed to express customised mapping rules from heterogeneous data structures (e.g., tabular, such as databases, CSV or TSV files, hierarchical, such as XML or JSON data sources, even semistructured, such as HTML) and serialisations of the RDF data model. It is a generalisation from the R2RML mapping language but remains backwards compatible with R2RML [28].

Relational to RDF Mapping Language (R2RML) is the recommendation to express data-customised mappings in relational databases to generate knowledge graphs represented using the Resource Description Framework (RDF) [24].

The main difference between RML and R2RML lies in the type of data they can support. While RML works with *heterogeneous data*, R2RML works with a homogeneous one [24]. However, there are more differences in behaviour and syntax. The following figure 2.1 resumes some of them:

Language	R2RML	RML
prefix	rr	rml
Relational DBs	myriad tables one DB	myriad tables myriad DBs
Other data structures	-	CSV, TSV, XML, JSON, etc.
Data transformation	pre-processing	pre-processing inline processing

Table 2.1: R2RML vs RML. Source: [24]

2.1.2 YARRRML

YARRRML is a language that aims to make the creation of functional mappings as accessible as possible. It was designed to be human-friendly while maintaining all the functionalities of RML. It has achieved this by having a comfortable syntax that allows us to read and understand what we write [29].

The following figure 2.1 visually illustrates the differences in syntax.

Figure 2.1: YARRRML to RML conversion. Source: OEG-UPM. Methodological Guidelines for the Generation of Linked Data.

In addition, it allows shortcuts in the definition of subjects, predicates, and objects, as well as contracted forms to define subject maps and predicates-objects maps.

2.2 Engines to construct Knowledge Graphs

Once we have reviewed the mapping languages currently in use, the next step is to build our Knowledge Graph using one of the following engines. A distinction can be made between engines that support homogeneous data sources or heterogeneous ones, similar to the mapping languages that we have just seen before.

2.2.1 RMLmapper

RMLmapper is a Java-based library that executes *RML rules* to generate Linked Data. Its main advantages are the ease with which it can be used and the quality of its results[7].

It supports both structured data in databases and in XML, CSV, and JSON formats. In the future, they plan to add more formats, such as Web API data and No-SQL databases.

2.2.2 SDM-RDFizer

SDM-RDFizer is a Python-based library engine that uses *RDFlib* to transform heterogeneous data into RDF knowledge graphs. The SDM-RDFizer engine uses optimised data structures and relational algebra operators to allow efficient execution of RML triple maps even when there is a lot of data [30].

2.2.3 RocketRML

RocketRML is a NodeJS implementation of the RMLmapper engine. Although it has some limitations, it performs efficiently for small and medium data, even more efficiently than SDM-RDFizer. Moreover, it was built for specific use cases working with XML files, so its performance with this type of file is quite fast [31]. Some important limitations are as follows:

- No support for JOINs.
- No support for named graphs.
- Only JSON and XML formats are supported on a logical source.
- Only JavaScript function implementations are supported.

[7]More details at: RMLmapper

As we can see, there are many interesting solutions, and each of them has its particularities because they were born to meet specific needs, although in their core, the main goal remains the same to convert non-ontological resources into RDF-based documents.

2.3 Cutting-edge tools related with Linked Data

During this section, we will explore two different tools that have very interesting features that allow us to outperform the creation and maintenance of linked data.

2.3.1 RMLEditor

RMLEditor proposes a solution based on a graph-based interface that facilitates the design and management of mappings to achieve data that meet the requirements of linked data. Under its skeleton, the *RMLmapper* engine runs to execute and visualise the mappings that are loaded into RMLEditor, and its main features are that it divides knowledge from execution, making it feasible to obtain a correct result without knowing anything about the syntax of the mapping language, and it also supports data sources in different formats [32].

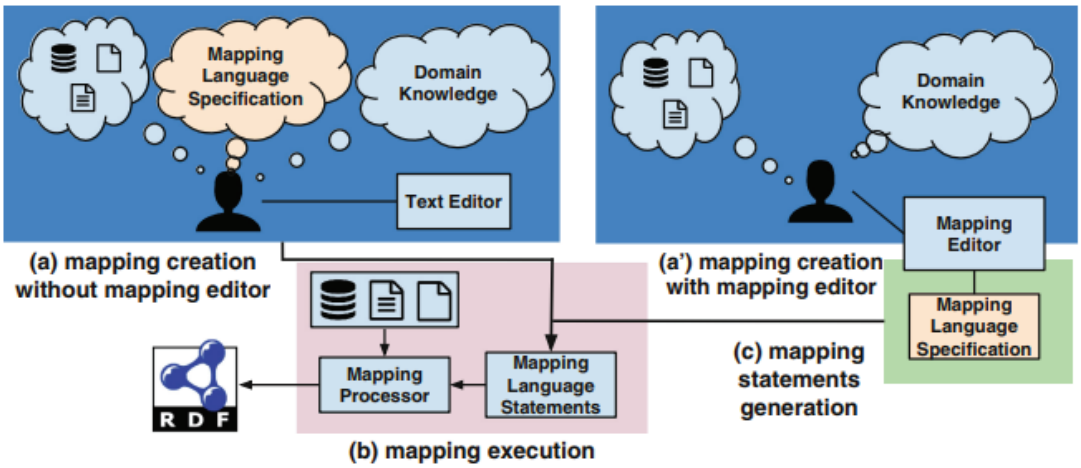

Figure 2.2: Importance of the mapping editor in the mapping process. Source: [32]

Mapping with **RMLEditor** allows non-Semantic Web experts to define mappings without knowing the language specification used (see figure 2.2).

2.3.2 Mapeathor

Mapeathor is a user-friendly mapping language-independent tool that uses spreadsheets to declare transformation rules and translate them into different mapping languages to simplify the creation process [33].

(a) Prefix sheet

Prefix	URI
noise	http://v.ciudadesabiertas.es/cont-acustica#
noise-res	http://v.ciudadesabiertas.es/res/cont-acustica#
sosa	http://www.w3.org/ns/sosa/

(b) Subject sheet

ID	Class	URI
Station	noise:EstacionMedida	noise-res:estacion-medida/{id}
Observation	noise:Observacion	noise-res:observacion/{idx}

(c) Source sheet

ID	Feature	Value
Station	query	SELECT id, name FROM Station
Observation	source	data/station.json
Observation	format	JSON
Observation	iterator	$

(e) Function sheet

FunctionID	Feature	Value
<Fun1>	fno:executes	grel:replace
<Fun1>	ex:param1	{obsProperty}
<Fun1>	ex:param2	" "
<Fun1>	ex:param3	"-"

(d) Predicate_Object sheet

ID	Predicate	Object	DataType	ReferenceID	InnerRef	OuterRef
Station	dcterms:identifier	{id}	string			
Station	schema:name	{name}	string			
Station	geosparql:hasGeometry	noise-res:punto/{id}	iri			
Observation	sosa:resultTime	{resTime}	Time			
Observation	sosa:madeBySensor			Station	{madeBySensor}	{id}
Observation	sosa:observedProperty	<Fun1>				

Figure 2.3: Mapeathor's mapping template.

The design (see figure 2.3) aims to be language independent and simplify the writing process so that the user does not need to learn a mapping language.

Chapter 3

Simple Knowledge Organisation System

SKOS is a data sharing standard that aims to be a common denominator for various modelling methods, offering a basis for representing and using vocabularies and ontologies within the Semantic Web system. Being RDF-based, SKOS structures the data in the form of RDF-triplets [14].

These modelling methods have evolved throughout history, including classification schemes, subject heading systems, taxonomies, or thesauri. An important point for SKOS is that, although they have particular characteristics, each of these approaches shares many similarities and can often be treated similarly [34].

After years of work and development, there are a multitude of knowledge organisation systems (KOSs), such as The International Press Telecommunications Council (IPTC)[1] thesauri which were designed to provide a universal news labelling system and nowadays more and more thesauri are being built with SKOS, because it provides a low-cost migration path to port existing knowledge organisation systems to RDF [14].

In the following section, we will review the SKOS specification, the core concepts, and its usability. Finally, we will take a look at the different tools that are already available on the Internet.

3.1 Specification

The SKOS namespace URI is:

[1]For further information, see: http://cv.iptc.org/newscodes

http://www.w3.org/2004/02/skos/core#

In the following paragraphs, we describe the entire SKOS vocabulary.

3.1.1 Concept and Lexical Labels

A Concept *(skos:concept)* is the basic component of the SKOS vocabulary. We can define it as a unit of thought, ideas, meanings, or (categories of) objects and events that support KOSs [14]. In a way, a possible analogy would be that classes that describe objects in object-orientated programming languages.

In addition, SKOS has three main properties to add labels to that concept:

- **skos:prefLabel** is used to describe this concept in a KOS and its implementation in a clear and unambiguous way. There can be no more than one *skos:prefLabel* per language tag associated with a concept.
- **skos:altLabel** is used to list synonyms, acronyms, or abbreviations available for a concept when necessary.
- **skos:hiddenLabel** is commonly used when misspelled variants of other lexical labels must be included.

It is important to highlight that these three properties are all subproperties of **rdfs:label** and are used to link a *Concept* to an RDF literal, which means a string that is usually combined with a language tag, e.g., 'en.'

3.1.2 Concept Schemes

A concept scheme *skos:ConceptScheme* is a grouping of concepts. The association of a concept with a scheme is made using the property*skos:inScheme*, while *skos:has TopConcept* is used to indicate the existence of a root concept. The inverse *owl:inverseOf* of this property is *skos:topConceptOf*.

3.1.3 Concept Collections

SKOS concept collections are labelled and/or ordered groups of SKOS concepts that share something in common or where some *skos:Concept* can be placed in a meaningful order. Its main use is to enrich the Concept Scheme without establishing explicit schematic relationships that distort the nature of the Concept Scheme.

The properties we use to manage the collections are the following:

- *skos:Collection* declares a collection.

- *skos:OrderedCollection* declares an Ordered collection. It is a subclass of *skos:Collection*.
- *skos:member* assigns a Concept to a Collection.
- *skos:memberList* assigns a Concept to a Ordered collection.

3.1.4 Documentation Notes

SKOS is a sufficiently efficient vocabulary in terms of providing additional information on Concepts, as it has different mechanisms to achieve this purpose. Moreover, there is no restriction on the nature of this information, e.g., it could be plain text, hypertext, or an image; it could be a definition, information about the scope of a concept, editorial information, or any other type of information. The inclusion of collections within other collections is also allowed.

In SKOS, there are seven properties for linking notes to concepts:

- *skos:note* is the general property that groups all the others.
- *skos:scopeNote* provides information on meaning in a given domain.
- *skos:historyNote* describes the relevant changes in the meaning of a concept.
- *skos:changeNote* allows the documentation of changes made to a concept in the management and maintenance processes of a Concept Scheme.
- *skos:definition* provides a complete definition of the meaning of a concept.
- *skos:editorialNote* provides administrative editing and publishing information.
- *skos:example* gives examples of use.

3.1.5 Notations

SKOS allows a notation *(skos:notation)* to be associated with a concept, that is, it allows a concept to be associated with its corresponding entry at the level of the organisational system used. In this way, a concept is associated with the specific domain of a conceptual scheme.

A notation *(skos:notation)* is different from a lexical label in that a notation is not normally recognisable as a word or a sequence of words in any natural language.

3.1.6 Semantic Relationships

SKOS semantic relations are connections between SKOS concepts that are inherent in the meanings of the linked concepts.

At this point, SKOS considers two types of semantic relationship. On the one hand, hierarchical relations are those that indicate when one concept is more general/specific (i.e. *skos:broader/skos:narrower*)than another. Basic hierarchical relationships are defined without transitive properties. However, to be able to make inferences, SKOS abstracts one more level, creating higher classes to allow the creation of hierarchical transitive relationships.

On the other hand, the associative relationship *skos:related* indicates that the two are inherently "related", but neither is more general than the other.

The following figure 3.1 shows the SKOS organisation of classes, where we can see how all properties are derived from the main property *skos:semanticRelation*

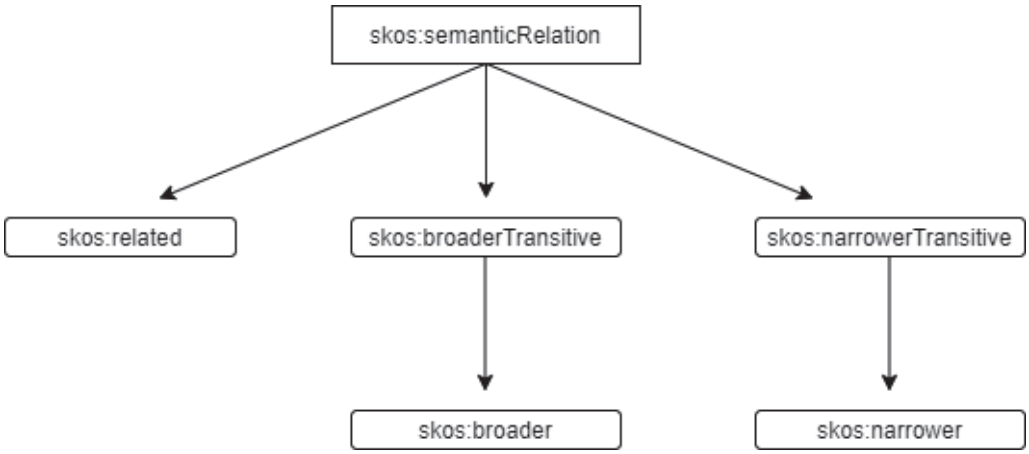

Figure 3.1: Flow chart of semantic relationships in SKOS.

3.1.7 Mapping Properties

SKOS presents a set of properties for making connections between Concepts from different Concept Schemes. They are subclasses of their corresponding semantic relations, as can be seen in the figure below 3.2:

However, the mapping properties do not have the same function as a semantic relation. They indicate the extent to which two concepts belonging to different Concept Schemes correspond to each other.

The *skos:closeMatch* property is used to link two concepts that are substantially similar so that they can be used interchangeably in some information retrieval applications. Also, it is not declared as a transitive property.

The *skos:exactMatch* property is defined as a subproperty of *skos:closeMatch* and indicates a complete and precise correspondence between concepts, which means that concepts can be used interchangeably in a wide range of information retrieval

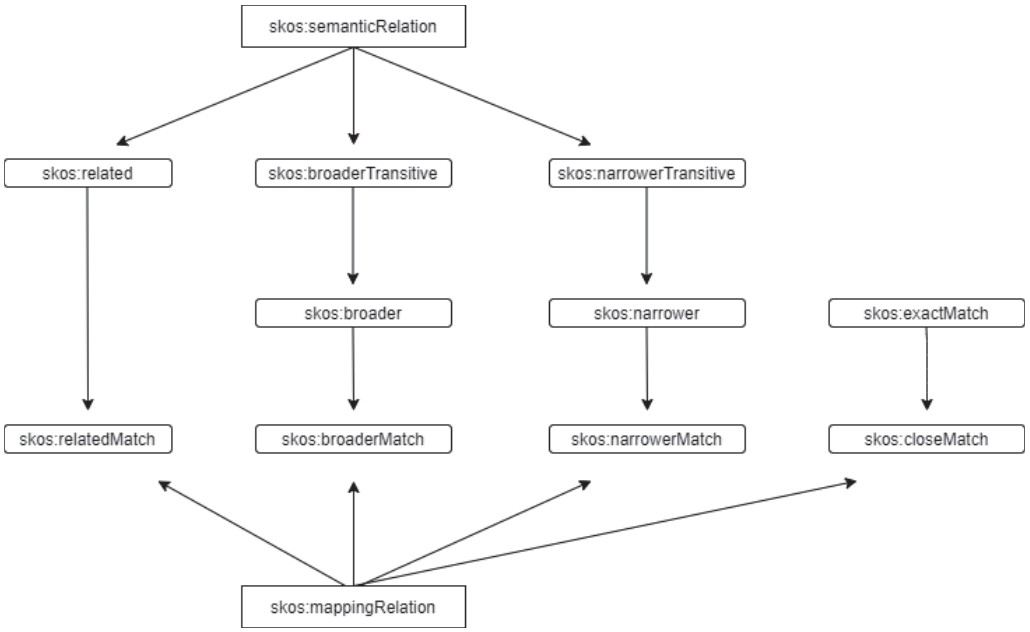

Figure 3.2: Organisational diagram of semantic and mapping relationships in SKOS.

applications. In addition, it is defined as a transitive property. The figure below 3.3 shows the difference between a transitive and a non-transitive property.

Property type	Example
Transitive	A skos:exactMatch B
	B skos:exactMatch C
	entails: A skos:exactMatch C
Non-Transitive	A skos:broadMatch B
	B skos:broadMatch C
	does not entails: A skos:broadMatch C

Figure 3.3: Transitivity in SKOS properties.

Using the SKOS properties that have been exposed, we can define, for example, a crime as *skos:Concept* with the following attributes:

```
<http://terminoteca.linkeddata.es/scheme/delitos/concept/A001000000>
  a skos:Concept ;
  skos:inScheme <http://terminoteca.linkeddata.es/scheme/delitos/> ;
  skos:prefLabel "Organisational crime"@en ;
  skos:altLabel "Corporate crime"@en ;
  skos:narrower <http://terminoteca.linkeddata.es/scheme/delitos/concept/A001001000
      >, <http://terminoteca.linkeddata.es/scheme/delitos/concept/A001002000>, <
      http://terminoteca.linkeddata.es/scheme/delitos/concept/A001003000>, <http://
      terminoteca.linkeddata.es/scheme/delitos/concept/A001004000>, <http://
      terminoteca.linkeddata.es/scheme/delitos/concept/A001005000>, <http://
      terminoteca.linkeddata.es/scheme/delitos/concept/A001006000>, <http://
      terminoteca.linkeddata.es/scheme/delitos/concept/A001007000> ;
  skos:note "Refers to acts in violation of the law that are committed by
      businesses, corporations, or individuals within those entities. Corporate
      crime is also closely associated with white collar crime, organised crime,
      and state-corporate crime."@en ;
  skos:definition "Conduct of a corporation or of employees acting on behalf of a
      corporation, which is proscribed and punishable by law."@en .
```

3.2 SKOS thesauri

Since Brownson first used the term thesaurus in 1957 [35], indicating it as a solution to the problem of translating concepts and their relationships expressed in documents into a more precise and unambiguous language to facilitate the retrieval of information, new thesaurus have been published using the SKOS paradigm (e.g., UNESCO). Thesaurus [2], IPTC Newscode [3], etc.).

Similarly to Brownson's definition, a thesaurus could be thought of as controlled lists to refer to concepts between which heuristic or intuitive relationships are established. Additional explanatory information on a term, such as a description (or scope note), bibliographic citations, etc., might be included in it [14].

Structural features include the following:

- It is a controlled vocabulary [4].
- Concepts can be root or leaf, that is, leaf concepts always have to be related to a single root.
- Relationships between concepts form hierarchical (broader/narrower) and associative structures.

Similarly, from the point of view of functionality, we can highlight that Thesauri have been used to solve problems of ambiguity in natural language and document retrieval problems, being a connector between them. In addition, the concept of thesaurus has evolved as more research has been done on it, from a linguistic tool to a tool for organising and analysing information.

However, SKOS has not been the only alternative to thesaurus representation on the Web using the Resource Description Framework (RDF). Several projects have been released with the purpose of representing thesauri and conceptual schemes in XML format to be used in organisations and particular contexts (i.e., Zthes[5] (a specification for the representation of thesauri), MeSH[6] (Medical Subject Headings) or Topic Maps). Other RDF vocabularies such as LIMBER (Language Independent Metadata Browsing of European Resources), CERES (California Environmental Re-

[2]More details at: UNESCO Thesaurus
[3]More details at: IPTC Newscode
[4]More details about controlled vocabularies at: What are Controlled Vocabularies?
[5]More details at: Zthes family of specifications
[6]More details at: Medical Subject Headings

sources Evaluation System), or The Food and Agriculture Organisation AGROVOC thesaurus have also been released.

Each of these alternatives offers particular characteristics, but if we compare them with SKOS 3.4, we can clearly see who wins.

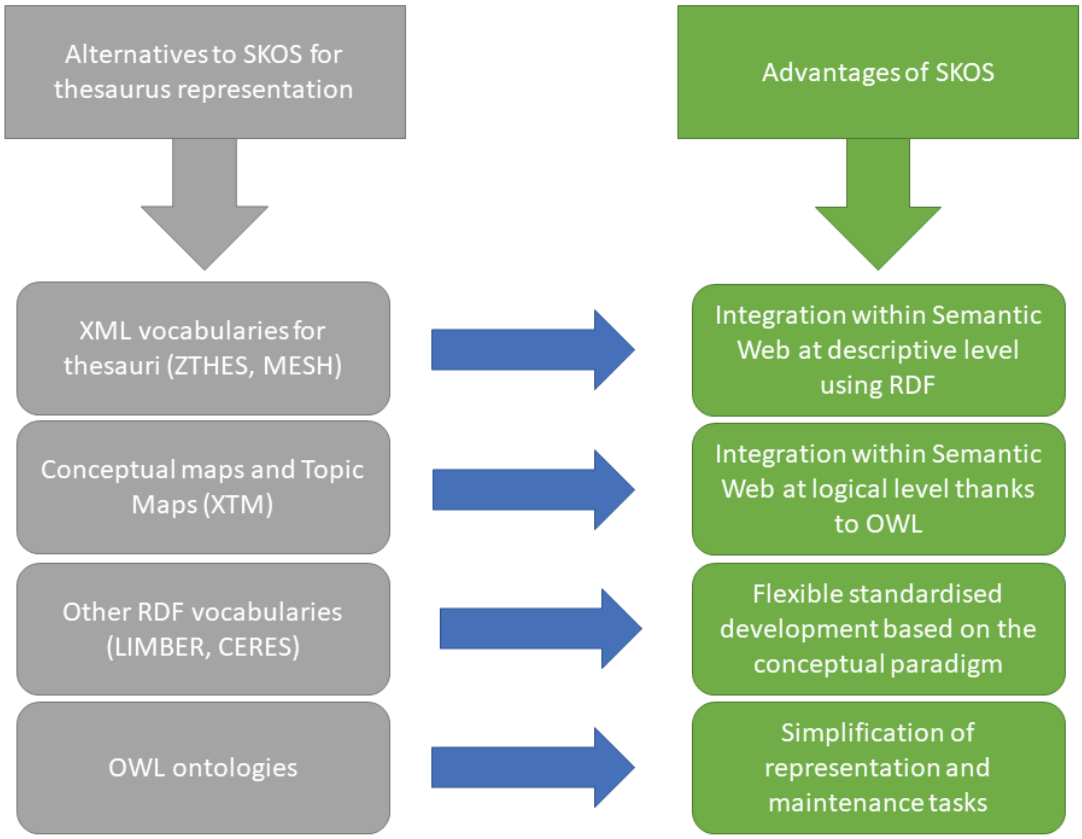

Figure 3.4: Advantages of SKOS over other alternatives. Source: [36]

In general, one of the key benefits of SKOS thesauri is that, thanks to their syntax and semantics, they can be expanded, enriched, and validated by the community and exchanged as linked open data because they follow the recommendations and meet the requirements of the Semantic Web. The core of a knowledge graph can be made up of a thesaurus that can be in continuous development, supporting, in the same way, processes of semantic search, reasoning, or data analysis, among others.

3.3 SKOS-related tools

First, we show the publishing and maintenance tools related to SKOS, as they can provide search, browsing, and other specific features.

Second, we would like to underline the importance of validation in any creation process and exclusively in this case. Therefore, we mention the tools that help us perform this validation automatically, *Skosify* and *qSKOS*.

3.3.1 Publishing and maintenance tools

3.3.1.1 PoolParty

PoolParty[7] is a platform to develop and maintain a multilingual SKOS thesaurus (Simple Information Organisation System) that aims to reduce the barrier for people without prior knowledge of the Semantic Web or special technical skills. It was conceived to facilitate various commercial applications for thesauri. However, its main disadvantage is that it is not open-source software [37].

PoolParty not only publishes thesaurus as Linked Open Data (LOD), but it also consumes it to enrich thesaurus with data from LOD sources (i.e., the concepts of thesaurus can be linked to, e.g., DBpedia, thereby creating an *owl:sameAs* relation between the concept in PoolParty and the one in DBpedia).

Moreover, PoolParty provides a functional tool that allows us to automatically perform quality checks on controlled vocabularies. It is able to check on *20 quality issues* [37].

3.3.1.2 Skosmos

Skosmos[8] is a controlled vocabulary publishing tool built on current Web standards, covers a wide variety of vocabularies, and is straightforward to use for both users without prior experience and developers. It has a multilingual user interface that allows users to browse and query the data, as well as visualise concept hierarchies. In addition to supporting the SKOS core, SKOSMOS also includes several Dublin Core and RDF properties [38] . In Skosmos, all instances of *skos:Concept* are displayed as concepts. Moreover, it supports all SKOS mapping properties and shows them on the Concept page.

3.3.1.3 HIVE

Helping Interdisciplinary Vocabulary Engineering (HIVE)[9] is a linked data, an automatic indexing application that is the main focus of the tool and is being implemented using KEA++[10] and Maui[11] algorithms.

[7] More details at: PoolParty.biz
[8] More details at: SKOSMOS
[9] More details at: HIVE
[10] KEA
[11] Maui

3.3.1.4 iQvoc

iQvoc[12] is an open source SKOS and mainly SKOS-XL vocabulary management tool that supports editing and publishing, among other features, developed by the Federal Environment Agency of Germany [39].

Some of its main features are as follows.

- Open-source availability.
- SKOS-XL compliance of the model.
- Web interface for browsing and querying.
- Multilingualism.
- Comfortably editing features with validation.
- Editorial team and workflow support.
- Linked Data support.

3.3.2 SKOS validation tools

The SKOS reference document [40] contains several integrity requirements. For a data set to be considered legitimate, the following integrity conditions must be met.

- *skos:ConceptScheme* is disjoint with *skos:Concept*, which means that in the animals vocab, Mammals is a **skos:ConceptScheme** and Horse is a **skos:Concept**. This means that mammals cannot be a *skos:Concept*, and Horse must not be a *skos:ConceptScheme*.
- *skos:prefLabel*, *skos:altLabel* and *skos:hiddenLabel* are pairwise disjoint properties, which means that no SKOS concept may be a member of more than one preferred, alternative, and hidden label.
- A resource does not have more than one *skos:prefLabel* value per language label.
- *skos:related* is disjoint with the *skos:broaderTransitive* property
- *skos:Collection* is disjoint with each of *skos:Concept* and *skos:ConceptScheme*.
- *skos:exactMatch* is disjoint with each of the properties *skos:broadMatch* and *skos:relatedMatch*.

[12] More details at: iQvoc

These conditions can be thought of as a minimum list of validation and/or consistency standards for SKOS vocabulary; moreover, many best practises related to vocabulary go beyond the SKOS integrity conditions, such as Skosify or qSKOS.

3.3.2.1 Skosify

Skosify was designed in the FinONTO project[13] with the objective of automatically validating SKOS vocabularies and, in addition, correcting some of the problems they present to increase the quality and validity of the vocabularies. It has been released as open source under the MIT Licence [41].

3.3.2.2 qSKOS

qSKOS[14] is a tool to find quality issues in SKOS-based documents. It works as a validation tool that can be used to measure vocabulary against a more comprehensive set of quality criteria.

[13]More details at: National Semantic Web Ontology Project in Finland (FinnONTO), 2003-2012
[14]More details at: qSKOS validation tool

III

Automated mapping process for creating SKOS thesauri

PART III. DEVELOPMENT

Chapter 4

Development of the easySKOS prototype

This section provides an overview of **easySKOS**, which represents the practical aspect of this final degree project and considers all information collected in the previous sections. It is a web service deployed on a virtual machine provided by the Ontology Engineering Group and can be accessed via the following URL:

https://terminoteca.linkeddata.es/easySKOS.html

The source code is stored in the following Github repositories:

For the back-end: https://github.com/mdelucasg/terminotecalib

For the front-end: https://github.com/mdelucasg/terminoteca

The remainder of this chapter is structured as follows. First, the objectives that guide the subsequent development of easySKOS are introduced, as well as the requirements are listed and explained. Then, the design and architecture are presented, describing its functionality and usage.

Finally, the evaluation of the web service will be discussed, providing some general conclusions and giving action lines that may be followed in the near future.

4.1 Objectives and requirements

To achieve a fast, versatile, and easy-to-use tool that experts or non-semantic Web experts could use, we identified a list of goals to be accomplished.

- **Effectiveness:** Creates a web service that satisfies the user's desire.

- **Efficiency:** Create a high-efficiency web service by reducing the steps necessary to complete the process.
- **Simple:** Provide a simple usage for experts and non-semantic Web experts.
- **Versatile:** Allow CSV and Excel spreadsheets to be input for the transformation.

Then, after having the list of objectives clear, we were able to define a list of development requirements, based mainly on the best practise of programming[1]. In the following, **architectural requirements** are identified:

- *easySKOS* is implemented using the Java programming language and Maven for project management.
- Run the **RMLmapper** engine to perform the transformation.
- The coupling between classes should be minimal.

The **functional requirements** are:

- Provide Drag&Drop function in the front-end to perform the transformation.
- Provide the most relevant templates for the most famous SKOS in CSV and XLSX serialisation.
- Use turtle[2] serialisation for the result.
- Provide metadata from the output, such as the number of concepts and triplets created.

The **non functional requirements** are:

- The source code should be formatted according to Prettier[3] guidelines.
- Run parameter input checks for each function to avoid errors and improper use.
- Comments for global functions and variables do not comment on what the source code can say.
- Use descriptive names for variables, objects, and functions.
- Make the functions short and focus on a single task.
- Functions must have at most four parameters.

[1]More information at: Coding best practises

[2]Turtle is a super set of N-Triple and has the same syntax as N-Triples serialisation. Turtle represents a RDF model as a list of statements, with the subject, predicate and object structure. [42]

[3]For further details see: prettier.io

- Provide complete documentation of the tool with real examples of usage.
- Provide different ways of contact to receive feedback and suggestions.

4.2 Design and architecture

easySKOS's high-level architecture is based on the *multilayered architecture* pattern [43]. This allows the presentation and transformation logic to be separated into the presentation layer and the application layer, respectively.

For the presentation layer, only HTML and CSS are used to build the front-end, in cooperation with some JavaScript for the drag&drop management. For the application layer, the Java programming language is used due to its robustness[4]. Here, **easySKOS** uses the power of the *RMLmapper* engine to perform the transformations according to the established mapping rules for each template.

Communication between the presentation and the application layer is carried out according to the pattern *Model-View-Controller* [44], in which the controller receives the template from the drag&drop and sends it to the application layer via the POST method. Then, the model performs the proper transformation to the template and sends it straightforwardly to the user.

To illustrate the architecture of the web service, the figure 4.1 presents a basic version of the class diagram. The View and Controller are part of the front-end (terminoteca project), whereas the Model is the backbone of the back-end.

[4]Write once, run anywhere

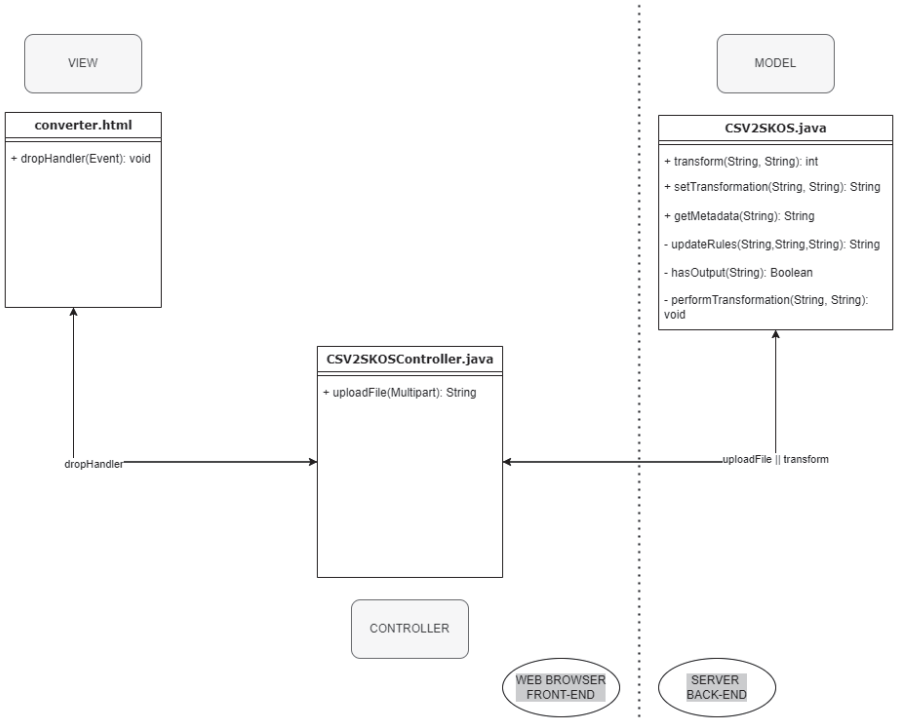

Figure 4.1: Model-View-Controller pattern in easySKOS' design.

4.3 Implementation of easySKOS

The implementation of the practical part of this degree project was carried out gradually, according to the waterfall methodology. First, we began with the front-end part, although both front and back-end were done in parallel. The final version is deployed on a virtual machine that the OEG Group has left us and is running at the following url:

https://terminoteca.linkeddata.es/easySKOS.html

Throughout the implementation process of **easySKOS**, several problems have arisen and many of them have been resolved. The following list contains the particularities of this project:

- It requires an Elasticsearch node to save all data, because it is a complementary part of Terminoteca[5].
- It is stateless, so if there were several servers to control user requests, it would not be necessary for one to always respond to them. However, in this initial version, it uses only one server for all requests. This stateless behaviour was chosen because the web service does not need data from the previous session to answer the next request. Each request is handled atomically. Furthermore,

[5]terminoteca.linkeddata.es

it can be redeployed when an accident occurs and scaled to add more servers if user demand increases.

- The drag&drop handler only accepts files with the same name as the provided templates, because we set the transformation without looking at the user data, just the file name.
- Currently, it only supports CSV and XLSX templates.
- The efficiency of the tool is affected when the number of rows is greater than 2.500.
- It does not work with special characters, such as adding quotes in a cell of the template.

In the following sections, we will thoroughly analyse each part of the project. First, we present the front-end structure. The back-end structure will follow, and finally we show the whole project and how it works.

4.3.1 Front-end structure

A goal that has always been present is to facilitate access to linked data transformation tools for experts and non-semantic web experts. To achieve this, the front-end part becomes a crucial part because it is what the user sees and uses. Consequently, we have made efforts to build a simple interface with all relevant information always available. It has been built using HTML, CSS and the table to download the templates is adapted from the OEG group[6].

[6]For further information see: OEG UPM home page

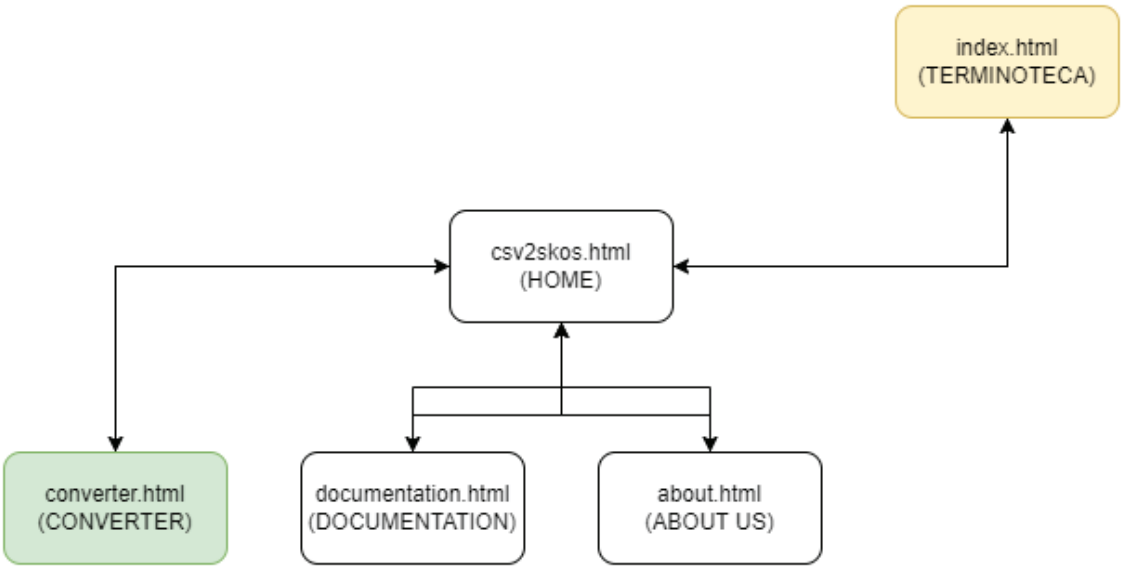

Figure 4.2: Front-end structure from the easySKOS project.

The final result can be seen in the following figures. We show the Hope (see Figure 4.3), Converter (see Figure 4.4), Documentation (see Figure 4.5) and About us (see Figure 4.6) screens.

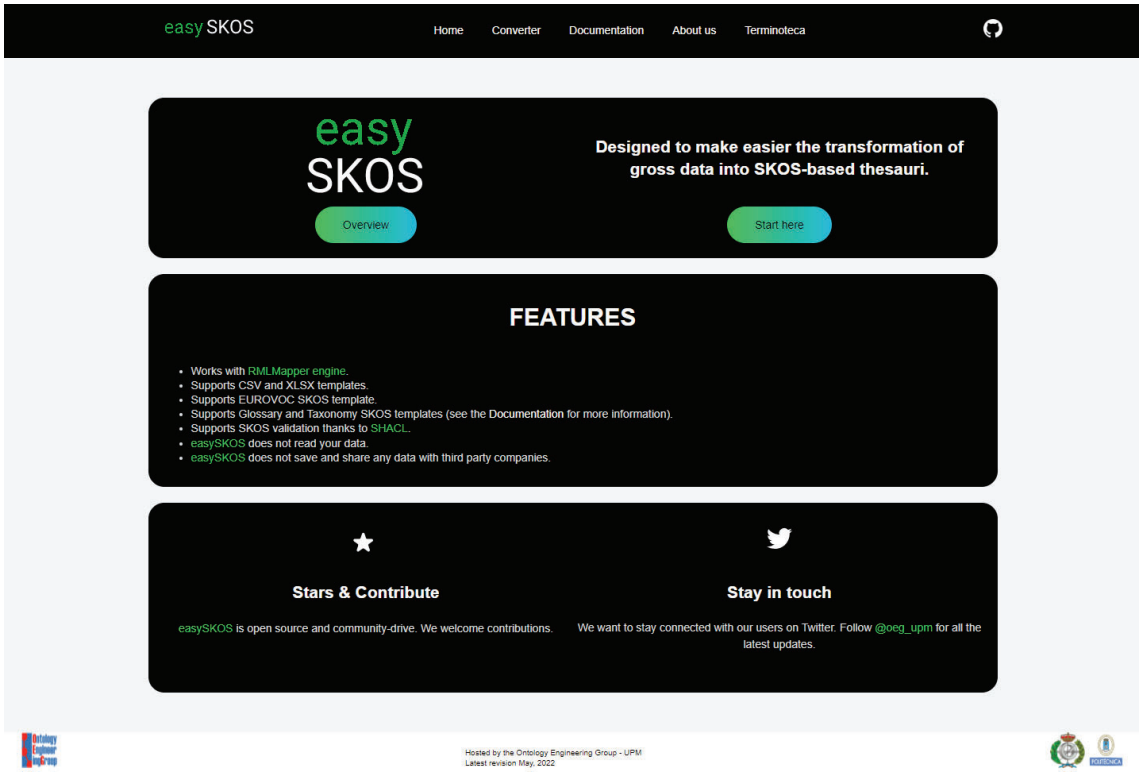

Figure 4.3: Home screen.

We follow a black and white palette with some green to highlight one of the purposes of the project, that is, to make the gross data transformation into SKOS-based thesauri easy. We also provide the *Overview* button that redirects to the documentation

screen and the *Start here* button that redirects to the converter screen. Moreover, we also show what features *easySKOS* have and how to contribute and contact us.

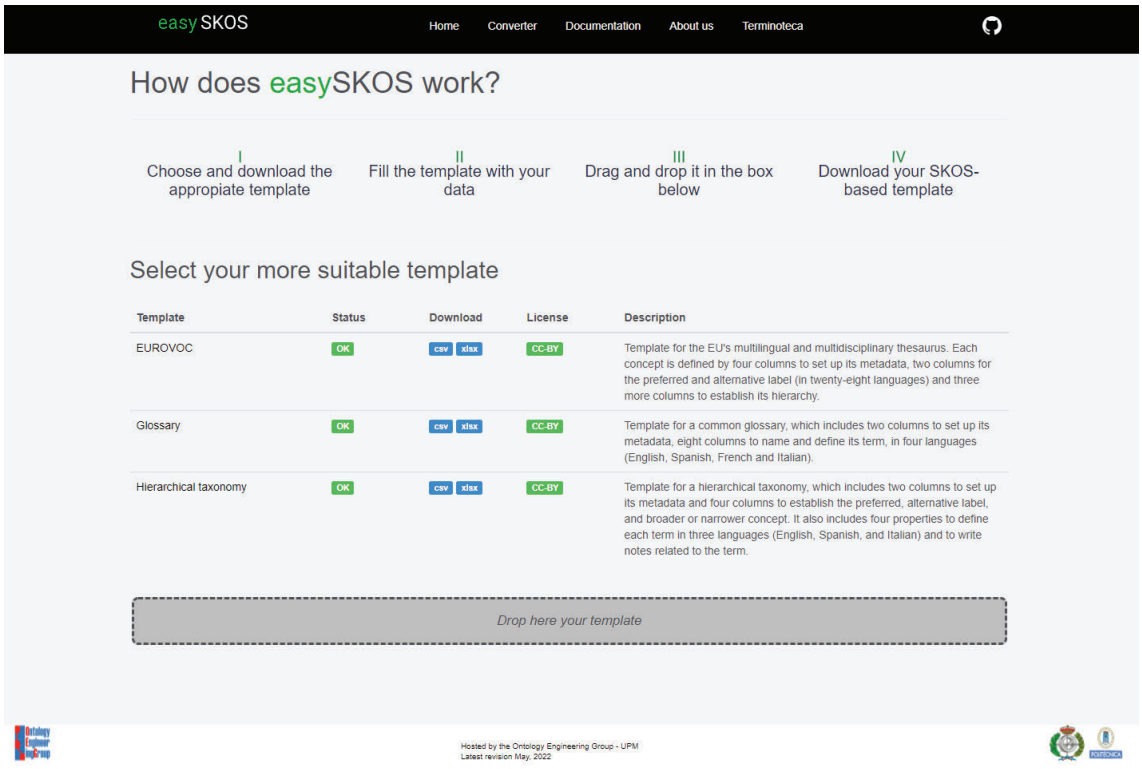

Figure 4.4: Converter screen.

This screen might be the most important because it is where the interaction with the user begins. To clarify how it works, we show the steps to be performed. In addition, in the table below, all relevant information related to the templates is provided. At the bottom, there is the box to *drag&drop* the template, once the user has already filled it. As a result, if everything has gone well, the download link may appear below the box. In other cases, an error will appear. The user should check the documentation to understand why the transformation process failed.

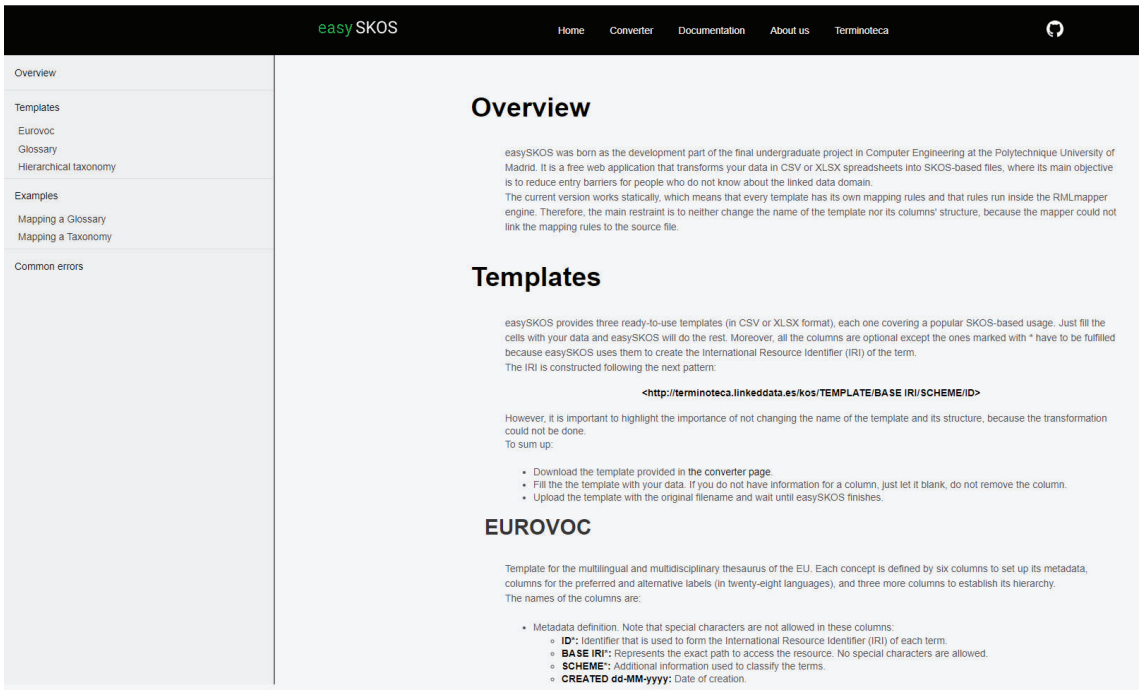

Figure 4.5: Documentation screen.

The documentation collects the structure of the templates provided with the definition of each column, examples of real-use of how to use the converter, and finally some common errors that may appear when using it incorrectly.

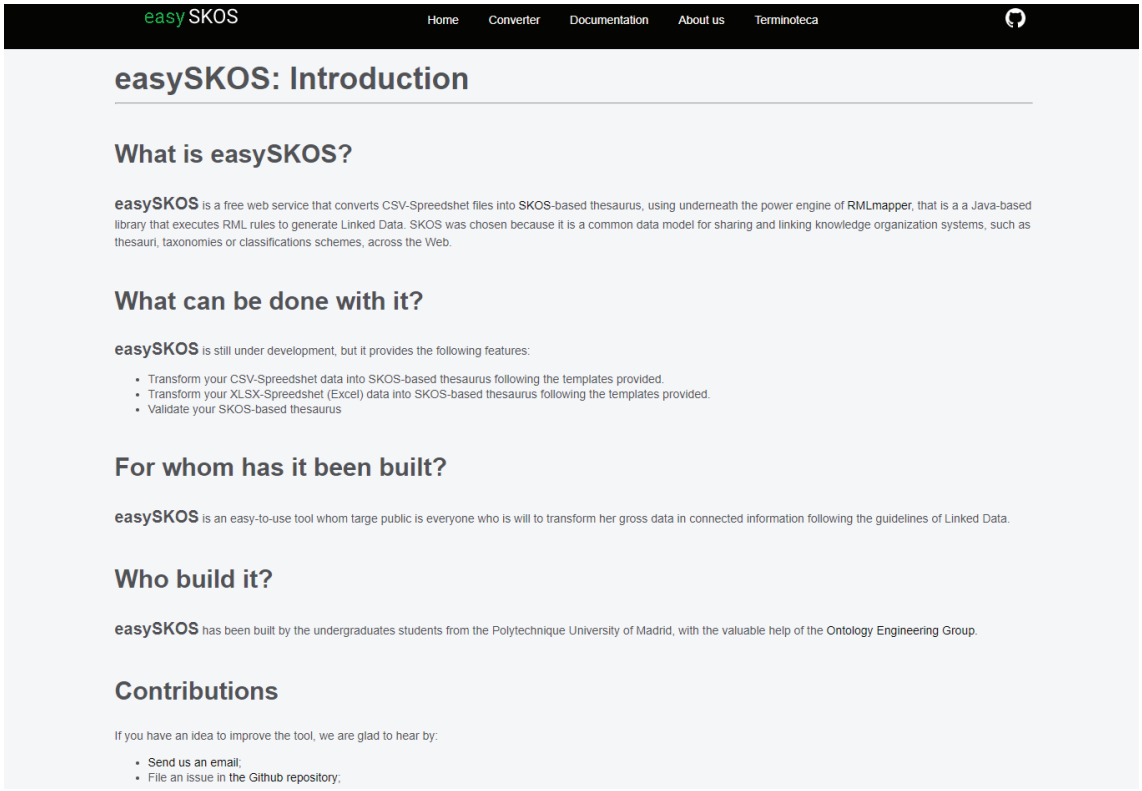

Figure 4.6: About us screen.

Here, we share more information about easySKOS and us and some ways to stay in contact.

4.3.2 Back-end structure

The back-end structure is divided into two main components that work together to respond to user requests.

These two components are the the RMLmapper which is the chosen engine to perform the mapping rules and the ***oeg.terminotecalib.transformacion*** package that receives the input source file to be transformed. Here, there are four private and three public functions (see figure 4.1) that are called in the Controller java class and are in charge of applying the rules, depending on the template received; call the RMLmapper engine to perform the transformation; get the metadata of the output file; and, send the answer back to the user.

4.3.3 How does easySKOS work?

The whole process is explained with the definition of three components which are the templates that easySKOS provides to the user, the RMLmapper engine that performs the transformation, and finally, the output SKOS-based file that is generated. In the following sections, we will describe each component.

4.3.3.1 Templates

easySKOS provides three ready-to-use templates (in CSV or XLSX format), each covering popular SKOS-based usage. Those templates are: EUROVOC, Glossary with definitions and hierarchical taxonomy. Below, we will explain the meaning of each column. Moreover, all columns are optional except those marked with ***** have to be fulfilled because **easySKOS** uses them to create the International Resource Identifier[7] (IRI) of the term.

The IRI is constructed according to the following pattern:

*<http://terminoteca.linkeddata.es/kos/**TEMPLATE/BASE IRI/SCHEME/ID**>*

where TEMPLATE refers to eurovoc, hierarchicalTaxonomy, or glossary, and BASE IRI and SCHEME are retrieved from the user request.

[7]For further information see: IRI documentation

In addition, the mapping rules implemented for each template were written following the human-friendly sintaxis of YARRRML. Then, using the YARRRML Matey parser[8], we converted them to RML serialisation.

EUROVOC

Template for the multilingual and multidisciplinary thesaurus of the EU. Each concept is defined by six columns to set up its metadata, columns for the preferred and alternative labels (in twenty-seven languages), and three more columns to establish its hierarchy.

The metadata is defined with the following columns. Note that special characters are not allowed in these columns:

- **ID*:** Identifier that is used to form the International Resource Identifier (IRI) for each term.
- **BASE IRI*:** Represents the exact path to access the resource.
- **SCHEME*:** Additional information used to classify the terms.
- **CREATED dd-MM-yyyy:** Date of creation.
- **VERSION:** Actual version of the term.
- **STATUS:** Actual status of the term.

In the columns below, the quotes can cause problems if they are not closed. Then, the labels are defined with:

- **PREFERRED LABEL:** Describe the concept in a clear and unambiguous way. There are 27 columns, each one for each European language.
- **ALTERNATIVE LABEL:** List synonyms, acronyms, or abbreviations available for a concept when necessary. There are 27 columns, each one for each European language.

Finally, the hierarchical relationships are defined with the next columns:

- **BROADER CONCEPT:** Represents, in English, that this concept has a broader concept, which means that this concept is more specific. NOTE: a SKOS concept can be attached to several broader concepts at the same time.
- **NARROWER CONCEPT:** Represents, in English, that this concept has a narrower concept, which means that this concept is more generic.
- **RELATED:** Used to assert an associative link between two SKOS concepts.

[8]YARRRML parser

Glossary

Template for a common glossary, which includes three columns to set up its metadata, eight columns to name and define its term, in four languages (English, Spanish, French, and Italian).

The metadata is defined with the following columns. Note that special characters are not allowed in these columns:

- **ID*:** Identifier that is used to form the International Resource Identifier (IRI) for each term.
- **BASE IRI*:** Represents the exact path to access the resource.
- **SCHEME*:** Additional information used to classify the terms.

Then, the concept is defined with the following columns. Note that the quotes can cause problems if they are not closed.

- **ES:** Term in Spanish.
- **EN:** Term in English.
- **FR:** Term in French.
- **IT:** Term in Italian.
- **DEFINICION:** Definition in Spanish.
- **DEFINITION:** Definition in English.
- **DÉFINITION:** Definition in French.
- **DEFINIZIONE:** Definition in Italian.

Hierarchical taxonomy

Template for a hierarchical taxonomy, including two columns to set up its metadata and four columns to establish the preferred, alternative label, and the broader or narrower concept. It also includes four properties to define each term in three languages (English, Spanish, and Italian) and to write notes related to the term.

The metadata is defined with the following columns. Note that special characters are not allowed in these columns:

- **ID*:** Identifier that is used to form the International Resource Identifier (IRI) for each term.
- **BASE IRI*:** Represents the exact path to access the resource.
- **SCHEME*:** Additional information used to classify the terms.

Then, the concept is defined with the following columns. Note that the quotes can cause problems if they are not closed.

- **PREFERRED LABEL:** Describe the concept in a clear and unambiguous way. There are 27 columns, each one for each European language.
- **ALTERNATIVE LABEL:** List synonyms, acronyms, or abbreviations available for a concept when necessary. There are 27 columns, each one for each European languages.
- **BROADER CONCEPT:** Represents, in English, that this concept has a broader concept, which means that this concept is more specific. NOTE: a SKOS concept can be attached to several broader concepts at the same time.
- **NARROWER CONCEPT:** Represents, in English, that this concept has a narrower concept, which means that this concept is more generic.
- **DEFINICION:** Definition in Spanish.
- **DEFINITION:** Definition in English.
- **DEFINIZIONE:** Definition in Italian.
- **NOTE:** Additional information.

4.3.3.2 Transformation

In this step of the process is where the mapping rules with the **RMLmapper** transform the input file into a SKOS-based thesauri.

First of all, each template has its own mapping rules, so when the user drops his template, easySKOS checks whether this is a valid structure. Due to the fact that easySKOS does not read the private information of the user, we made the decision of forcing the user to not change the name of the template. By this way, the **setTransformation** method can compare the filename and throws an error if it does not match any of the templates provided. Again, if the user changes the internal structure of the template, an error will be thrown in the next step.

Second, the **transform** method is the one who calls the RMLmapper engine and gives it the mapping rules, along with the template to be transformed. As mentioned above, if the internal structure of the template is altered, the mapping rules would fit the new structure and RMLmapper would throw a run time error to the user.

Finally, the **getMetadata** method retrieves the relevant metadata of the output file and sends it back to the users. The current version retrieves the number of terms

and triplets created, as well as the number of unique objects that are used in the relationships.

4.3.3.3 Result

If there are not errors, the front-end displays a box with a download link (see image 4.7.

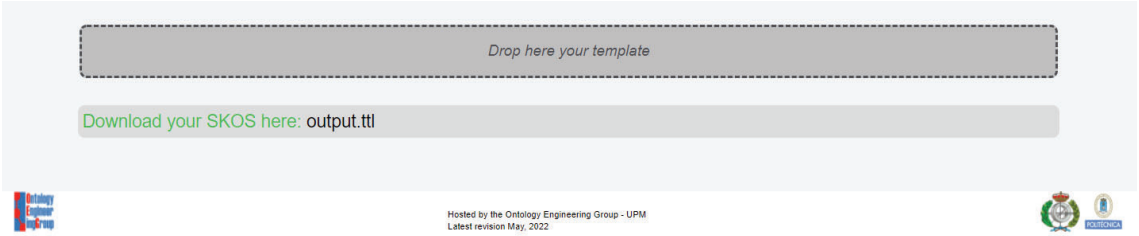

Figure 4.7: Output when there are not errors.

On the contrary, if something happens in the transformation process, a error box is displayed. There are two types of error:

- An error occurred when transforming your template, which means that some columns of the template have been changed or the RMLmapper engine is having problems (see image 4.8).

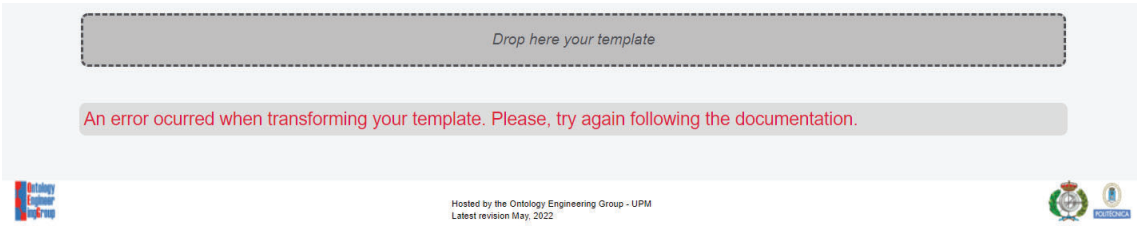

Figure 4.8: Output when there are not errors.

- Cannot set the mapping rules for that file which means that the filename of the template was changed and the converter is not able to link the mapping rules with source file (see image 4.9).

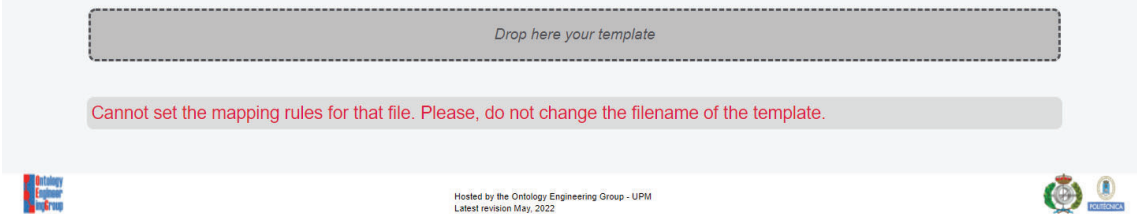

Figure 4.9: Output when the user changes the filename of the template.

Furthermore, if there are special characters like ?,;*ˆ,º$%&, the converter will encode them and the final result may be changed and illegible.

4.4 Evaluation in real use cases

After the implementation of **easySKOS**, we started to evaluate it with real users, monitoring every step they took to see which parts of the process should be fixed. We conducted four interviews with the following structure:

1. Brief introduction to easySKOS and the development team.

2. Half of the interviewees showed an explanation of how **easySKOS** works.

3. Propose the following task to the user: Download one template (each user has to download different templates) and insert five concepts. While they were performing the task, we measured how much time they needed to complete the task and checked if they committed some errors.

4. Review the process with users, asking them how they felt while performing the task, and then share the results (time needed and errors committed).

Finally, we obtained satisfactory results that demonstrate that users can easily follow the four steps of the process and that users who did them without reviewing the documentation first did not make many mistakes. The main problem was when testing the template for hierarchical taxonomy, because it is not normal to duplicate the information, but after the explanation, the users quickly got the message and did not get bothered. Furthermore, the users who showed the explanation did not make errors, although they lasted longer than the other group of testers. All this information can be seen in the Appendix C.

Furthermore, **easySKOS** has been used to transform a CSV file into a crime taxonomy in the context of the Reputation Reporter project[9]. The taxonomy contains more than 90 terms with their hierarchical relationships, following the hierarchical taxonomy template provided by easySKOS.

[9]This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825182: Prêt-à-LLOD.

4.5 Impact analysis

In this chapter, we will discuss the impact that easySKOS could have on different areas related to the SDG (Sustainable Development Goals[10]), such as cultural, economic, social, personal, or business.

The results of this final degree project will greatly contribute to the expansion of the Semantic Web as a paradigm for publishing and sharing knowledge through the Web. In addition, **easySKOS** is built to offer a power tool for transforming semi-structured data in CSV or XLSX files into SKOS thesauri. Consequently, if there are more RDF-based data, there are more data to upload to the open knowledge base of the Web. Therefore, this work could act in the field of action of SDG 4, **Quality Education** and SDG 10, **Reduce Inequalities**.

On the one hand, as more information is shared on the Web, people all over the world (with Internet access) could benefit from this and hence *education may be improved* as a consequence of this process.

On the other hand, the ideal world would be one where the vast majority of information is public and free. Although we do not live in such a scenario, we must work to achieve this goal, and this work may help *reducing these inequalities.*

4.6 Conclusion

In this chapter we recapitulate the work done, pointing out the limits of **easySKOS** and the decisions that were taken, but we can conclude that *main contribution of this project is* **the new way of transforming CSV or XLSX structured data into SKOS thesauri**.

This document has described the state-of-the-art of **mapping languages**. In chapter 2 we could see the comparison between *dedicated mapping languages* RML and R2RML and how RML suits the needs of this project. In addition, we have outlined the human-friendly YARRRML mapping language because it plays an important role in the creation of the mapping rules for the templates that easySKOS provides. Moreover, this paper has presented the specification of **SKOS vocabulary**. The main advantage of using SKOS is that, due to its syntax, thesauri can be expanded, enriched, and exchanged as open linked data (see Chapter 3). Therefore, SKOS was chosen as the vocabulary to build our templates.

[10]The Sustainable Development Goals are a call to action from all kinds of countries to promote prosperity while protecting the planet.

The RML mapping language and the SKOS vocabulary are the basis of the practical part of this project, the **easySKOS** converter. It is a web service deployed on a virtual machine provided by the Ontology Engineering Group and can be accessed via the following URL:

https://terminoteca.linkeddata.es/easySKOS.html

Regarding the bibliographical part of the project, we have witnessed how difficult it is to gather and select good references. It has been a "real" research experience, and we have learnt to organise the large information available on the Web and how important it is to focus the spotlight of the research so that it does not get lost in the process.

Regarding the development part of the project, we have faced the path of building from scratch an operative web service.

On the one hand, the front-end part has been developed using HTML, CSS, and Javascript. Among these programming languages, we did not have experience with HTML and CSS, so this was a major challenge. However, it was never a limitation.

On the other hand, the back-end part has been developed using Java, which is the programming language that we have been learning throughout the degree years. Here, we had to face several limitations. First, the original idea was to offer a versatile template that users could completely modify. The reason was mainly the ability to add new language columns or modify the existing one. However, this feature could not be added because we did not find a way to update in real time the mapping rules needed for the transformation step. Second, we wanted to add a visualisation screen in which the data would be displayed following a graph. This feature was not started because of the limited time we had to complete all other objectives. Finally, the proposed templates are based on the most commonly used SKOS examples. However, we wanted to add one additional template that incorporates the provenance of a term, following the open provenance model [45]. This new feature was stopped because SKOS did not support this property.

In general, the learning objectives that guided the road map of this project have been fully met. Moreover, this work has revealed the amazing knowledge one needs to have for developing a software application (i.e., visual design, accessibility, copy-writing for documentation, web development, REST practises, front-back communication, back-end logic, version control with Github, etc.).

In conclusion, we are proud to show you the final result of this long but interesting journey that started a year ago and today reaches its destination, but not its end.

4.7 Future work

As mentioned above, during the development of **easySKOS** we have encountered many problems and limitations that restrict our purpose. The main goal of the project was to create a complete and versatile template in which the user could add, delete, and modify each column. However, technical and physical limitations appeared along the way, but it should be a logical action line to be achieved in the near future. Also, for analytical purposes, it would be interesting to see how much time the average user spends using the converter. It would also be interesting to analyse how many users read the documentation before using the converter. This information would give us more information on how to improve the user interface and user experience.

In addition, the project was born with the aim of reducing the barriers for people who do not know about the linked data domain. **easySKOS** provides the basis for the creation of linked data from semi-structured data sources. However, future studies should aim to implement new features, such as the possibility of sharing the generated SKOS with Terminoteca[11] or any other open platform. In this way, future studies should focus on the part of sharing and publishing data to help in the expansion of the benefits of linked data and contribute to the generation of the open knowledge graph (see Appendix A).

[11] https://terminoteca.linkeddata.es/

IV

Automated mapping process for creating SKOS thesauri

PART III. APPENDIX

Appendix A

The Linked Open Data Cloud

The following picture shows every document uploaded to **Linked Open Data Cloud** with its category.

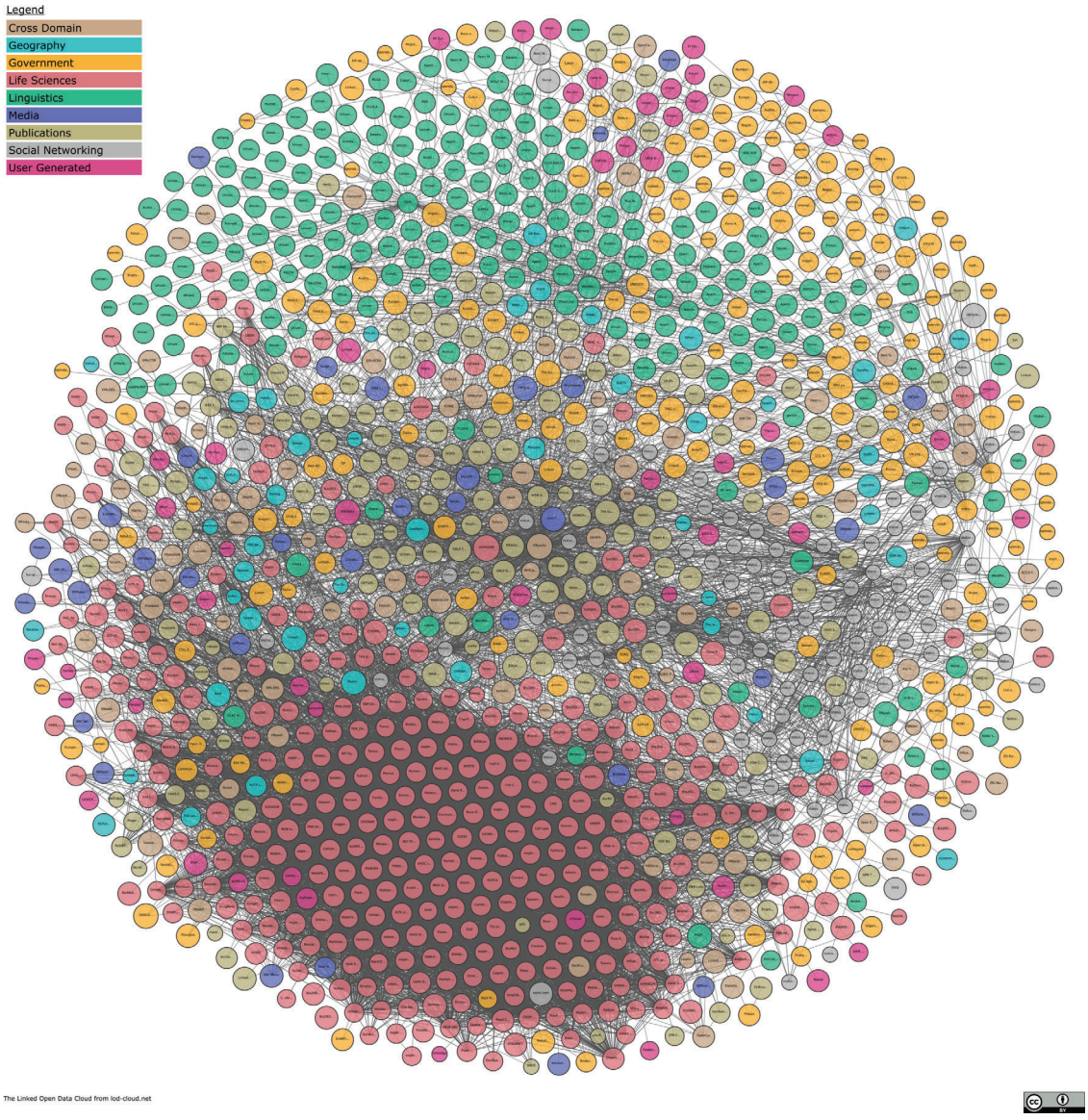

Figure A.1: The Linked Open Data Cloud 2022.

Appendix B

SKOS vocabulary

This table gathers the properties of each SKOS property:

Table B.1: SKOS vocabulary

URI	PROPERTY DEFINITIONS
skos:Concept	Instance of owl:Class
skos:ConceptScheme	Instance of owl:Class Is disjoint with skos:Concept.
skos:inScheme	Instance of owl:ObjectProperty rdfs:range = skos:ConceptScheme
skos:hasTopConcept	Instance of owl:ObjectProperty rdfs:domain = skos:ConceptScheme rdfs:range = skos:Concept Is owl:inverseOf the property skos:hasTopConcept
skos:topConceptOf	Instance of owl:ObjectProperty Is sub-property of skos:inScheme
skos:altLabel	Instance of owl:AnnotationProperty Is sub-property of rdfs:label rdfs:range = RDF plain literals
skos:hiddenLabel	Instance of owl:AnnotationProperty Is sub-property of rdfs:label rdfs:range = RDF plain literals

URI	PROPERTY DEFINITIONS
skos:prefLabel	Instance of owl:AnnotationProperty Is sub-property of rdfs:label rdfs:range = RDF plain literals A resource has no more than one value of skos:prefLabel per language tag
skos:notation	Instance of owl:DatatypeProperty
skos:changeNote	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:definition	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:editorialNote	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:historyNote	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:note	Instance of owl:AnnotationProperty
skos:scopeNote	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:example	Instance of owl:AnnotationProperty Is sub-property of skos:note
skos:broader	Instance of owl:ObjectProperty Is sub-property of skos:broaderTransitive
skos:broaderTransitive	Instance of owl:ObjectProperty Is sub-property of skos:semanticRelation Instance of owl:TransitiveProperty
skos:narrower	Instance of owl:ObjectProperty Is owl:inverseOf skos:broader sub-property of skos:narrowerTransitive
skos:narrowerTransitive	Instance of owl:ObjectProperty Is sub-property of skos:semanticRelation Instance of owl:TransitiveProperty Is owl:inverseOf the property skos:broaderTransitive
skos:related	Instance of owl:ObjectProperty Is sub-property of skos:semanticRelation Instance of owl:SymmetricProperty Is disjoint with the property skos:broaderTransitive
skos:semanticRelation	Instance of owl:ObjectProperty rdfs:domain = skos:Concept rdfs:range = skos:Concept

URI	PROPERTY DEFINITIONS
skos:exactMatch	Instance of owl:ObjectProperty Is sub-property of skos:closeMatch Instance of owl:SymmetricProperty Instance of owl:TransitiveProperty
skos:mappingRelation	Instance of owl:ObjectProperty Is sub-property of skos:semanticRelation
skos:narrowMatch	Instance of owl:ObjectProperty Is sub-properties of skos:mappingRelation Is sub-property of skos:narrower Is owl:inverseOf skos:broadMatch
skos:relatedMatch	Instance of owl:ObjectProperty Is sub-properties of skos:mappingRelation Is sub-property of skos:related Instance of owl:SymmetricProperty
skos:Collection	Instance of owl:Class rdfs:domain = skos:Collection Is disjoint with each of skos:Concept and skos:ConceptScheme.
skos:OrderedCollection	Instance of owl:Class Is sub-class of skos:Collection
skos:member	Instance of owl:ObjectProperty rdfs:range = the union of classes skos:Concept and skos:Collection
skos:memberList	Instance of owl:ObjectProperty rdfs:domain = skos:OrderedCollection rdfs:range = rdf:List Instance of owl:FunctionalProperty
skos:broadMatch	Instance of owl:ObjectProperty Is sub-properties of skos:mappingRelation Is sub-property of skos:broader
skos:closeMatch	Instance of owl:ObjectProperty Is sub-properties of skos:mappingRelation Instance of owl:SymmetricProperty

Appendix C

Results from the evaluation

We conducted four interviews with four different students from the Polytechnic University of Madrid. The students were asked to perform a task and then shared their feedback with us.

The first task consisted of building a glossary related to computer engineering. In the picture C.1 you can see the data used in this task.

Spanish	English	French	Definition in Spanish	Definition	Definition in French
Abstraccion	Abstraction	Abstraction	El proceso de eliminar los detalles físicos, espaciales o temporales	In software engineering and computer science, the process of removing physical, spatial, or temporal details[2] or attributes in the study of objects or systems in order to more closely attend to other details of interest; it is also very similar in nature to the process of generalization	Le processus de suppression des détails physiques, spatiaux ou temporels
Algoritmo	Algorithm	Algorithme	Especificación de cómo resolver una clase de problemas	An unambiguous specification of how to solve a class of problems. Algorithms can perform calculation, data processing, and automated reasoning tasks. They are ubiquitous in computing technologies.	Spécification non ambiguë de la façon de résoudre une classe de problèmes. Les algorithmes peuvent effectuer des tâches de calcul, de traitement de données et de raisonnement automatisé. Ils sont omniprésents dans les technologies informatiques.
Afirmación	Assertion	Affirmation	Declaración de que un predicado es siempre cierto en un punto de la ejecución.	In computer programming, a statement that a predicate (Boolean-valued function, i.e. a true–false expression) is always true at that point in code execution	En programmation informatique, déclaration selon laquelle un prédicat (fonction booléenne, c'est-à-dire une expression vrai-faux) est toujours vrai à ce stade de l'exécution du code
Numero binario	Binary number	Nombre bina	Número expresado en base-2 y que solo puede tomar como valores 0 o 1.	In mathematics and digital electronics, a number expressed in the base-2 numeral system or binary numeral system, which uses only two symbols: typically 0 (zero) and 1 (one).	En mathématiques et en électronique numérique, nombre exprimé dans le système numérique de base 2 ou système numérique binaire, qui n'utilise que deux symboles : généralement 0 (zéro) et 1 (un).
Byte	Byte	Octet	Unidad de información digital que contiene 8 bits.	A unit of digital information that most commonly consists of eight bits, representing a binary number.	Une unité d'information numérique qui se compose le plus souvent de huit bits, représentant un nombre binaire.

Figure C.1: Data provided for the first task.

The second task consisted of building a hierarchical crime taxonomy, similar to the example provided in the documentation. The data provided for this task can be seen in the picture C.2.

TYPE OF TAXONOMY	TERM	ALIAS	PARENT	CHILDREN	DEFINITION	NOTE
crime	Legal Requirement			Disqualification for public employment or office; Dispatch of special commissaries; Prefecture		Update on May
crime	Disqualification for pu	Legal disqualification	Legal Requirement			Update on May
crime	Dispatch of special commissaries		Legal Requirement			Update on May
crime	Prefecture	Jurisdiction	Legal Requirement			Update on May
crime	Organisational crime	Corporate crime			Conduct of a corporation, or of employees acting On behalf of a corporation, which is proscribed and punishable by law. C.V2º1	Update on May

Figure C.2: Data provided for the second task.

In **task 1**, the interviewees were a computer engineer who knew about the semantic web and an architect with any relationship with this field of study. The explanation was given to the architect, but both completed the task without problems. The

engineer took two minutes, and the architect lasted three minutes and forty-two seconds.

In relation to **task 2**, the first interviewee was a nurse with any knowledge of the semantic web, and the second was another computer engineering graduate. The explanation was given to the nurse.

On the one hand, the nurse asked us questions while she was doing the task (i.e., how should I identify the terms?; What do I have to put in the broader/narrower concept columns, the ID or the name?; The duplicated row has to have different IDs?). Therefore, she did not make any mistakes, but the task took her six minutes.

On the other hand, the engineer did not read the documentation and we did not explain anything to him. As a result, he was unable to finish without errors. Here, the problem was that he did not duplicate the lines, so the output had not had all the information (i.e., there were no hierarchical relationships). However, after the task, he read the documentation and was quick to spot his errors.

Bibliography

[1] David Reinsel-John Gantz-John Rydning et al. “The digitization of the world from edge to core”. In: *Framingham: International Data Corporation* (2018), p. 16.

[2] Emanuel Knill. “Quantum computing”. In: *Nature* 463.7280 (2010), pp. 441–443.

[3] Priyanshu Srivastava and Rizwan Khan. “A review paper on cloud computing”. In: *International Journal of Advanced Research in Computer Science and Software Engineering* 8.6 (2018), pp. 17–20.

[4] Elaine Rich, K Knight, and SB Nair. *Artificial intelligence third edition*. 2009.

[5] Michael Friendly. “A brief history of data visualization”. In: *Handbook of data visualization*. Springer, 2008, pp. 15–56.

[6] Tim Berners-Lee, James Hendler, and Ora Lassila. “The semantic web”. In: *Scientific american* 284.5 (2001), pp. 34–43.

[7] H. Paulheim. “Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods.” In: *Springer* (2016).

[8] Richard Cyganiak, David Wood, Markus Lanthaler, Graham Klyne, Jeremy J Carroll, and Brian McBride. “RDF 1.1 concepts and abstract syntax”. In: *W3C recommendation* 25.02 (2014), pp. 1–22.

[9] Grigoris Antoniou and Frank van Harmelen. “Web ontology language: Owl”. In: *Handbook on ontologies*. Springer, 2004, pp. 67–92.

[10] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. “Dbpedia: A nucleus for a web of open data”. In: *The semantic web*. Springer, 2007, pp. 722–735.

[11] Gene Ontology Consortium. “The gene ontology resource: 20 years and still GOing strong”. In: *Nucleic acids research* 47.D1 (2019), pp. D330–D338.

[12] María Hallo, Sergio Luján-Mora, Alejandro Maté, and Juan Trujillo. "Current state of Linked Data in digital libraries". In: *Journal of Information Science* 42 (2 Apr. 2016), pp. 117–127. ISSN: 17416485. DOI: 10.1177/0165551515594729.

[13] Tim Berners-Lee et al. *Semantic web road map*. 1998.

[14] A. Miles and S. Bechhofer. "SKOS simple knowledge organization system reference." In: *W3C* (2009).

[15] Tania Tudorache, Csongor I Nyulas, Natalya F Noy, and Mark A Musen. "Using semantic web in ICD-11: three years down the road". In: *International Semantic Web Conference*. Springer. 2013, pp. 195–211.

[16] Christian Bizer, Tom Heath, and Tim Berners-Lee. "Linked data: The story so far". In: *Semantic services, interoperability and web applications: emerging concepts*. IGI global, 2011, pp. 205–227.

[17] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. "Systematic mapping studies in software engineering". In: *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*. 2008, pp. 1–10.

[18] M. Arenas, A. Bertails, E. Prud'hommeaux, and J. Sequeda. "A Direct Mapping of Relational Data to RDF. Recommendation, World Wide Web Consortium (W3C)". In: *W3C* (2012).

[19] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. "The MASTRO system for ontology-based data access". In: *Semantic Web* 2.1 (2011), pp. 43–53.

[20] Christoph Lange. "Krextor-an extensible framework for contributing content math to the Web of Data". In: *International Conference on Intelligent Computer Mathematics*. Springer. 2011, pp. 304–306.

[21] Juan F Sequeda, Freddy Priyatna, and Boris Villazón-Terrazas. "Relational Database to RDF Mapping Patterns." In: *WOP*. 2012.

[22] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. "RML: a generic language for integrated RDF mappings of heterogeneous data". In: *Ldow*. 2014.

[23] François Scharffe, Laurent Bihanic, Gabriel Képéklian, Ghislain Atemezing, Raphaël Troncy, Franck Cotton, Fabien Gandon, Serena Villata, Jérôme Euzenat, Zhengjie Fan, et al. "Enabling linked data publication with the Datalift platform". In: *Workshops at the twenty-sixth aaai conference on artificial intelligence*. 2012.

[24] A. Dimou. "R2RML and RML Comparison for RDF Generation, their Rules Validation and Inconsistency Resolution". In: *Ghent University* (May 2020).

[25] Franck Michel, Loïc Djimenou, Catherine Faron Zucker, and Johan Montagnat. "xR2RML: Relational and non-relational databases to RDF mapping language". PhD thesis. CNRS, 2017.

[26] M. Lefrançois, A. Zimmermann, and N. Bakerally. "A SPARQL extension for generating RDF from heterogeneous formats." In: *Semantic web 14th International Conference, ESW* (2017).

[27] C. B. Aranda, O. Corby, S. Das, L. Feigenbaum, P. Gearon, B. Glimm, S. Harris, S. Hawke, I. Herman, N. Humfrey, N. Michaelis, C. Ogbuji, M. Perry, A. Passant, A. Polleres, E. Prud'hommeaux, A. Seaborne, and G. T. Williams. "SPARQL 1.1 Overview". In: *W3C* (2013).

[28] B. De Meester, P Heyvaert, and T Delva. "RDF Mapping Language (RML)". In: *Ghent University* (2020).

[29] Pieter Heyvaert, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. "Declarative Rules for Linked Data Generation at your Fingertips!" In: *Proceedings of the 15th ESWC: Posters and Demos*. 2018.

[30] E. Iglesias, S. Jozashoori, D. Chavez-Fraga, D. Collarana, and M. E. Vidal. "SDM-RDFizer: An RML Interpreter for the Efficient Creation of RDF Knowledge Graphs". In: (). DOI: 10.1145/3340531.3412881.

[31] Umutcan Şimşek, Elias Kärle, and Dieter Fensel. "RocketRML - A nodejs implementation of a use-case specific RML mapper". In: *CEUR Workshop Proceedings* 2489 (2019), pp. 46–53. ISSN: 16130073.

[32] P. Heyvaert, A. Dimou, A. L. Herregodts, R. Verborgh, D. Schuurman, E. Mannens, and R. Van de Walle. "RMLEditor: A Graph-Based Mapping Editor for Linked Data Mappings". In: *ESWC* (2016).

[33] A. Iglesias-Molina, L. Pozo-Gilo, D. Doña, E. Ruckhaus, D. Chaves-Fraga, and O. Corcho. "Mapeathor: Simplifying the Specification of Declarative Rules for Knowledge Graph Construction". In: ().

[34] Jon Phipps and Daniel Rubin. *SKOS Use Cases and Requirements*. 2009. URL: http://www.w3.org/TR/2009/NOTE-skos-ucr-20090818/Latestversion: http://www.w3.org/TR/skos-ucr.

[35] H. Brownson. "Proceedings of the International Study Conference on Classification for Information Retrieval". In: *ASLIB* (1957), pp. 99–100.

[36] Juan-Antonio Pastor-Sanchez, Francisco Javier Martínez Mendez, and José Vicente Rodríguez-Muñoz. *Advantages of thesaurus representation using the Simple Knowledge Organization System (SKOS) compared with proposed alternatives*. URL: http://informationr.net/ir/14-4/paper422.html.

[37] Thomas Schandl and Andreas Blumauer. *PoolParty: SKOS Thesaurus Management Utilizing Linked Data*. 2010, pp. 421–425. URL: http://www.openrdf.org/doc/sesame2/system/ch05.html.

[38] Osma Suominen, Henri Ylikotila, Sini Pessala, Mikko Lappalainen, Matias Frosterus, Jouni Tuominen, and Thomas Baker. *Publishing SKOS vocabularies with Skosmos*. URL: http://zbw.eu/stw/.

[39] Thomas Bandholtz, Till Schulte-Coerne, Robert Glaser, Joachim Fock, and Tim Keller. *iQvoc-Open Source SKOS(XL) Maintenance and Publishing Tool*. URL: http://apps.innoq.com/iqvoc/about.html.

[40] A. Miles and S. Bechhofer. *SKOS Simple Knowledge Organization System Reference*. URL: http://www.w3.org/TR/2009/REC-skos-reference-20090818/Latestversion:http://www.w3.org/TR/skos-reference.

[41] Osma Suominen and Eero Hyvönen. *LNAI 7603 - Improving the Quality of SKOS Vocabularies with Skosify*. URL: http://www.seco.tkk.fi/.

[42] World Wide Web Consortium et al. "RDF 1.1 Turtle: terse RDF triple language". In: (2014).

[43] Mark Richards. *Software architecture patterns*. Vol. 4. O'Reilly Media, 2015.

[44] Addy Osmani. *Learning JavaScript Design Patterns: A JavaScript and jQuery Developer's Guide*. O Reilly Media, 2012.

[45] Luc Moreau, Juliana Freire, Joe Futrelle, Robert E McGrath, Jim Myers, and Patrick Paulson. "The open provenance model: An overview". In: *International provenance and annotation workshop*. Springer. 2008, pp. 323–326.

Este documento esta firmado por

Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
Fecha/Hora	Wed Jun 01 21:28:57 CEST 2022
Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
Numero de Serie	561
Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)