FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Masters' Degree in Informatics Engineering
Dissertation

ESOTERIC

Exploring domain specific industry verticals for USDL

July 2, 2013

João Duro
jduro@student.dei.uc.pt

Advisor at DEI:
Jorge Cardoso

Abstract

Thanks to the considerable growth of the service industry we now have access to all kinds of services provided by all kinds of people. This motivated the creation of the Unified Service Description Language (USDL) as a standard to describe services. In effort to merge the business, operational and technical perspectives of service systems. However, general purpose languages are often not expressive enough to capture the specificities of domain specific industry verticals. In order to attain the benefits of using a domain specific approach we build a computer-readable standard model to describe educational services by means of extending Linked USDL.

This model is focused on the curricular objectives of a given course or degree by using the Bloom's taxonomy to describe these objectives, which are divided into three different dimensions: Cognitive, Knowledge and Skill. With such model we hope to solve problems in the education domain, such as the lack of standard information and lack of structured data on courses from different universities or on online distance platforms. To validate and evaluate the model, we created a service dataset by extracting courses and degrees from existing platforms by means of a scraper. Also, we developed a web application that provides visualization of the courses by means of graphics and tables, which helps to establish relations between them and to answer important society questions regarding education. To achieve this, some challenges, such as the description of the learning outcomes and objectives of a given course, were overcome by using the Bloom's taxonomy intersected with a context or an area of expertise. Therefore, we are able to capture the essence of each learning objective.

This approach has implications to society by granting detailed standard descriptions on each course by using a standard model such as *Linked USDL4EDU*. Therefore, allowing the establishment of a match between these types of services. Therefore, this enables, for example, the creation of new degrees based on courses of universities from all around the world, and the exchange of information about existing courses between universities.

Keywords: education , domain , services , Linked usdl , Bloom , learning outcomes

Acknowledgements

I would like to thank my supervisor, Jorge Cardoso, for all the help, incentive, and attention given to this thesis, including sharing alternative points of view.

I would also like to thank the rest of the Genssiz research group for all the insightful discussions and the useful feedback provided.

In addition, I would like to thank my grandparents, Ana Maria Duro and Abilino Duro, for the everyday support, my ants, Cristina Duro and Angelina Pena, and my girlfriend, Catarina Pereira, for her patience and support. I am also grateful to the help provided by Filipe Coelho and Maria da Conceição Pereira.

Lastly, I would also like to thank my Bachelor and Master's degree colleagues for the companionship and shared knowledge over all these years.

João Duro

Contents

List of Tables

List of Figures

List of Acronyms

ACM Association for Computing Machinery

API Application programming interface

DSL Domain-Specific Language

ECS Electronics and Computer Science

FOAF Friend of a Friend

GR Good Relations

HTTP Hypertext Transfer Protocol

IT Information technology

LOD Linked Open Data

MIT Massachusetts Institute of Technology

MOOC Massive Open Online Course

OECD Organisation for Economic Co-operation and Development

RDF Resource Description Framework

SKOS Simple Knowledge Organization System

SVN Service Value Networks

UC University of Coimbra

URI Uniform Resource Identifier

URL Uniform Resource Locator

USDL Unified Service Description Language

XML Extensible Markup Language

Listings

1 Introduction

This first chapter introduces the reader to the research background in Section 1.1, which depicts the nature of current services along with a brief description of USDL, a language to describe services, followed by a pertinent classification of services according to their interaction with the costumer. The motivation to this work is shown in Section 1.3 by means of analogies to other domains of studies. Section 1.4 describes the problems in the education domain regarding to the information shown in the web platforms of some universities. Section 1.5 proposes the objectives of this work, which are followed by the challenges identified and the approach that will be considered to attain the previously identified objectives, in Section 1.6.

1.1 Background

The service industry, which involves the provision of services (also known as "intangible goods") to other businesses as to final costumers, has been growing very rapidly, as can be seen in Figure 1.1, retrieved from OECD[1], an international economic organisation that focus on stimulating economic progress and world trade by means of publishing books, reports and statistics. Owing to this growth we now have access to a wide range of services provided by all kinds of industries. In order to be able to find the right services one needs to access a services' description tool. In order to be able to describe a service we must first understand what a service is. Many have tried to describe a service and still many descriptions can be found, according to Cardoso et al. [8]:

> *"It is considered to be an activity which is intangible by nature which is provided by a service provider to a service consumer to create a value possibly for both parties. Services normally provide a human value in the form of work, information, advice, skills and expertise. In traditional economies, services are typically discovered and invoked manually, but their realization maybe performed by automated or manual means (or a combination of both)"*

[1] http://www.oecd.org/

We consider one of the most important parts of this definition: "create a value possibly for both parties", because if no value is created by the service provisioning then there is no point in providing the service, because the goal of consuming a service is to change the state of the service receiver/consumer [48]. This definition also points out that services are typically discovered, selected and invoked manually, for example if someone needs to repair a car, the first step is to find someone that can perform this service (discovery), choose the best service according to their needs (selection) and then perform the service (invocation). The issue is that we are emerging to a world where this is false (discovery, selection and invocation manually), where every (or almost) service discovery and invocation is done in a semi-automated way, but though, despite this, most of its consumption is performed in person. For example, the taskrabbit[2], a online marketplace that allows users to outsource small jobs and tasks (or services) to others in their neighbourhood, allows people to hire services through the Internet in a semi-automated way but the service consumption will be done in person.

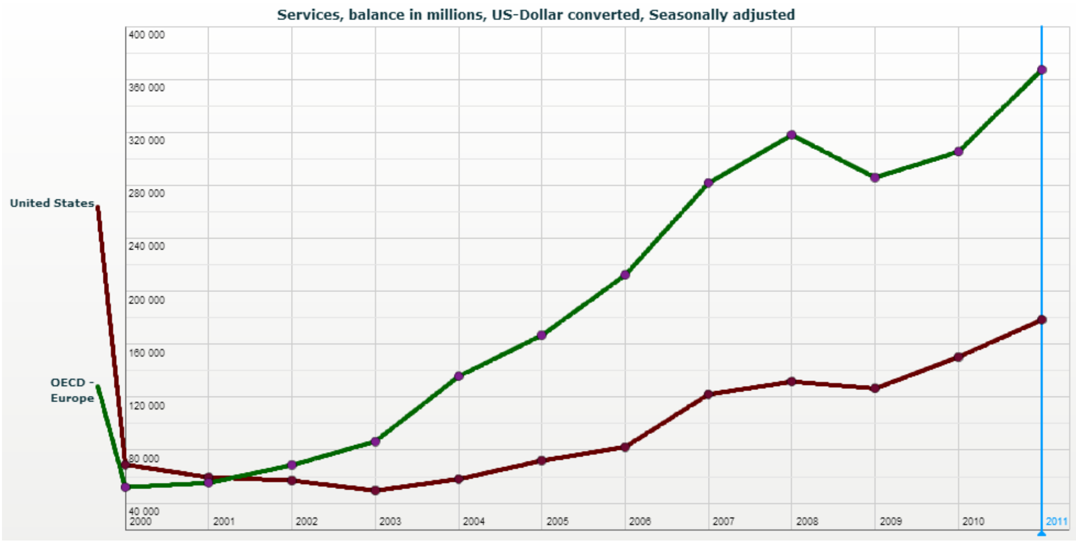

Figure 1.1: Services, balance in millions in US-Dollars according to OECD[38]

The industry is also transitioning from goods-based to a service-based focus and is struggling to find a balance between keeping trade secrets and exposing information about their products or services, which makes the service description a very subtle aspect in business industry. According to Porter in [41], on one side the business industry must worry about existing competitors and keep their service descriptions secret in order to maintain their business running, and on the other hand there is the cooperation between their rivals, their suppliers and customers who demands that the service is well described and transparent allowing all entities to understand the service value and service workflow, allowing to understand where

[2]`https://www.taskrabbit.com/`

can the value be added in the service and how to reduce costs in their workflows. This applies not only to industry but to all kinds of services.

Such a need to describe all kinds of services in a standard way lead to the creation of the Unified Service Description Language (USDL) [45], a standard to describe services through XML and Ecore (the meta-modelling language of EMF). USDL is aimed to cover all kinds of services and provides a unified way to "fit them all", despite of their business area or domain. Even though USDL made progress regarding service description, this language presents some disadvantages. Among them is its complexity, especially to beginners and the fact it that can not be extended to new domains. Because of this a new version of USDL was built using Semantic Web technologies, called Linked USDL [39], emerging from Linked Data initiatives. These initiatives focus on using the Web to connect related data by means of semantic that was not previously linked. This is the reason for using the term linked. Linked USDL was designed based on Linked Data principles [7]. From a technical perspective, Linked Data prescribes a set of principles that will be followed when publishing data on the Web, so that it is machine-readable, its meaning is explicitly defined, and it can be interlinked with other datasets [7]. To go from USDL to Linked USDL every single element was reviewed and most were replaced or even deleted to produce a tool easier to use. Linked USDL was built using RDF[3], a standard model for data interchange on the web, which encouraged users to implement new elements which inherit from existing ones, producing a much more flexible tool to describe services.

Having a description language such as Linked USDL makes the discovery, selection and invocation easier for both parties, when they are semi-automated, because the machines can have the semantic meaning of each detail of the service, and with an Information System the invocation could be semi-automated, as stated before. However several domains of services exist (e.g. education, telecommunications or healthcare services), thus it would probably be important to create domain specific languages by means of extending Linked USDL to describe the services included in each division of domain. This thesis will explore this question in the education domain, since this kind of approach could not be done to all domains on time. Although there is no standard language to describe educational services, if we somehow needed to create a new degree based on courses provided by world wide universities and we needed to describe this new degree or each of the courses, we would have to transform the information available at each university platform into Linked USDL structure using, for example, scrapers, a computer software technique of extracting information from websites. In some universities, an ontology was created to describe the resources available. Using these ontologies we could use algorithms to match [49], merge and align these ontologies to transform the data into Linked USDL structure. Some of these universities ontologies were studied in the domain knowledge acquisition phase, since they represent the important elements in the Universities. The ontologies studied were:

[3]Resource Description Framework (RDF) is a family of World Wide Web Consortium (W3C) specifications originally designed as a meta data data model

Electronics and Computer Science - ECS Ontology [4] despite its lack of documentation, this is one of the more complete ones, describing the university and all of its resources, such as: professors, researchers, publications, degrees, among others. It has 88 classes and 80 properties. This ontology has as predecessor the AKT Reference Ontology[5], which has been developed by the AKT partners to represent the knowledge used in the CS AKTive Portal testbed [47].

Univ-Bench Ontology [6] this ontology was developed in order to be used as a benchmark to evaluate the performance of repositories with respect to extensional queries over a large data set that commits to a single realistic ontology. Despite its large number of classes, with a total of 99 classes and 62 properties, it seems that it is used only in that single benchmark.

University Ontology [7] This ontology was built in the SHOE[8] project of the University of Maryland and it defines elements to describe universities and its activities. This ontology has a very good hierarchy on the classes and appears to be complete, but we never had access to the full RDF file to proceed to its analysis.

We believe that these ontologies are somehow part of the existing technologies in the field of describing educational resources and could be important in the analysis of the important properties of an educational service.

1.2 Service Classification

Since this work aims to create a domain-specific description of services, we must first look at the classification of services to better understand the services' characteristics and the domains. In this case, according to [50][40], the services can be classified by how personal the service is and if it serves a wide range of costumers or not. If a service is personal it means that the interaction (when the value of the service is created) is done in person, i.e. the service consumer and the service provider are both present in the interaction phase. These two characteristics of services are important to understand the classification shown in Figure 1.2, which leads to the following classification:

[4] http://id.ecs.soton.ac.uk/docs/
[5] http://www.aktors.org/ontology/
[6] http://swat.cse.lehigh.edu/projects/lubm/
[7] http://www.cs.umd.edu/projects/plus/SHOE/onts/univ1.0.html
[8] http://www.cs.umd.edu/projects/plus/SHOE/

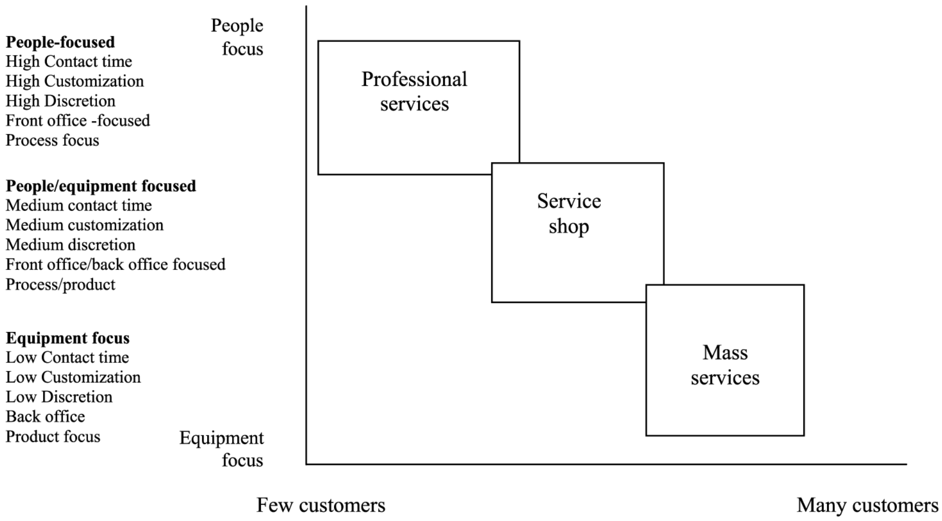

Figure 1.2: Classification of Service Processes, according to [50]

Human-to-Human the service is personal and is people-focused, and in this kind of services usually there is a shorter number of clients, because it would be almost impossible to have Human-Resource capabilities to handle a big number of consumers. Otherwise the service provider would need to have some automated processes in order to be able to handle all the service consumers. The grocery can be compared to a multinational retail corporation. In the first one there is a shorter number of costumers compared to the second one and it would be impossible for the grocery to handle the same number of costumers of the retail corporation without automate processes in their service provisioning. In other words, this type of services (grocery) have a high customization and contact time between both parties and the value of the service is added in front-office, i.e. in the interaction phase, and such interaction is performed in person, i.e. between the service consumer and provider. Marlon et al. [16] calls these services as *"Manual"*, as shown in Table 1.1, whilst Silvestro et al. [50] labels them as *"Professional Services"*. Examples are: education, healthcare and advocacy services.

Software-to-Human in this case it is easier for the service supplier to handle a big number of clients or consumers, due to a high standardization and a limited time with the consumer, when compared with the *Human-to-Human* services. The value of the service is added in the back-office, i.e. it is not in contact with the consumer. Marlon et al. [16] calls these services as *"Fully Automated"*. Silvestro et al. [50], the on other hand, calls these services *"Mass Services"*. Examples of these services are: Software as a System (SaaS), Cloud Computing [54].

Hybrid Marlon's classification has also a hybrid categorization, in which we can consider the services that are not seen neither as *"Professional"* nor as *"Mass"* services, but lye instead in between the two. Such type of services is classified by Marlon as *"Service Shop"*.

	Fully Automated	**Partially Automated**	**Manual**
Intermediary	Transaction Services Persistence Services	B2B workflow-driven services	Equipment Repair
Final	Web-based info sources Digital Libraries	Telephone banking E-Commerce retailing	Hairdressing Medical Services

Table 1.1: Classification of services according to their degree of automation and their relationship to the consumer.

Marlon et al. [16], points out another characteristic of the service: the type of relationship between the service provider and the consumer, as shown in Table 1.1, in which the rows represent the relationship of the service to its final consumers, while the columns represent the degree of automation. The contents of the cells are examples of services. This characteristic of the service implies that it can be an "Intermediary" relation, where the service supplier and the consumer have always a broker in between, whilst the "Final" relation implies that the consumer had a direct liaison with the service provider. As we can see, there is still a difference between the services which are consumed with a final relationship to the service provider, this denotes that some attention must be paid when describing these types of services. Services with a final relation with the client are related to the Manual services described above, requiring from the service provider more time and attention. These kinds of services are consumed manually, requiring both parties to be involved in the service consumption. The specification of the service must take this into account, in order to become evident to the client what type of service he will be consuming.

Both of these classifications will be useful on the next section to better understand the type of services that this thesis will be dealing with.

1.3 Motivation

We can divide our motivation in two parts: the motivation to use a domain-specific approach in service description and the motivation to build a standard model to describe educational services.

In regard to the first one, as Deursen et al. [53] have said, "in all branches of science and engineering one can distinguish between approaches that are generic and those that are specific". Obviously the service description using the USDL language is a generic approach and may not fit a specific one in a restricted domain equally well, once it is a solution for a big set of problems (domains). Our motivation is that by extending Linked USDL meta-model to a domain-specific approach, allows the service description to better fit the services in that specific domain. This kind of approach has been important in other application areas such as: ERP Systems or Domain-Specific Languages (DSL).

An example of this applicability of domain specific approaches is the SAP ERP business solution, that has 23 distinct industries, allowing them to better fit in each specific industry. As said in [46]:

"You don't run a generic business.
So why use generic solutions?"

This expresses the mind of SAP solutions. In other words, if you have a specific business that needs specific improvements, then you will need to have a specific solution that will fit in your business, in order to produce the outcomes that are expected. The SAP industry solutions provide end-to-end functionality needed to run the enterprise more efficiently and it is available in a list that currently includes 23 industries, shown in Table 1.2.

Aerospace and defence	Automotive
Banking	Chemicals
Consumer products	Engineering, construction, and operations
Financial service providers	Healthcare
High tech	Higher education and research
Industrial machinery and components	Insurance
Media	Mill products
Mining	Oil and gas
Pharmaceuticals	Professional services
Public sector	Retail
Service providers	Telecommunications
Utilities	

Table 1.2: Twenty three different domain industries solutions from SAP

This list was not built with a single thought, this list was built with several years of experience in IT solutions and it took some time for people to be sure that domain-specific approaches were a better solution than a generic one.

Domain-Specific Languages (DSL) are another good example of how domain-specific approaches produce better results than generic ones, because they trade generality for expressiveness in a limited domain [33] providing a much better solution for a smaller set of problems [53]. According to Deursen et al. [53] the benefits of DSL's include the following(among others):

- solutions can be expressed in the idiom and at the level of abstraction of the problem domain;
- increase productivity, reliability, maintainability and portability;
- incorporate domain knowledge, which enables conservation and reuse of this knowledge;

- validation and optimization at the domain level are possible.

We hope to achieve these benefits, at some point in our business verticals architecture once it will be a domain-specific approach to describe the services.

The author in [33] also points out that DSL development is hard, requiring both domain and language development expertise. In our case, we will need both education domain and service description expertise in order to produce our education domain-specific model.

Our efforts will focus on how Linked USDL needs to be extended in a specific approach to gain expressiveness, ease of use and to capture the singularities of domain specific industry verticals. In this work, the domain will be education. Some other domains were considered, including the Telecommunications, but the decision to build the model to the education domain was mostly because:

- Easy access to service descriptions (e.g. research and lecture classes).
- Common operation mode of educational institutes (e.g. most of universities follow a similar operating mode, such as *Bologna Process*).
- This work is embedded in the education domain for a long time, which reduces the time to acquire the knowledge to become an expert in this domain.

Another important motivation, in terms of building a standard model to describe educational services is the increase of distance education platforms, in particular the web platforms, also known as massive open online course (MOOC), such as *Coursera*, *Udacity* and *edX* which are presented in Subsection 2.3. Sebastian Thrun and Peter Norvig [9] started by sending one email offering a class to the world, free of charge, and within two weeks 160000 students from 39 countries had signed to the class, this motivated them to the creation of *Udacity*. With this, becomes clear that this new way of teaching is starting to be well accepted. If we think in one computer-readable model to describe these services we can achieve a scenario where all platforms have their information on each course with the same structure and each online education platform is able to access another's data, creating a global network with educational services. Connecting this, with the previously discussed usage on Universities, we can have this online platforms using resources from universities, thereby increasing the range of education of each educational institution. For example, Coursera has already 2,427,327 students in their courses[10].

1.4 Problem Description

As said in Section 1.3, we can find approaches that are generic and those that are specific. Since USDL is a generic approach which tries to describe all kinds of services (e.g. Software as Service, Web Services) then this language may not cover

[9]In TED conference. Video available in: `http://www.ted.com/talks/peter_norvig_the_100_000_student_classroom.html`
[10]`https://www.coursera.org/`

all points of every single service. Hence, some specific services may have important details in their service processes and workflow that could not be covered using a generic language.

From the various domains studied where Linked USDL could be used, we will consider the education industry. If we somehow needed to create a new degree based on courses provided by world wide universities, we would need to gather information about each course, and we could face some problems. Some of these problems were identified by analysing several universities web platforms and some of the online education platforms. The problems identified were:

Problem 1 - Information does not follow a standard The data about each course and each module in the platforms studied does not follow a standard, as we can see in Table 1.3, where we analysed several universities platforms according to the information available. This means that whenever someone tries to find a specific detail on a course module, such information will be presented in different forms, with different characteristic names and even in different languages.

Problem 2 - Incomplete information As we can see in Table 1.3, some important information does not appear in some Universities. If we look carefully we can notice that the Curricular Unit Record from Universidade de Coimbra does have most information, though that same information is not present in the university platform, meaning that some information is lost. We can also observe that with the exception of the "Course Unit Title" and the "ECTS", there is no other property that is included in all the platforms analysed. This means that whenever someone needs to look at the characteristics of a module, he (she) will have to face all these issues. Also, there is a current lack of description in educational services. For example, in the *Bologna Process* the European Credit Transfer System (ECTS) only values the amount of work that a student invests into an educational module. If we would implement a service description in each module, we would be able to not only measure the amount of work, but also the content or the level of the module. This would have implications, for example in the student exchange programmes it would be much easier to establish equivalence between modules, preventing that each professor have to do this by hand, each time a student requests some ECTS equivalence.

Problem 3 - Non structured data In Table 1.3 is also available information on how structured data is, and in the *Objectives* of a curricular unit the information is never structured, which means that is described in plain text. Only the *Syllabus* and the *Prerequisites* have information with enough structure, and only the University of *Southampton*has its data slightest structured. This may not be a problem to a final user, because people can understand the plain text, but if we consider a scenario where we would need to automate the processes this would become a big problem, because computers can not understand this non structured information without a complex system or model to help.

	Coimbra		Portugal	Foreign		Online Distance Education		
	FUC (FCT-UC)	FCTUC	Universidade Lisboa	University Southampton ECS	University Amsterdam	Coursera	Udacity	edX
Course unit title	—	—	—	—	—	—	—	—
ECTS	—	—	—	—	—			
Contact hours	●			◐		◐		—
Curricular year	—	—	—	—	—			
Semester	—	—	—	—	—			
Course unit type	—	—						
Responsible academic staff member	—	—			—	—	—	
E-mail	—					—		—
Level	—	—		—				
Mode of delivery	—	—			—	—	—	—
Recommended prerequesites	○			●		○	◐	◐
Language(s) of instruction	—	—	—		—			
Other academic staff members involved in the curricular unit	—				—	—	—	—
Objectives of the curricular unit and competencies	○		○	◐	○	◐	○	○
Syllabus	◐		◐	◐	○	◐	●	◐
Demonstration of the syllabus coherence with the curricular unit's objectives	○							
Teaching methodologies (including evaluation)	○		○	◐				
Demonstration of the teaching methodologies coherence with the curricular unit's objectives	○							
Specific Resources	○							
Main bibliography	◐		◐	●	◐	◐		

● - is structured
◐ - is hardly structured
○ - is not structured, plain text
— - present, but not applicable
(empty) - not present

Table 1.3: Modules information on several universities platforms, in Curricular Unit Record from Universidade de Coimbra and Online Education

By correctly identifying the actual problems in the educations domain becomes easier to identify the objectives that need to accomplished to overcome these problems.

1.5 Objectives

After identifying the problems, it becomes clear what we need to do and achieve in order to overcome those problems:

Objective 1 Build a computer-readable model to describe educational services. This primary objective will be the responsible in solving the problems described above, whilst the next ones will be secondary objectives that will focus in improving and evaluating the model to better fit the educational services.

Objective 2 Create a service dataset, i.e. a set of educational services by instantiating the model created before. This objective will be very important in order to evaluate the model, to understand if the model is suitable to the service description.

Objective 3 Develop a tool to combine services allowing to establish a relation between services. At this point it is not clear what kind of relations could be established, but the most evident are: how similar they are and if one is prerequisite to another. This objective will also be important to evaluate the model, since we can develop, for example, algorithms to match educational services allowing to detect if any adjustments are necessary to the model. These algorithms will have their social implication allowing, for instance, the creation of European Degrees from courses provided by European universities through the relations created.

We believe that with these three objectives we are able to attain good expressiveness to the model and to demonstrate the applicability of the model in terms of real world usage.

1.6 Challenges and Approach

Since we aim to build a domain-specific model to describe educational services, as stated in [53] as an analogy to the creation of Domain-Specific Languages (DSL's), one of the first steps when using a domain specific approach is the domain analysis. This analysis started when we identified the problems in the education domain.

To achieve some knowledge in the education domain two types of analysis were done: ontologies from universities were studied, as shown in Section 1.1, and a study of institutional web platforms, that resulted in the Table 1.3, presented in Section 1.3. The starting point on the creation of this table was the Curricular Unit Record from Universidade de Coimbra and at least three different curricular units of each web platform were viewed and studied, in order to avoid misapprehensions. The *n/a* (not applicable) means that the field does not require structured data and when it is hardly structured it means that in most cases the structure depends on the person who populates the fields of each curricular unit/course, which results in inconsistency on the data. Some curricular units have their data structured and others do not.

The universities chosen to study, and the respective reason:

FCTUC is the faculty where this work takes place and we got access to Curricular Unit Records from several courses;

Universidade de Lisboa was chosen to have another example of the same country as the first one, Portugal;

University of Southampton was chosen because some previous research has been done in the ontology developed by this university, when analysing existing universities' ontologies;

University of Amsterdam entered the study because it is the university where the work in Section 2.1.3 was done and since another university from a foreign country was needed;

Online Distance Education was chosen for being a clear example of the need to create computer readable models to describe educational services.

As said in *Objective 1*, the model needs to be computer-readable as Linked USDL is. We will use RDF[11] specification mostly because it is the specification used by Linked USDL, allowing to reuse existing elements both from Linked USDL and Linked Data cloud, and this way allowing the model to be used in a distributed way, e.g., to be used in different platforms or systems.

This thesis will try to focus in, when building the model in RDF, reusing existing elements from existing ontologies or from the Linked data cloud, that already describe components of a University, human resources and even the existing elements in Linked USDL, since that is the goal of using a semantic web technologies. Our work does not impose tools to be used in building the model, once the model can be created using a text editor.

In order to attain *Objective 1* the big challenge is to define a way to describe the *Learning Outcomes* and the *Objectives* of a curricular unit, which could be one of the most important characteristics of this type of services. We believe that this is the key to the success of the model, once the learning outcomes are the major value added by this types of services to the consumers (the learners). If no learning outcomes are achieved by the service consumer then there is no point in consuming the service at all. Chapter 2.2.1 presents the Bloom's Taxonomy [6], which proved to be a good framework to describe both the *Learning Outcomes* and *Objectives* of a curricular unit, because of their great usage and because it is being redefined and improved since 1956. The revision done by Michael Hoffman [26], presented in Subsection 2.2.1.3, which focus on using Bloom's taxonomy to make courses comparable, appears to be the best revision to be applied in the study case of this thesis once it focus in adapting the taxonomy to make courses comparable.

After the creation of the model in RDF we will have to populate it, i.e. instantiating. To do this we must overcome the challenge on the transformation of the data in plain text to the desired model structure. As we can see in Table 1.3, most of the information is neither present or structured in the platforms studied. In order to be

[11] http://www.w3.org/RDF/

able to evaluate if the model is suitable, we will need to populate it, as mentioned in *Objective 2*, and to do so we will focus on two different approaches:

- Populate a service "by hand", using for example a Curricular Unit Record and, this way, describing the educational service of a curricular unit.
- Populate using data from web sites, mostly web platforms of Universities or the online systems shown in Section 2.3, where crawlers will be developed.

After the model is populated, in order to proceed to its final evaluation, we will try to create a tool, as shown in *Objective 3*, that can allow, through the creation of algorithms, to compare the services and proceed to its testing and evaluation. With this algorithm we may be able to: compare educational modules between different points in the world; and create world wide courses, which will allow to decrease the bureaucracy in student exchange programmes.

2
State-of-the-Art

This chapter is divided in four sections. Section 2.1 provides reviews on related work in the field of service description approaches, focusing on the USDL family and on an overview of the e^3-family. Section 2.2 provides an introduction to the models that were investigated and used to describe educational services, focusing on Bloom's taxonomy, which proved to be a good model to describe the learning outcomes achieved by learners. Section 2.3 introduces a brief description on some of the online distance education platforms. This chapter concludes by showing an analogy to the online recruitment process in Section 2.4 which is evidence that the processes used in this work can achieve the desired goals.

2.1 Service Description

2.1.1 Unified Service Description Language

The Unified Service Description Language (USDL) is a language to describe various types of services, ranging from professional to electronic services [3].

USDL is an effort to merge the business, operational and technical perspectives of service systems: the business perspective describes the fundamental properties for characterizing a service; the operational perspective describes the operations that are executed by a service; and the technical perspective concerns the specification of technical information regarding the services offered by an organization [9]. USDL is based on 9 modules: foundation, service level, participants, pricing, legal, service, interaction, functional and technical. Their relation is shown in Figure 2.1.

The USDL family (*-USDL) is composed of α-USDL, which is the original USDL version that was released in 2009. USDL is a data model for describing all kinds of services. Despite its promise, USDL was too detailed because it not only considers the description of the service's technical interface but also takes aspects like legal, pricing, and service level agreement, which makes USDL too complex to implement. The emergence of Linked USDL solved such problems, changing the paradigm of USDL to the domain of semantic web, having computer-readable descriptions of the services using RDF[1], which is a standard model for data interchange on the Web, and uses the Linked Open Data (LOD), which increases interoperability by allowing sophisticated ontology representation techniques [14]. It also greatly simplifies USDL's original structure, having only 4 modules: USDL-Core, USDL-Pricing (for description of prices), USDL-SLA (for description of service level agreements). and USDL-SEC (for description of security issues) [2].

This change reduced the complexity of the data model, making use of namespaces that have been on the market for a long time. Because it uses semantic web technologies, every user can implement new elements from the standard model, also enabling model extensions, for example, for specific domains.

This family of service description languages are classified by Ferrario et al. [17] as purely economic, since they model generic service systems regardless of their technological implementation, merging the business and technological scope of services [17].

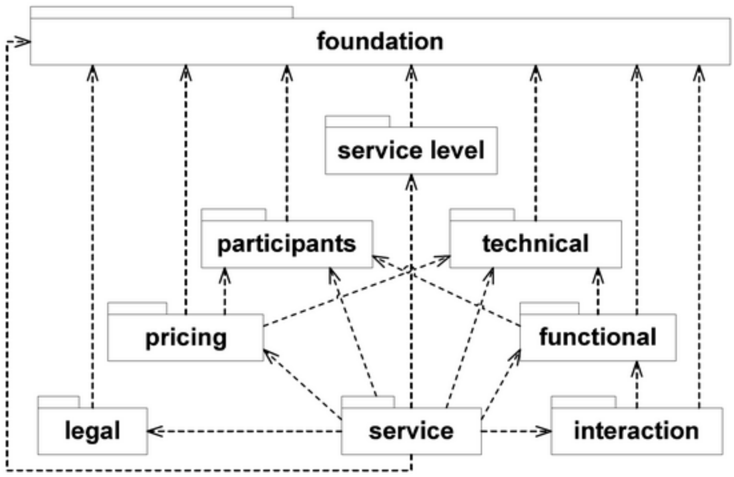

Figure 2.1: USDL dependencies between modules [37]

2.1.2 The e^3-family

The e^3-family provides the following ontologies [24]: e^3-value, e^3-control, e^3-strategy, e^3-alignment, e^3-service and e^3-boardroom.

In this work the ontology that is important to introduce is the e^3-service, an ontology for configuring IT services based on consumer needs [12] proposed in [30]. This ontology puts a greater focus on satisfying consumer needs and on displaying the various value offerings from different services for an easier comparison, as discussed in the previous section. In this ontology, there is an Elementary service performed by a Supplier, that has a certain Benefit (which, in turn, has a Consequence). A Consumer gives a Sacrifice in order to have a Functional need, which is

[1]http://www.w3.org/RDF/
[2]http://www.linked-usdl.org/

concretized by a Want, which in turn is Concretized by a Demand that has a certain Benefit. Wants and Consequences are also connected by an Adds value element [12]. As we can observe, in the next section, Ivan S. Razo-Zapata was able, in his work, to describe educational services using the e^3-service ontology by mapping the resources available in the ontology to the new ones in education, e.g. a *Teaching Course* was mapped into *Value activity* and a *Diploma* was mapped into a *Value Object*. This means that, if an ontology for educational services existed, this mapping would not be necessary and the service could be much more detailed and understood.

2.1.3 Service Value Networks

In order to develop knowledge in this field, and since Iván S. Razo-Zapata's work [44] intersects the e^3-family ontologies in educational services, we decided that it would be important to do a research in his thesis. As we said in Chapter 1, the model we aim to develop could help in building a new degree based on courses provided by world wide universities. This implies that we need to establish a relation between courses from around the world. This relation, in terms of service description is related to Service Value Networks, where services are connected by interactions.

In [44] the main goal is to compose Service Value Networks (SVN) with computer-based support that match specific customer needs. Once a costumer need is defined, service suppliers and enablers must provide a solution to such needs. To fulfil this, two ontologies were created to provide the required input knowledge to compose SVNs: the customer ontology and the service supplier ontology.

The customer ontology takes into account four main concepts that were taken from e^3service [43] customer ontology: Need, Consequence, Want and Functional consequence. A costumer need can be realized by one or more Functional consequences, which are contained in Wants.

In the service suppliers' ontology [43] the e^3service and e^3value ontologies were aligned to build the ontology. Consequently, the final ontology enables the description of how a commercial service can offer valuable resources that have specific Functional consequences. To give an example on how to describe a service, the National Database of Accredited Qualifications[3] was used to populate with educational services.

[3] http://register.ofqual.gov.uk/

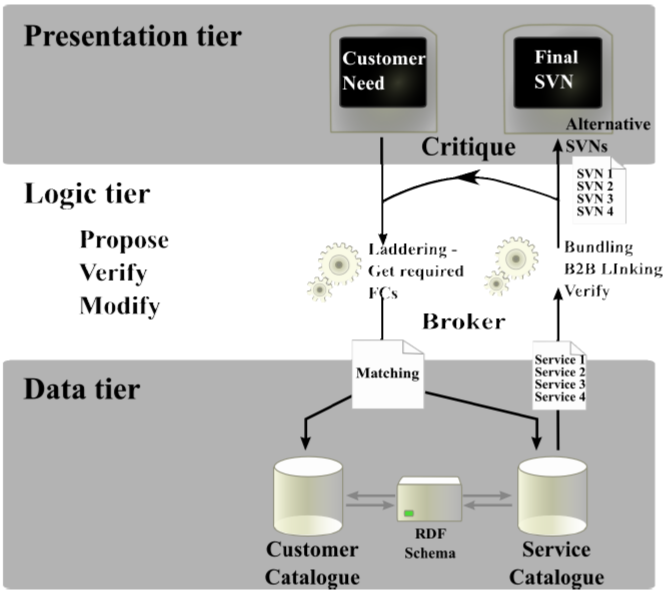

Figure 2.2: The three-tier architecture shown in software prototype

To support this, a software prototype was build, as shown in Figure 2.2. Having the needs of the consumer specified in the ontology, and having a service catalogue with educational services, the logic tier then tries to match those needs with the available functional consequence available in the service catalogue. Subsequently, a SVN is generated. This match requires the user to "critique" the selected SVN, allowing the logic tier to change the required functional consequences which, in turn, enables the generation of a new SVN. This process is repeated until the user is satisfied with the SVN generated. This kind of approach could be a solution for us to build our algorithm to establish a match between courses, and this can be observed in the architecture in Figure 2.2.

This solution to describe educational services, contains a closed world assumption, as the service catalogue is not available to everyone to use. Their services are offered in an off-line way, and the ontologies were built to serve all kinds of needs and service suppliers, despite its domain, which is not specific to the education domain. In this educational case study the needs were mapped onto specific competencies that improve the costumer curriculum vitae, which will allow him to obtain a job. The service catalogue describes the educational services focusing on the awards and certificates that provide specific competencies to the consumer. Thus, it is likely that, if some domain knowledge is used it will be much easier to establish a relation between the needs and the services that are presented in the catalogue.

2.2 Educational Models

2.2.1 Bloom's Taxonomy

Benjamin Bloom introduced the *Taxonomy of Educational Objectives* to facilitate the exchange of test items among faculty at various universities, in order to measure educational objectives. This lead to the creation of their final book in 1956 [6], which is referred to as The Original Taxonomy. At the time it was introduced, potential users did not understand what it meant. Therefore, little attention was given to the original taxonomy at first. However, readers increasingly saw its potential, thus becoming widely known and cited, being translated into 22 languages. As this taxonomy is the most used solution to describe both the learning outcomes and objectives of a curricular unit or course, than the PALO model presented in the Subsection 2.2.2.

Before constructing the taxonomy, Bloom identified three domains of educational activities [26][6]:

Cognitive: classifies the whole of stored knowledge and the learnt processes to handle that knowledge (Knowledge).

Affective: classifies areas of learning that are related to emotions (Attitude).

Psychomotor: classifies areas of learning where physical movement and/or co-ordination are involved (Skills).

The taxonomy itself was build to describe the cognitive and the affective domains, whilst the psychomotor has been the subject of research by other authors.

In Section 2.2.1.1 we will discuss the Original Taxonomy, in Section 2.2.1.2 we will discuss the revised version of the taxonomy, and in Section 2.2.1.3 we will discuss the usage of this taxonomy to make courses comparable.

2.2.1.1 The Original Taxonomy

The original Taxonomy is divided into six major categories: *Knowledge*, *Comprehension*, *Application*, *Analysis*, *Synthesis* and *Evaluation*.

The categories were ordered from simple to complex, and from concrete to abstract. This allows the establishment of a hierarchy in which each simpler category is a prerequisite to the mastery of the next more complex one. The idea is that in order to attain a certain category the learner has to first master a previous simpler one (e.g., to attain the *Synthesis* category the learner must first be able to *Analyse*). The rule of a simpler category being a prerequisite to a more complex one is present in all the subsequent revisions, and it is shown in Figure 2.3 along with a brief description of each category of the taxonomy.

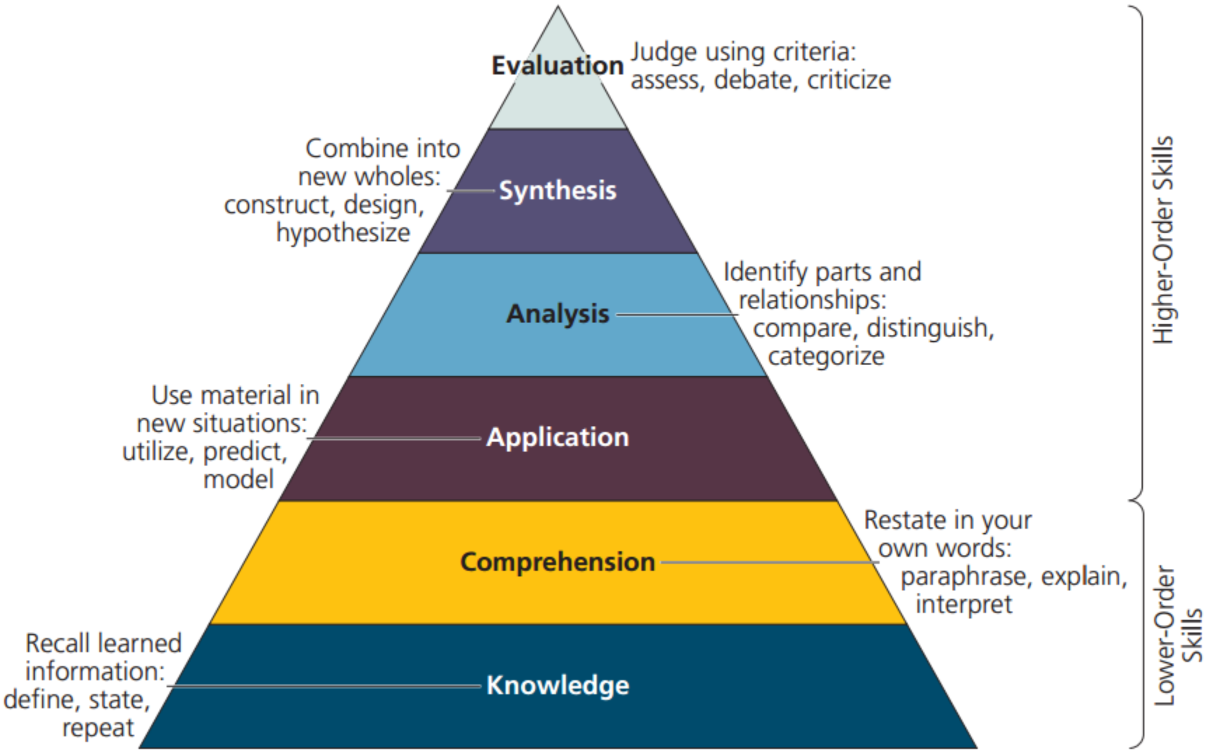

Figure 2.3: Pyramid of cognitive activities - Bloom Taxonomy [27]

Table 2.1 presents a detailed description of each category of the Original Taxonomy, in accordance with [10]. Table 2.1 can be very important when we try to convert non-structured data from web pages to the model, since we may have to establish a match between an objective described in plain text and the desired value in the cognitive dimension.

Knowledge: Recall data or information.	**Examples**: Recite a policy. Quote prices from memory to a customer. Knows the safety rules. **Key Words**: defines, describes, identifies, knows, labels, lists, matches, names, outlines, recalls, recognizes, reproduces, selects, states.
Comprehension: Understand the meaning, translation, interpolation and interpretation of instructions and problems. State a problems in one's own words.	**Examples**: Rewrites the principles of test writing. Explain in one's own words the steps for performing a complex task. Translates an equation into a computer spreadsheet. **Key Words**: comprehends, converts, defends, distinguishes, estimates, explains, extends, generalizes, gives an example, infers, interprets, paraphrases, predicts, rewrites, summarizes, translates.

Continued on next page

Application: Use a concept in a new situation or unprompted use of an abstraction. Applies what was learned in the classroom into novel situations in the work place.	**Examples**: Use a manual to calculate an employee's vacation time. Apply laws of statistics to evaluate the reliability of a written test. **Key Words**: applies, changes, computes, constructs, demonstrates, discovers, manipulates, modifies, operates, predicts, prepares, produces, relates, shows, solves, uses.
Analysis: Separates material or concepts into component parts so that its organizational structure may be understood. Distinguishes between facts and inferences.	**Examples**: Troubleshoot a piece of equipment by using logical deduction. Recognize logical fallacies in reasoning. Gathers information from a department and selects the required tasks for training. **Key Words**: analyzes, breaks down, compares, contrasts, diagrams, deconstructs, differentiates, discriminates, distinguishes, identifies, illustrates, infers, outlines, relates, selects, separates.
Synthesis: Builds a structure or pattern from diverse elements. Put parts together to form a whole, with emphasis on creating a new meaning or structure.	**Examples**: Write a company operations or process manual. Design a machine to perform a specific task. Integrates training from several sources to solve a problem. Revises and process to improve the outcome. **Key Words**: categorizes, combines, compiles, composes, creates, devises, designs, explains, generates, modifies, organizes, plans, rearranges, reconstructs, relates, reorganizes, revises, rewrites, summarizes, tells, writes.
Evaluation: Make judgements about the value of ideas or materials.	**Examples**: Select the most effective solution. Hire the most qualified candidate. Explain and justify a new budget. **Key Words**: appraises, compares, concludes, contrasts, criticizes, critiques, defends, describes, discriminates, evaluates, explains, interprets, justifies, relates, summarizes, supports.

Table 2.1: Bloom Taxonomy - Cognitive Domain

2.2.1.2 The Revised Taxonomy

In the 1990's, a revised Bloom's Taxonomy was published [2][31]. In this revision, three major changes were made: the names in the six categories were changed from noun to verb forms, which were rearranged them as shown in Figure 2.4, and the so called Structural changes, which led to a two dimensional representation. This new representation using two dimensions appears to be much more descriptive on the objectives and points, perhaps, to a more dynamic solution to the problem this work tries to solve.

In this revision, the author splits the taxonomy into two dimensions: one representing the Knowledge dimension (the noun of the phrase), and the other the Cognitive Process dimension (the verb). Given this change, the Cognitive categories where changed to the verb form and some where adjusted to more used terms (see Figure 2.4).

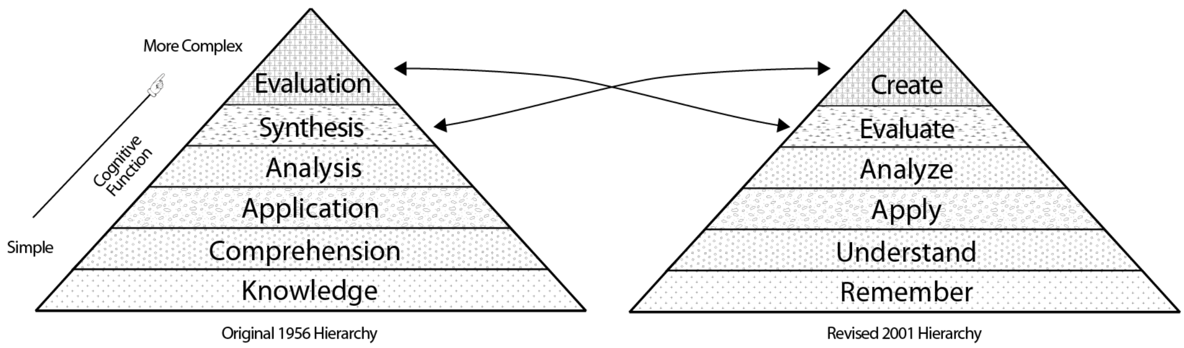

Figure 2.4: Changes on the Bloom's Taxonomy [4]

As represented in the Table 2.2 the intersection of these two dimensions form twenty-four separate cells. The knowledge dimension contains four, instead of three main categories [21]:

Factual Knowledge The basic elements students must know to be acquainted with a discipline or solve problems in it.

Conceptual Knowledge The interrelationships among the basic elements within a larger structure that enable them to function together.

Procedural Knowledge How to do something, methods of inquiry, and criteria for using skills, algorithms, techniques, and methods.

Meta-Cognitive Knowledge Knowledge of cognition in general as well as awareness and knowledge of one's own cognition.

whilst the Cognitive Process Dimension now has 6 categories: *Remember*, *Understand*, *Apply*, *Analyse*, *Evaluate* and *Create*.

Some attention must be taken to the rule presented in Subsection 2.2.1.1. This rule states that the categories are ordered from simpler to more complex ones, with the simplest being *Remember*, and the most complex one being *Create*. Accordingly, it can be stated that, it is much easier to Remember something than to Create something. With this, a simpler category is always a prerequisite to a more complex one.

The Knowledge Dimension	The Cognitive Process Dimension					
	Remember	**Understand**	**Apply**	**Analyse**	**Evaluate**	**Create**
Factual Knowledge	List	Summarize	Classify	Order	Rank	Combine
Conceptual Knowledge	Describe	Interpret	Experiment	Explain	Assess	Plan
Procedural Knowledge	Tabulate	Predict	Calculate	Differentiate	Conclude	Compose
Meta-Cognitive Knowledge	Appropriate Use	Execute	Construct	Achieve	Action	Actualize

Table 2.2: Bloom's Revised Taxonomy: Intersection of the knowledge and cognitive process categories

2.2.1.3 Using Bloom's Taxonomy

In the original taxonomy the authors addressed the issues of cognitive and affective objectives in education, and provided a hierarchy of types of capabilities in each of these domains that could be used as evidence of achievement. The psychomotor dimension, however, needs to be elevated to a more abstract level. Using the hierarchical taxonomy of psychomotor skills proposed by Ferris and Aziz [18], we achieve a new domain: *Skills*. According to Michael Hoffman [26], some levels are dispensable, since they may not be too different from some existing ones. Therefore, this new Skills dimension is present in Table 2.3.

Process Action	Explanation	Example	Level
Recognition of tools and materials	Recognize tools and materials	Recognize a differential equation	1
Basic operation of tools	Set tools in action and perform elementary tasks	Use a derivative to solve the problem	2
Competent operation of tools	Fluently use tools for successful problem solving	Solve problem including boundary conditions	3
Expert operation of tools	Ability to use tools with ease to rapidly, efficiently, effectively, and safely perform task on a regular basis	Select and apply advantageous numerical methods	4

Table 2.3: The skills domain

With this new dimension added to the previous two described in Subsection 2.2.1.2, we now have a full representation using three domains: *Knowledge*, *Cognitive* process and *Skills*. In Figure 2.5 we have an example of a learning objective provided [26], where it is described an educational objective from a lecture on engineering mathematics. Here we can see that the intersections of the three domains results, for each category k of the knowledge dimension, and for each category p of the cognitive process dimension, a skills level s is indicated. In the last row and column is the "centre of gravity" of each corresponding category. In the last cell, we can see that the "centre of gravity" of the learning objective is $(p, k) = (3, 2)$, meaning that the main emphasis of the course program has been placed in category *3* of the *Cognitive process dimension - Apply*. And the centre of gravity of the *Knowledge dimension* is in category 2 - *Conceptual Knowledge*. When we cross, in the table, the point $(p, k) = (3, 2)$ we have the value of the emphasis on the *Skills* domain, which is *4 - Expert operation of tools*.

			cognitive process dimension						
		p	1	2	3	4	5	6	
	k		remember	understand	apply	analyse	evaluate	create	
knowledge dimension	1	factual knowledge	4	2	4	1	1	1	$p_k = 3$
	2	conceptual knowledge	2	4	4	1	1	0	$p_k = 3$
	3	procedural knowledge	2	2	4	1	1	0	$p_k = 3$
	4	meta-cognitive knowledge	1	1	1	0	0	0	$p_k = 2$
			$k_p = 2$	$k_p = 2$	$k_p = 2$	$k_p = 2$	$k_p = 2$	$k_p = 1$	$(k,p) = (3,2)$

Figure 2.5: Diagram to describe the skills-level and cognitive coordinates of a learning objective

We have to remember that the objective of this extension of Bloom's diagram is to characterise the learning objectives of an educational model, and this work also needs to describe the Learning outcomes of an educational service. The main objective of an educational service is to transform the objectives previously established into learning outcomes, and in a perfect scenario both objectives and learning outcomes will be the same. This may not always be true, since students may feel more akin to some topics than others. Even if the learner has achieved some kind of approval to the full course or educational module, this does not imply that the learner has mastered all the topics. Since this is the goal of those who provide the service (the teacher), this work will use this extension of Bloom's taxonomy to describe both *objectives* and *learning outcomes*.

2.2.2 Describing Learning Outcomes

Naijar et al. [35] point out that employers seek people that match particular qualifications, whereas graduates seek jobs that match their qualifications. To establish this balance between both sides, it is necessary that universities have heterogeneous management systems that capture achieved learning outcomes as well as corporate information systems that capture those qualifications. In this thesis, we are interested in the description of achieved learning outcomes. Accordingly, in [35] the author proposes a common data model called The Personal Achieved Learning Outcomes (PALO) Data Model, as shown in Figure 2.6.

The model is composed by the following elements:

Personal Achievement Profile this element will have the collection of learner's achievements with the relation *has part*;

Achievement this element is the achievement record itself. This achievement has an evidence through the relation *has evidence*, has a Context with the relation *has*, and is taken from a Learning Outcome through *states attainment of*;

Learning Outcome this element represents the learning outcome that was achieved by the learner, and the *type* property defines whether it is knowledge, skill or competence. A learning outcome could have a relation named *is related to* to other learning outcome and it has a level;

Level this element captures ranking information about the learning outcome;

Context this element gives meaning to the learning outcome through a set of factors;

Assessment record this element gives information as evidence to the Achievement, e.g. as a certificate or license;

This model can be a solution to describe each of the learning outcomes achieved by the learner when he consumes an educational service. However, this model still lacks a description of each learning outcome, since it is focused mostly on the achievements of a person. Still, we can see some similarities on the Level class used in the Section 2.2.1.3 by Michael Hoffman when using Bloom's taxonomy to compare courses. The idea of a Context class associated to each Learning Outcome is also used in the prototype of the model.

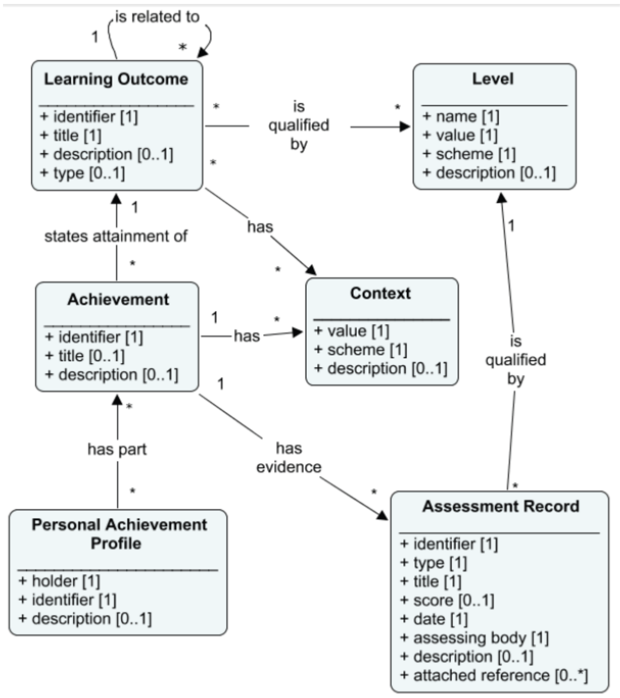

Figure 2.6: The PALO data model

2.3 Educational Platforms

Michael Moore and Greg Kearsley [34] describe Distance Education as:

> *Distance education is teaching and planned learning in which teaching normally occurs in a different place from learning, requiring communication through technologies as well as special institutional organization.*

Teachers and students are in different places for all or most of the time that they teach and learn. Because they are in different places, in order to interact with each other they are dependent on some form of communication technology. This definition is sometimes mistaken for e-learning, but e-learning is a means for distance education by use of electronic means, mostly the Internet. Since we aim to achieve a standard model to describe educational services using semantic web technologies we will focus this section on Online Distance Learning Systems. These online systems differ from the traditional education services mostly in terms of interaction. In the traditional service the interaction is final, meaning that both teacher and learner are face-to-face during the class, while the online systems use technologies, such as videoconference, to simulate the presence of both parties in the class, whilst providing the comfort of the home to students and, most importantly, allowing an unlimited number of students to consume these types of services.

In order to better understand these services we looked into three platforms:

Coursera [4] Is a social entrepreneurship company that partners with the top univer-

[4] https://www.coursera.org/

sities in the world to offer courses online for anyone to take, for free. Coursera has 33 Universities that have partnered with them.

Udacity [5] Was founded by three roboticists who believed much of the educational value of their university classes could be offered online at a very low cost. A few weeks later, over 160000 students in more than 190 countries enrolled in their first class, "Introduction to Artificial Intelligence". These courses are provided by teachers and not by universities, as Coursera.

edX [6] Is a not-for-profit enterprise founded by Harvard University and the Massachusetts Institute of Technology that features learning designed specifically for interactive study via the web. EdX is based in Cambridge, Massachusetts, and is governed by the MIT and Harvard. The courses are provided by 3 universities: Harvard, Berkeley and MIT.

In Table 1.3 we can see the information available for each course on these three systems and the corresponding conclusions.

If we think about the impact of having a standard model describing each educational service associated with these online learning systems, the conclusion is that it should be great, enabling the attainment of some important goals, like having the possibility to compare them by means of establishing a match, having standard descriptions of these services, and even enabling the platforms to allow users to download a specific course by means of a RDF file with their description.

2.4 Methodologies and Processes

In Bizer's work [5] we can establish an analogy with the process of supporting recruitment and the process of finding a specific course. This online recruitment process identified in [5] is shown in Figure 2.7. We can establish the following analogies with our case study:

Job Posting in order to make the job postings available, employers publish their postings in their own website. In our case, universities publish their courses in their own websites or platforms, as we can analyse in Table 1.3. Bizer points out that this paradigm must be changed from a centralised approach to a distributed one. Accordingly, in our case study, we will build a model that can be used in a distributed way.

Semantic Matching this matching can be used to determine the semantic similarity between concepts. Allowing the comparison of job descriptions and applicants' profiles. In our case, we will establish a match between courses that will allow to, for example, create new degrees based on courses from world wide universities, as stated in Chapter 1.

[5] http://www.udacity.com/

[6] https://www.edx.org/

Recommend The end result is a selection of open positions that are recommended to a specific applicant's profile. Again, in our case, we will have a selection, according to the previous matching between educational services or courses, of courses that are recommended to create a new degree.

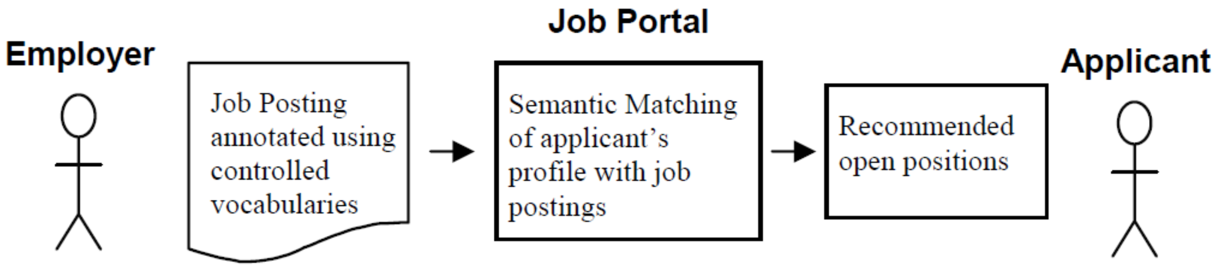

Figure 2.7: Recruitment process using Semantic Web technologies [5]

All this was carried out because Bizer et al. [5] refer that a large number of job portals are rising, given that the number of institutions and online distance education platforms is rising. This is causing information to be scattered, preventing the offers from reaching all qualified applicants, as we could see before, in Table 1.3, were we analysed the available information on several universities' platforms. In our case the information is also scattered, and not structured. In order to improve market transparency, participating employers need to use terms from a controlled vocabulary to categorise their postings using variations of HR-XML[7] data format or, in another way, add semantic annotations to job postings on their corporate websites using terms from a set of controlled vocabularies and the RDF data format. This would allow job portals to crawl these pages and find the exact data through those semantic annotations. Bizer points that, despite these solutions, there is still a problem regarding proofing testimonials of experience and education, because there is no common standard for the description of skills acquired by the learners in universities. This thesis could be the solution to this problem: by describing the educational services available in universities, we will focus on describing the value offered by each university, which is the learning outcomes and skills acquired by the learners. This could be done by intersecting the bloom's taxonomy with an educational area, i.e., the model could be able to determine that an arbitrary course educates a person in a specific area, and that the focus of that course is in a specific category of each dimension of the Bloom's Taxonomy.

Despite these different areas of expertise, we believe that if a schema, or a model, was used to publish the courses or degrees available in an education institution, we could achieve the results mentioned in [5]. For example: calculate the similarity between courses, create European courses through the matching of the courses available in different countries, and facilitate the interoperability and reuse between

[7] http:\hr-xml.org

learning software platforms [1].

3
Educational Service Model

This chapter introduces the methodologies used to created our educational service model along with the implementation of the model itself.

The first section presents an introduction to the education as a service. The second provides an overview about modelling and methodologies used to design the model followed by the existing initiatives about existing models in the education domain. Finally, the fourth section is describes the model created, which was named Linked *USDL4EDU*. In the last chapter is described the integration with the Linked USDL model.

3.1 Education as a Service

Universities and other companies are beginning to deal with the unsustainable growth of education costs. As a result, new solutions are being explored to make higher education more available – and cheaper - for all people. MIT, Stanford, Harvard, New York University, and other prestigious institutions have explored the provision of free online courses (Moocs). The success of these online courses, either through popularity or buzz, has influenced the creation of a new definition - Education as a Service [28].

In [36] is suggested that a marketing orientation could also help universities compete in the global arena by means of integrating research in education and services marketing to produce a framework for the analyses of education as a service. The education industry, particularly higher education institutions, has not had the benefit of the progress of the research in marketing to assist firms competing in a complex world.

Education resembles a service more than a product. The characteristics of a service are that production and consumption occur simultaneously. Yet, education is special, in the sense that sharing is non rival; education has this characteristic in common with information and knowledge. When one teaches something to someone, the knowledge is doubled in the sense that both have the knowledge, whereas in the case of a rival service or product the seller transfers the use of the sold good over to the buyer. And sometimes the process of education increases the knowledge of the teacher throughout the process of educating [13]. In this work, we will focus our perspective of education as a service in the knowledge and skills that are acquired by the learners and not in the knowledge that a teacher can acquire throughout the process of educating someone.

By creating a service model that can describe services in the scope of education as a service implies that universities can access and use standard specifications. This will have strong implications in the universities or organizations such as Moocs share information. An example of a case in which it is important to a certain university to access another's description is, as we already mentioned, when a student needs to establish equivalences between courses when doing an exchange programme. Nowadays, most of the information is centralized in local databases meaning that each university has its own relational (or non relational) models. So, if an university has the need to exchange information about a specific course or degree, the implication is that both universities must understand each other's models. For the reasons presented before, this model could have a huge impact on education, in particular higher education due to the growth of the courses available online, sometimes called *"À la carte"*, i.e. it is possible and is easy to "consume" a course that is on a course catalogue from an open online course, with the learner just having to choose which course he wants to participate. The term *"À la carte"* comes from the fact that these courses are available from a menu, usually called course catalogue, and are becoming very easy to consume and acquire.

Understanding the definition of education as a service is also helpful to the creation of the model, in order to promote better understanding of what is exchanged between both parties involved in this type of services and the role that this model has in the service itself. Given our focus on the service description we decided to place the model between the service provider and the service consumer, implying that the description will serve as an interface that the service consumer will use to better understand the service.

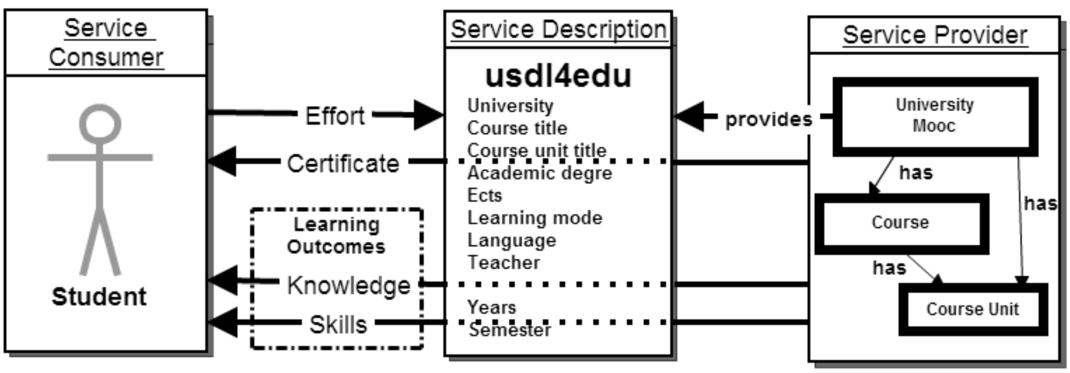

Figure 3.1: Our view of Education as a Service

As we can see from figure 3.1 the service provider will provide a service that can be either a degree or a course unit. With regard to what the student must provide to access the service is, as we called, the effort and in some cases there is also the price to acquire the service, since the price can already be defined by the *Linked USDL* and *Linked USDL-price*. In this project we will ignore the price in the service model. By consuming the service, the consumer acquires the Learning Outcomes by means of *Knowledge* and *Skills*, and also an academic degree or academic certificate that certifies that the learner has received specific education or has passed a test or series of tests.

We will focus our efforts in the description of the learning outcomes and learning objectives, because we believe that the learning objectives represent the essence of the service, thereby providing the background information about a given educational service. As mentioned before, one of the main challenges will be how to describe Learning Outcomes or Learning Objectives. In the best-case scenario these two properties will represent the same, since both aims and objectives are in common language synonymous with goals and they are both suggestive of a form of goal-oriented education. For this reason some educational organisations use the term learning outcome since this term is inclusive of education in which learners strive to achieve goals, but it is more inclusive as it includes other forms of education. For example, in Learning through play children are not made aware of specific goals but, nevertheless planned, beneficial outcomes result from the activity. This way, we choose to represent, in our model, only the Learning Objectives, being the Learning Outcomes inherent from the previous one. Section 3.3 presents a solution to describe educational objectives.

3.2 Modelling

Before we go through the process of creating our educational service model we must first understand what modelling is. In an abstract way, modelling is the process of generating abstract, conceptual, graphical or mathematical models by means of methods, techniques and theories. A model can provide a way to read elements easily that have been broken down to a simpler form since it involves simplification and formalization [22]. Modelling refers to the process of generating a model as a conceptual representation of some phenomenon. Usually, a model will refer only to some aspects. In the case of service description models, it will depend on the information from the viewpoints of the service provider and from the service receiver. This way, two models of the same domain may be essentially different due to differing requirements of the models' end users, or to conceptual or aesthetic differences among the modellers or even to contingent decisions made during the modelling process.

Since we aim to build an ontology to represent our educational model it is important to review the methodologies for building ontologies in order to acquire knowledge in how to evaluate and validate the final ontology, also known as Ontology Engineering. In [32] it is presented the *Methontology*, a framework for specifying ontologies at the knowledge level. This methodology starts from the identification of the ontology's purpose and the need for domain knowledge acquisition, tasks which have been already presented in previous chapters. Considering that most of the knowledge has been acquired, the "*ontologist*" has a substantial amount of unstructured knowledge that must be organized, and thus conceptualization is required to organize and structure the acquired knowledge using external representations that are independent of the implementation language and environments. In this phase, we presented the *Bloom's taxonomy*, an external representation of educational objectives that organized the objectives of a given curricular unit into arranged categories. *Methontology* proposes that evaluation occurs throughout ontology development, mostly in the conceptualization phase. In our case, as noted before, we will evaluate our model by populating the model with two different approaches - manual and automatic - and, in the end, by building an infrastructure that can host the model and its instances and establish relations between the services.

About the evaluation of an ontology, in [25] it is also presented another methodology for the design and evaluation of ontologies that refers that the development of ontologies is motivated by scenarios that are not adequately addressed by existing ontologies. The author also refers that the motivating scenario will raise a set of queries that can be considered to be requirements that are in the form of questions that the ontology must be able to answer, called competency questions. In Section 4.1 we present our competency questions.

Another important process, referred in [32], is the *integration*, in which we had to identify terms that could be included from other ontologies. This analyses is presented in the next chapter.

Finally, the last step referred in the *Methontology* framework, is the *implementation*. In this final step, we used the *TopBraid* [1] composer over the *Protégé* [2]. Protégé is a free, open source ontology editor and knowledge-base framework based on Java, whilst TopBraid is an Eclipse plug-in. Despite the fact that Protégé is the most used software in the creation of ontologies, we choose to use the TopBraid because the interface is more friendly when we want to develop some "out of the box" definitions, such as, using Classes or Properties from external ontologies without having to import them. Overall, after using both technologies, my best evaluation concerns TopBraid.

3.3 Existing Initiatives

As noted in previous chapters, the *integration* phase in the process of creating an ontology allows us to identify terms that could be included from existing ontologies. These terms can either be existing ontologies that were created to describe abstract knowledge, or more elaborate ideas that can describe knowledge in our domain of study. This way, it is important to find and study existing initiatives in the

[1]http://www.topquadrant.com/products/TB_Composer.html
[2]http://protege.stanford.edu/

education domain, since these will be the building blocks to the ontology we aim to build. Accordingly, we divided the identified ontologies into two categories, with the first one called auxiliary ontologies since they will help us describe common properties of our educational services, and the second one called existing initiatives since they will help us understand how to better describe this kind of services.

The auxiliary ontologies are:

GoodRelations Language Reference GoodRelations is a standardized vocabulary (also known as "schema", "data dictionary", or "ontology") for product, price, store, and company data that can (1) be embedded into existing static and dynamic Web pages and that (2) can be processed by other computers. This ontology will be used to describe details of the service, if needed, in a more commercial perspective.

FOAF FOAF (an acronym of Friend of a friend) is a machine-readable ontology describing persons, their activities and their relations to other people and objects. Anyone can use FOAF to describe him- or herself. FOAF allows groups of people to describe social networks without the need for a centralised database. This ontology will describe the people involved in the education process, mostly teachers and the universities as organizations.

Dublin Core The Dublin Core metadata terms are a set of vocabulary terms which can be used to describe resources for the purposes of discovery. This ontology will be used to describe the classes created in our ontology.

SKOS Simple Knowledge Organization System SKOS provides a standard way to represent knowledge organization systems. This ontology will be used to represent our schema about *Education Areas* (more on these topics will be discussed later).

The existing initiatives in terms of building ontologies related to the education domain are [3]:

ECS - School of Electronics and Computer Science, University of Southampton In [11] is described the ECS ontology, an ontology with 88 classes to describe all the available resources in the School of Electronics and Computer Science. This ontology provides a perspective on a description tilted towards the university itself rather than at educational services, as in our case. This ontology was one of the starting points in the conceptualization phase, where we needed to organize the acquired knowledge.

Some of the available classes:

- Organisation
- Group
- Project
- (Theme) ResearchTheme
- Seminar
- Presentation

[3] the reader may notice that some of these ontologies were already presented in previous chapters

- (People)Person
- Role
- Publication
- Document
- Interest
- Location
- Degree
- Cohort
- Modules
- Semester
- Session
- Year of Study

Academic Institution Internal Structure Ontology The Academic Institution Internal Structure Ontology (AIISO) in [51] provides classes and properties to describe the internal organizational structure of an academic institution. AIISO is designed to work in partnership with Participation [4], FOAF [5] and aiiso-roles [6] to describe the roles that people play within an institution.

Available classes:

- Center
- College
- Course
- Department
- Division
- Faculty
- Institute
- KnowledgeGrouping
- Module
- Programme
- ResearchGroup
- School
- Subject
- OrganizationalUnit

The Bowlogna Ontology This ontology [15] aims at providing a standard schema for European universities involved in the Bologna Reform of higher-education studies.

Available classes:

- AcademicDegree
- Certificate
- ECTS-credits
- LearningOutcomes
- Field-of-studies
- Curriculum
- Person
- LearningActivity
- Department
- Evaluation
- Thesis
- Semester
- Student
- Module

Teaching Core Vocabulary Specification TEACH [29], the Teaching Core Vocabulary, is a lightweight vocabulary providing terms to enable teachers to relate things in their courses together.

Available classes:

[4] http://purl.org/vocab/participation/schema
[5] http://xmlns.com/foaf/0.1/
[6] http://purl.org/vocab/aiiso-roles/schema

- Course
- Assignment
- Material
- Teacher
- StudentGoup
- Student
- Module
- StudyProgram
- Lecture
- Building
- Room

Mathematics Subject Classification The Mathematics Subject Classification Linked Wiki [7] presents the effort of implementing the Mathematics Subject Classification (MSC2010) as a Linked Open Dataset using SKOS - Simple Knowledge Organization System. This is more a dataset than an ontology, because it uses the SKOS ontology to represent a dataset of Mathematics subject classification, i.e. educational areas related to Mathematics, and it is important because it is a mirror of the schema we built to describe the ACM Computing Classification System [8] to represent educational areas, more information on this schema in the following chapters.

All these ontologies were important in the construction of our model, some by being inherit in classes of our model and others being the building blocks, or the starting points, in our model, providing knowledge in the education and universities domain. By looking at the most common classes used in these ontologies we can understand if a given class is important to be considered for the model or not.

The *Bowlogna Ontology* is the only ontology (from the ones we studied) that were comprehensive with the Learning Outcomes by creating knowledge in their model about this field. The Bowlogna ontology defines Learning Outcomes as:

> "Declaration stating the knowledge and skills acquired by a student during a study period."

However, this concern over the learning outcomes or objectives is not fully present in the ontology, because there is not a Class to describe objectives. The existing Class about learning outcomes is only related to the Curriculum class, meaning that in this ontology there is no direct relation between a course or course unit and its learning outcomes which, in our point of view, is an important property of an educational unit. Nevertheless, the definition provided by the ontology led us to believe that the Bloom's taxonomy is a good solution to describe this important property of an educational service, because the definition says: "*Knowledge* and *Skill* acquired by a student" and the Bloom's taxonomy is divided into *Knowledge*, *Cognitive* and *Skill*.

[7] http://sci-class.math.auth.gr/MSCLW/index.php/Main_Page
[8] http://dl.acm.org/ccs_flat.cfm

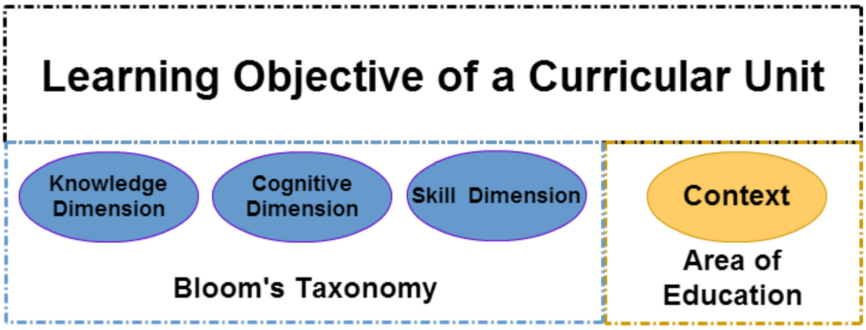

Figure 3.2: Intersection of Bloom's taxonomy with an educational context to describe educational objectives

However, in Chapter 2 (State-of-the-Art), we presented a model capable of describing Personal Achieved Learning Outcomes where the author relates the Learning Outcome with a context, i.e., an educational area that "gives meaning to the learning outcomes through a set of factors" [35]. Therefore, we believe that intersecting the Bloom's taxonomy with an educational context/area will better describe the learning objectives of an educational service. This is shown in Figure 3.2.

3.4 The Model and its implementation

Before we started building the model using the *TopBraid* software, some sketches were developed through a number of iterations and discussions with experts, leading to the final model presented in Figure 3.3. We decided to name this educational service model: *Linked USDL4EDU*. The main Classes of this model are: *Educational Service*, *Degree* and *Course Unit* because they represent the top level of the educational service, i.e. the service consists of a Degree or a Course Unit. This way it is possible to represent either a degree or a single unit. Single units could be, for example, courses from Moocs. Then, a Degree consists in a set of course units. The *Course Unit* Class will then have an overall objective and an overall prerequisite. The prerequisite could be as simple as an area of education (represented using the *skos:Concept* Class) or could be another service, e.g. if we have a Master degree that has a Bachelor degree as its mandatory prerequisite, then this prerequisite is attainable by this relation.

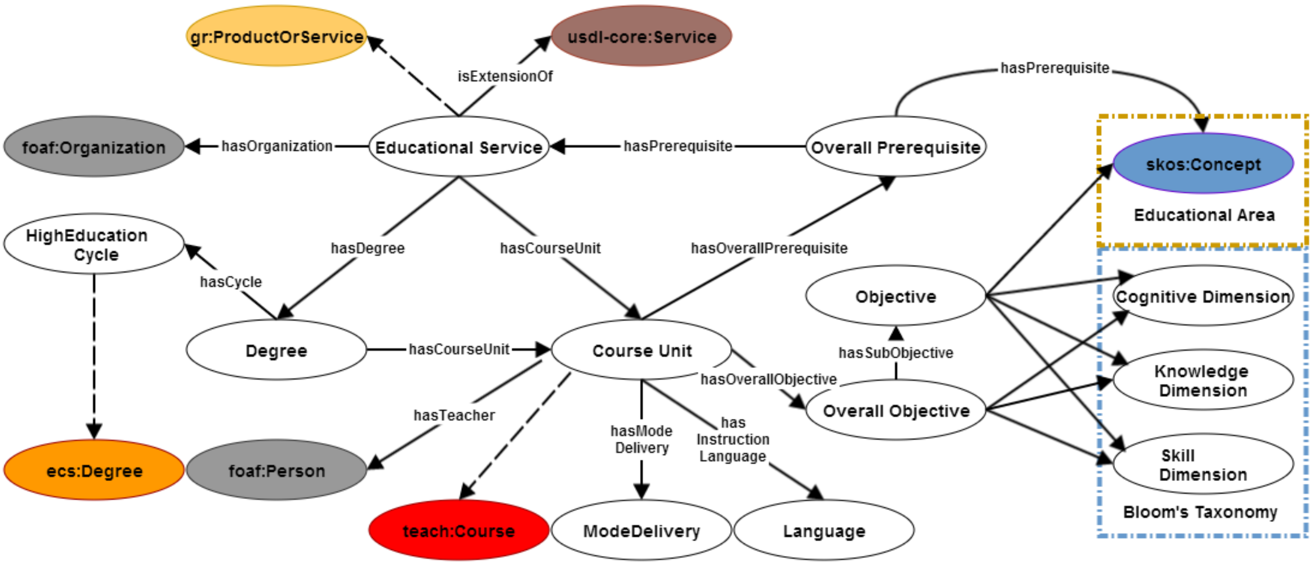

Figure 3.3: Educational Service Model - *Linked USDL4EDU*

Some changes were made during the evaluation phase while instantiating the model. In early sketches the classes *OverallObjective* and *OverallPrerequisite* did not exist, but when instantiating the model in an automated way we decided to include these classes between the *CourseUnit* and the *Prerequisite* and *Objective* in order to have a class that can describe the summary of all objectives and prerequisites and the average of the values of each objective according to the Bloom's taxonomy categories. With these classes, we can describe each objective with its values from the Bloom's taxonomy and the context in which that objective is inserted. With those values we can calculate the average of the entire curricular unit, i.e. the trend of the curricular unit in terms of the Bloom's taxonomy values and store that value in the *OverallObjective* Class.

Another challenge that was faced consisted in converting the Bloom's taxonomy into an ontology. We decided to create one Class for each dimension: *Cognitive*, *Knowledge* and *Skill* dimensions; and each category of each dimension is created by instantiating the specific Class. Listing 3.1 shows the description of the Class *CognitiveDimension* and Listing 3.3 shows an instance of the *CognitiveDimension* class, which represents the *Remember* category. This relation is present by the *"rdf:type :CognitiveDimension"*, which defines that the element *CognitiveDimension_1Remember* is an instance of *CognitiveDimension*. These classes serve also as an example of all the other implemented classes. In the *label* property is the text we want to show about this category; the description property has a small description that helps to understand what this category is about; and finally the value, being the smallest the lower thinking skills and the highest the high order thinking skills. This value is important to establish a numeric value to each category that will help to apply mathematical terms in order to be able to compare courses and to visualize them as graphs, for example.

In the *hasKeyword* literal property is the list of verbs that are representative of that dimension according to what was already shown in Table 2.1 and Table 2.2, which shows each category of the available dimensions of the Bloom's taxonomy. In the evaluation phase of the model, more specifically when we were extracting data using web scrapers, we notice that this property (*hasKeyword*) containing the list of verbs could also be in a class, simplifying the modification of these verbs, since these verbs were very important to identify which category was present in a given curricular objective. By creating a Class to represent these verbs it would have been much easier to define which verb belongs to each category. At this point we decided to do not change the ontology, because many work had been done and this small change would imply to rebuild a lot of developed code.

Listing 3.1: Cognitive dimension

```
:CognitiveDimension
    rdf:type rdfs:Class ;
    rdfs:comment "This class will be representing the Cognitive dimension of Bloom's Taxonomy for one educational objectives of a curricular unit"^^xsd:string ;
    rdfs:label "Cognitive Dimension"^^xsd:string .
```

Listing 3.2: hasKeyword Property

```
:hasKeyword
    rdf:type rdf:Property ;
    rdfs:domain :SkillDimension , :KnowledgeDimension , :CognitiveDimension ;
    rdfs:label "has Key Word"^^xsd:string ;
    rdfs:range xsd:string .
```

Listing 3.3: Cognitive dimension - Remember

```
:CognitiveDimension_1Remember
    rdf:type :CognitiveDimension ;
    rdfs:comment "Remember category of Cognitive Dimension"^^xsd:string ;
    rdfs:label "Remember category"^^xsd:string ;
    dc:description "Retrieve relevant knowledge from long-term memory"^^xsd:string ;
    :hasKeyword "recall"^^xsd:string , "describe"^^xsd:string , "list"^^xsd:string , "identify"^^xsd:string , "know"^^xsd:string , "reproduce"^^xsd:string , "match"^^xsd:string , "recognize"^^xsd:string , "outline"^^xsd:string , "select"^^xsd:string , "tabulate"^^xsd:string , "label"^^xsd:string , "remember"^^xsd:string , "state"^^xsd:string , "name"^^xsd:string , "define"^^xsd:string ;
    :hasValue 1 .
```

As the reader will notice, these verbs will be very important in future chapters once they will be the foundation to the identification of these categories in the automated population of the service model. The verbs that were included in each category of each dimension were the following:

Cognitive Dimension					
1-Remember	2-Understand	3-Apply	4-Analyse	5-Evaluate	6-Create
define describe identify know label list match name outline recall recognize remember reproduce select state tabulate	comprehend convert defend distinguish estimate execute extend generalize infer interpret paraphrase predict rewrite summarize translate understand	apply calculate change classify compute construct demonstrate discover experiment manipulate modify operate prepare produce relate show solve use	achieve analyse compare contrast deconstruct diagram differentiate discriminate distinguish explain illustrate infer order outline relate select separate	appraise assess compare conclude contrast criticize critique defend discriminate evaluate justify rank reflect relate support	categorize combine compile compose create design devise generate modify organize plan rearrange reconstruct relate reorganize revise rewrite tell write

Table 3.1: Verbs used to establish the linkage to the Cognitive dimension of Bloom's taxonomy

Knowledge Dimension			
1-Factual	2-Conceptual	3-Procedural	4-Meta-Cognitive
list summarize classify order rank combine	describe interpret experiment explain assess plan	tabulate predict calculate differentiate conclude compose	identify execute construct achieve reflect create

Table 3.2: Verbs used to establish the linkage to the Knowledge dimension of Bloom's taxonomy

With this set of verbs we do not want to establish that only these verbs are correct, because many more synonyms of these verbs could be added to the model and perhaps it would be important to define verbs to different areas of education. However, and according to the literature, we decided to use such verbs because they are the main verbs to express each category.

Another important detail about the model is the namespaces, which identifies each ontology used by means of setting a prefix of each of those ontologies and a given URI, the namespace does not necessarily point to a real URL. In our case the namespace of our ontology is `http://rdf.genssiz.dei.uc.pt/usdl4edu#` which is a real URL that points to a real web page that uses Neologism to describe the model. Neologism is a lightweight web-based vocabulary editor and publishing tool built with Drupal, which is an open source content management platform powering millions of websites and applications. It is built, used, and supported by an active and diverse community of people around the world. This way, the description of each class and property of the model is available to everyone that is interested in understanding the model in a public website, as we can see in Figure 3.4.

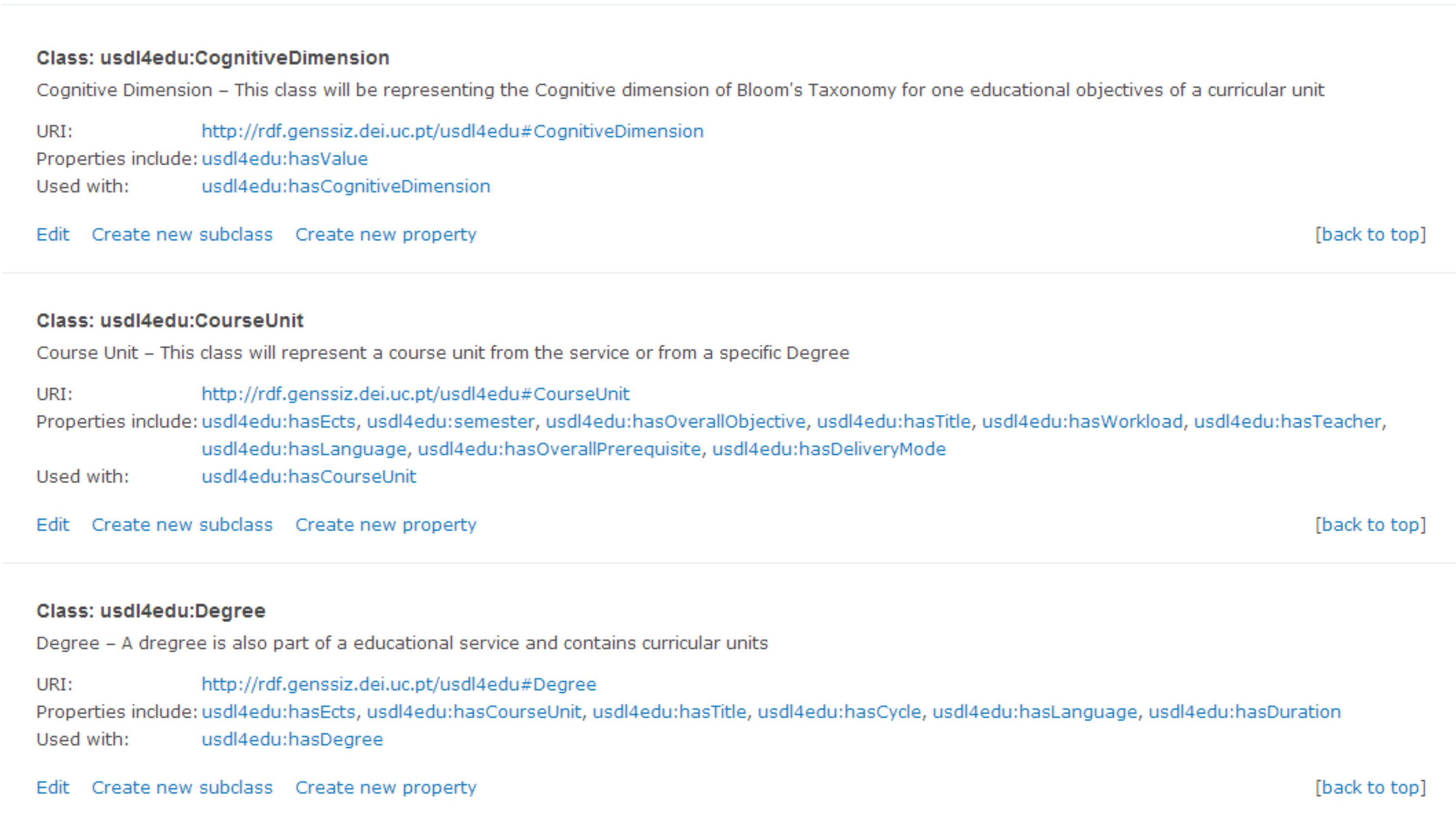

Figure 3.4: Information about the model in a public web page

3.5 Integration with Linked USDL

Another important requisite for our model is to be integrated with Linked USDL. We had two options: we could either define our main class (*EducationalService*) as *subClass* of the Linked USDL *Service* class or we could define a new property that would create a relation between the two models. Regarding the first option, we decided that it was not a good solution because our main class would inherit all the properties of *Service* class from Linked USDL, and most of those properties do not fit this type of service, and could thereby result in redundant information. Accordingly, we defined a new property *isExtensionOf* - this is visible in Section 3.4 - that would define that a given instance of *Linked USDL4EDU* model is an extension of a given service defined in Linked USDL, enabling, this way, to define, for example, the price of the service and the location by instantiating the Linked USDL.

4 Model Evaluation

This chapter presents two approaches of evaluation that were developed to demonstrate the applicability of the model to real world usage.

The first section describes a set of questions to which the model should answer to, providing some real world context into the model. In the second section is provided the developed manual population of the educational service model created previously followed by the automated population in order to evaluate the model. This chapter ends with some statistical results about the usage of the verbs that better describe each category of the Bloom's taxonomy.

4.1 Competency Questions

In section 3.2 we presented to the reader the competency questions and the role they have in the design and evaluation of ontologies, according to [25]. We divided our competency questions into the roles or situations that people with questions may assume, and we gave a real context to these questions to better prove the impact the *Linked USDL4EDU* model can have on society. These roles are: European Union deputy, Ministry of Education, university dean, professor and student. By providing these competency questions concerning different positions we are able to give value to the ontology itself. These questions can have an answer by querying directly the ontology or by using the tool/application developed in the next chapter.

Question 1 - European Union deputy Due to the European crisis it is important encourage the countries most affected by it to train their population to be able to innovate or create new ideas and new solutions. Therefore, it is important for an European Union delegate to check the *Cognitive* values of the Bloom's taxonomy of the courses of a given country or set of countries, i.e. it is possible to check if the courses are moving towards to the *Create* category of the Cognitive dimension instead of the lower levels of this dimension, i.e. *Understand and Remember.*

Question 2 - Ministry of Education With the unemployment rate increasing to numbers never seen before, the Ministry of Education now starts to observe that it can be important to merge or extinguish courses that have low employment rates. The courses to be merged can be similar courses that are not far apart in their location. The Linked *USDL4EDU* model can provide such evidence by comparing the values of the dimensions of the Bloom's taxonomy and comparing the educational areas of each course. That is, if two given courses are near according to their location, share educational areas (e.g. both from Informatics Engineering) and do not have a considerable distance in terms of Bloom's taxonomy dimension, then it is likely that both courses are teaching exactly the same. Accordingly, these courses can be merged to one single course to avoid the excessive costs of maintaining both courses.

Question 3 - University dean An university dean can be interested, for example, in standardizing all the engineering courses of the university. This can be done by checking the Bloom's taxonomy values of the engineering courses in order to identify if there are courses that are more practical than others. This can be important, for example, for the admission of students by the engineering's order. The dean can have an histogram that displays the Bloom's taxonomy values of each engineering course in his university to check if the courses are correctly distributed and standardized.

Question 4 - Professor A professor can describe its own course unit and check if the values of the dimensions of the Bloom's taxonomy are according to his judgement as well as if the curricular unit is evolving along the years.

Question 5 - Student The student can use the model as an auxiliary framework to help him choose the units that better fit his needs from a degree's curricular plan, that is, the student can have an informed judgement on which course units better fit his expectations in terms of knowledge acquisition.

With these competency questions we do not want to generate ontological commitments; they are used instead to evaluate the ontological commitments that have been made by evaluating the expressiveness of the ontology. As we will see in the next chapter, we can verify that the model provides an answer to these questions through the tool/application developed to support the model itself.

4.2 Manual Population

We decided to start with a manual population of the model in order to get a faster evaluation of the model, i.e. by creating a small number of instances of the model we have a new perspective of what is or it is not important to exist in the model, and the things that could be improved. As we already mentioned one of the major changes in this step was the introduction of an *overralObjective* to describe the objectives of a given curricular unit as a whole.

In this step we decided to start with a curricular unit from our degree, i.e. a course unit that we already knew, which provides us a better perspective about that curricular unit. In order to create these manual instances we had access to all curricular units' records from the Computer Science Master degree from our university.

In this phase of the evaluation of the model, we managed to identify the categories of each dimension of the Bloom's taxonomy by the verbs inserted in the *Linked USDL4EDU* service model, which helped us in the following step, the automated population. We also noticed that it was important to have an hierarchy of educational areas already instantiated in order to understand what is important to define as a *Context* (area of education) to a given objective. In this phase, we instantiated some educational areas that we used, but since there was no hierarchy amongst them we believe that they do not have a good expressiveness. In the following listings it is shown an extract of the RDF code created to instantiate this curricular unit. The complete RDF code can be found in the `https://github.com/jduro/esoteric/blob/master/manual%20population/occi.ttl` file.

Listing 4.1: Organizational, Behaviour, Knowledge and Innovation: Educational Service

```
:service_occi a usdl4edu:EducationalService;
  gr:name "Organizational, Behaviour, Knowledge and Innovation"^^xsd:string;
  rdfs:label "Organizational, Behaviour, Knowledge and Innovation Service"^^xsd:string;
  rdf:comment "Service about Organizational, Behaviour, Knowledge and Innovation, a Curricular unit from Master's Degree in Computer Science from University of Coimbra";
  usdl4edu:hasCourseUnit :unit_occi;
  usdl4edu:hasOrganization :university_coimbra.

:university_coimbra a foaf:Organization;
  dc:description "FOAF of University of Coimbra";
  foaf:member :jcardoso;
  foaf:name "University of Coimbra";
  foaf:homepage <http://www.uc.pt/> .

:jcardoso a foaf:Person;
  foaf:name "Jorge Cardoso";
  foaf:homepage <http://eden.dei.uc.pt/~jcardoso/> .
```

Listing 4.2: Organizational Behaviour Knowledge and Innovation: Unit

```
:unit_occi a usdl4edu:CourseUnit;
  dc:description "Organizational, Behaviour, Knowledge and Innovation from Master's Degree in Computer Science from University of Coimbra"^^xsd:string;
  rdfs:label "Organizational, Behaviour, Knowledge and Innovation"^^xsd:string;
  usdl4edu:hasEcts "6"^^xsd:integer;
  usdl4edu:semester "1"^^xsd:integer;
  usdl4edu:title "Organizational, Behaviour, Knowledge and Innovation"^^xsd:string;
  usdl4edu:hasWorkload "8"^^xsd:integer;
```

```
  usdl4edu:hasOverallObjective :overallobjectiveOCCI;
  usdl4edu:hasTeacher usdl4edu:jcardoso;
  usdl4edu:hasLanguage usdl4edu:LanguagePT;
  usdl4edu:hasLanguage usdl4edu:LanguageEN;
  usdl4edu:hasDeliveryMode usdl4edu:ModeDeliveryPres.

:LanguageEN a usdl4edu:Language ;
     rdfs:label "English"^^xsd:string .

:LanguagePT a usdl4edu:Language ;
     rdfs:label "Portuguese"^^xsd:string .

:ModeDeliveryPres a usdl4edu:ModeDelivery ;
     rdfs:label "Presential"^^xsd:string .
```

Listing 4.3: Organizational Behaviour Knowledge and Innovation: OverallObjective

```
:overallobjectiveOCCI a usdl4edu:OverallObjective;
  dc:description "The objective of this course is to help students
      to understand how people and organizations operate as a complex
       system. The course covers individual behavior in organizations
      , group behavior, and how behavior affects the overall
      performance of organizations. The course also provides an
      overview of innovation methods. Covered topics include
      motivation, job satisfaction, communication, leadership, change
      , and organizational culture. It also provides the necessary
      competencies in synthesis, written communication, problem
      solving, critical reasoning, practical application of knowledge
       in managing organizations and assists students in developing
      skills of analysis and diagnosis in the organizational context.
       "^^xsd:string;
  rdfs:label "Objectives of Organizational, Behaviour, Knowledge and
       Innovation from UC"^^xsd:string;
  usdl4edu:hasPartObjective :objectiveOCCI1;
  usdl4edu:hasPartObjective :objectiveOCCI2;
  usdl4edu:hasPartObjective :objectiveOCCI3;
  usdl4edu:hasCognitiveDimension usdl4edu:
      CognitiveDimension_5Evaluate;
  usdl4edu:hasKnowledgeDimension usdl4edu:
      KnowledgeDimension_2Conceputal;
  usdl4edu:hasSkillDimension usdl4edu:SkillDimension_2Basic.
```

Listing 4.4: Organizational Behaviour Knowledge and Innovation: Objective

```
:objectiveOCCI1 a usdl4edu:Objective;
  dc:description "The objective of this course is to help students
      to understand how people and organizations operate as a complex
       system"^^xsd:string;
  rdfs:label "Curricular Objective"^^xsd:string;
  usdl4edu:hasCognitiveDimension usdl4edu:
      CognitiveDimension_2Understand;
```

```
usdl4edu:hasKnowledgeDimension usdl4edu:
    KnowledgeDimension_3Procedural;
usdl4edu:hasSkillDimension usdl4edu:SkillDimension_1Recognition;
usdl4edu:hasContext usdl4edu:contextSocial;
usdl4edu:hasContext usdl4edu:contextSociology.
```

This example clearly shows the role of the *overallObjective* class. In this class, we have information about each identified *Objective*, in this case *:objectiveOCCI1*, *:objectiveOCCI2* and *:objectiveOCCI3*, where each one has its own categories of Bloom's taxonomy, and we also have information about these Bloom's taxonomy values regarding to the trend of the unit itself. In this case we can observe that this unit is trending to the category of *Evaluate* in the cognitive dimension, *Conceptual* in the knowledge dimension, and *Basic* in the skills dimension.

After this instantiation we did two more: Systems Integration and Information Systems Management from the same degree as the first one - the full RDF code of these courses can be found in `https://github.com/jduro/esoteric/tree/master/manual%20population`. At the end, we were satisfied with the results and with the model itself because we were able to fulfil all the created classes of our service model just by looking at the curricular unit record, which shows positive signs towards real word usage.

4.3 Automated Population

Before we stared to develop a tool to populate the model we first had to know where educational services could be found. Some of this work had already been done, mostly when we studied how structured data is in some universities platforms and online educational platforms(Table 1.3). In general, we can find educational data about degrees, modules, teachers, curricular objectives, etc. in universities' platforms (such as the ones studied before), Moocs, ontologies, databases, etc. Unfortunately, we do not have access to all these infrastructures and data sources. In this step we had to make decisions in terms of which platforms we could use to extract data from, and which technologies we could use to better extract the data. At this point we tried to get access to the database behind the University of Coimbra in order to access all the available information, but this option was declined by the database managers. In terms of ontology data, we considered to use the Web API (a specification of how some software components should interact with each other, available through HTTP protocol) provided by the Department of Electronics and Computer Science from the University of Southampton, which returns data using as schema the ontology we already presented in previous chapters (ECS Ontology). However, this data was mostly about research (people, research groups, papers, among others), instead of degrees and curricular objectives.

So we decided to use a scraper, which is a computer software technique for extracting information from websites, namely from Udacity (23 online courses) , Edx (26 online courses), Coursera (369 online courses), and the University of Coimbra student platform (NONIO). We also extracted data from the ACM (Association for Computing Machinery) web site in order to get the available educational areas that belong to the ACM Computing Classification System. NONIO is the platform to support students at the University of Coimbra, carrying all existing information on each Unit and Degree available in the entire university. In the public platform of the university, the information available is very limited when compared to the NONIO platform, which is only accessible with credentials from the university. This demonstrates that the University of Coimbra is not much concerned with describing and showing their own degrees to public users.

By extracting data from these sources, we obtained data from the most popular online platforms and from the University of Coimbra. It was our intention to extract data from more other sources, mostly from other universities, in order to achieve a more scattered data. Since each scraper takes a long time to develop because it involves some study to understand the patterns in each website, we have only developed scrappers to the sources indicated above. In this chapter, we will explain in detail the challenges and the solutions that were found in the scraper developed to each website.

We decided to use Scrapy to extract the data because it is a framework that is easy to use, it is extensively documented, and has a comprehensive test suite with very good code coverage and community, having 850 questions on StackOverflow [1], which helped a lot in the development of each scraper. Scrapy is a fast high-level screen scraping and web crawling framework written in *Python*, used to crawl websites and extract structured data from their pages. It can be used for a wide range of purposes, from data mining to monitoring and automated testing and, of course, web scraping. The following diagram shows an overview of the Scrapy architecture with its components and an outline of the data flow that takes place inside the system. The key elements in this architecture, in terms of what needs to be done to develop a scraper are:

Spiders Spiders are custom classes written by Scrapy users to parse responses and extract items (also known as scraped items) from them or additional URLs (requests) to follow. Scrapy provides a library called *HtmlXPathSelector* that allows users to extract the data from the HTML using *XPATH*, a standard query language that can be used to select arbitrary parts of HTML and XML documents.

Items Item objects are simple containers used to collect the scraped data. They provide a dictionary-like API with a convenient syntax for declaring their available fields. An example of an item is shown in Listing 4.5.

Item Pipeline The Item Pipeline is responsible for processing the items once they have been extracted (or scraped) by the spiders. Typical tasks include cleansing, validation and persistence (like storing the item in a database). In our case this element will parse the content of the *Item* to the ontology files.

[1] Stack Overflow is a question and answer site for professional and enthusiast programmers.

Listing 4.5: Example of Scrapy Item

```
class EducationalServiceItem(Item):
    title = Field()
    objectives = Field()
    prereq = Field()
    summary = Field()
    teachers = Field()
    url = Field()
    code = Field()
```

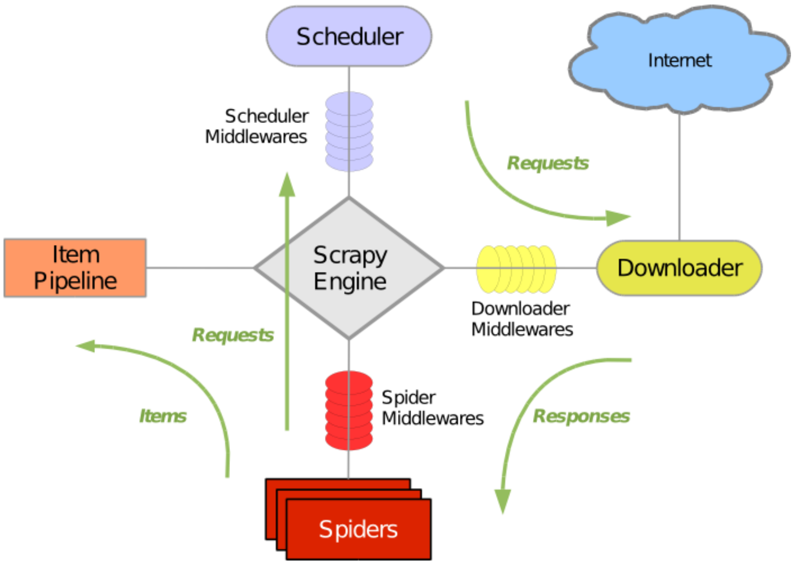

Figure 4.1: Scrapy architecture

The Scrapy web page [2] has information available about how to install Scrapy. In our case, since we work in a Windows OS, it was necessary to install *easy_install*, which is a python module that lets you automatically download, build, install, and manage Python packages.

In total we built 5 different scrapers to extract data from the 5 different sources. In all these scrapers we used the Google Chrome Developer Tools to allow us to understand the patterns and the key tags in the web pages, in order to be able to access the data. This tool, provided with the Google Chrome browser, allows users to navigate through the code of each web site by means of a user friendly interface.

This process of scraping web pages produces instances of our *Linked USDL4EDU* model, i.e. ontology files. Thus, in order to be able to parse the scraped items into these ontology files we had to use one extra *Python* library. After some analyses about existing *Python* libraries (in order to find one compatible with Scrapy) to manage ontologies, we choose to use *RDFLib* [52], a pure *Python* package to work with RDF. This library makes use of a *Graph* interface to load the content of an

[2] http://scrapy.org/

ontology file and is able to serialize, i.e. save, to *RDF/XML*, *N3*, *Triples*, *N-QUADS*, *Turtle*, etc., which are syntaxes for expressing data in the Resource Description Framework (RDF) data model. *RDF* is a specification for conceptual description or modelling of information that is implemented in web resources. We decided to serialize the files using the *Turtle* syntax since it is more Human-friendly, i.e. it is easier to read and understand its content.

All the following scrapers are freely available in a public repository, in `https://github.com/jduro/esoteric/tree/master/scrapy_project`.

4.3.1 ACM Scrapper

The first scraper to be created was the ACM (Association for Computing Machinery) scraper, which extracted all the available education areas from the ACM Computing Classification System website [19]. We decided to start with ACM scraper because we needed to have this instantiated in order to establish educational areas to the future services we would instantiate. This classification system is represented through an hierarchy of educational areas. Accordingly, we extracted the available areas into an *SKOS* ontology, in which each is represented by the *Concept* Class of *SKOS*. A SKOS concept can be viewed as an idea or notion; a unit of thought. However, what constitutes a unit of thought is subjective, and this definition is meant to be suggestive, rather than restrictive.

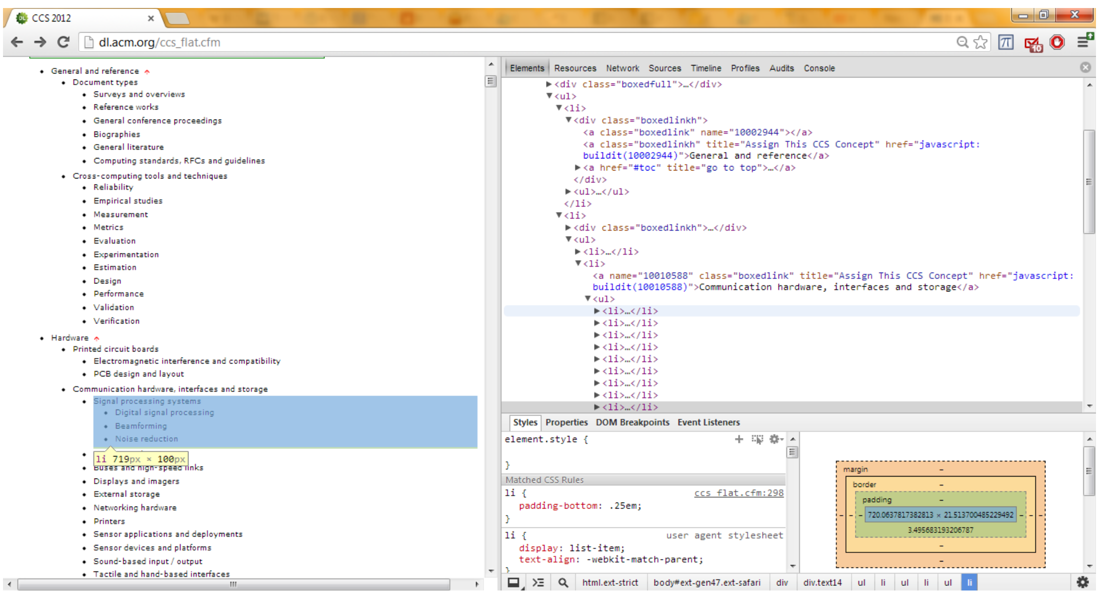

Figure 4.2: ACM Computing Classification website

The page to be extracted is shown in Figure 4.2. As we can see, there is no fixed number of educational areas within a broader educational area. So, in order to be able to create the spider that could go through an unknown level of depth, we have created a recursive function that extracts all the existing areas and sub areas until no more levels are found - this is shown in Listing 4.6. After we selected the data, it was stored in the created Items to be parsed in the Item pipeline. In Listing 4.6 we can see how easy it is to build a spider using *Scrapy*. We simply have to define the starting URLs, and in our case, it is only one page that needs to be scraped. Subsequently, in the *parse* function, we use the *XPATH* library to select the elements (*tags*) in HTML that contain the data we want to use.

Listing 4.6: ACM Scrapy Spider

```
class ACMSpider(BaseSpider):
    name="acm"
    start_urls=["http://dl.acm.org/ccs_flat.cfm"]

    def parse(self, response):
        hxs = HtmlXPathSelector(response)
        lis=hxs.select("./body/div/div/ul/li")
        items=[]
        items=getContext(lis)
        return items

  def getContext(lis):
    items=[]
    for li in lis:
      item = ContextItem()
      item['subContext'] = []
      item['title'] = li.select('./div/a/text()').extract()
      if len(item['title'])==0:
        item['title'] = li.select('./a/text()').extract()
      ul=li.select('./ul/*')
      if len(ul)!=0 :
        item['subContext']=getContext(ul)
      items.append(item)
    return items
```

Listing 4.7: ACM Scrapy Item

```
class ContextItem(Item):
    title = Field()
    subContext = Field()
```

As we can see from Listing 4.7, the ACM item only had two fields: one to store the title of the educational area, and another that will store the sub-areas of that educational area as an *Array* of *ContextItem*.

Afterwards, in the Item pipeline we use the *RDFLib* package to parse all the scraped items into the *Graph* using the *SKOS* namespace and the *skot:Concept* Class.

In *automated population/context.ttl* is the full RDF code, which hosts the educational areas that will be used in future scrapers. This file has 12320 lines and 2467 instances of *skos:Concept*, i.e. 2467 educational areas were scraped.

4.3.2 Udacity Scraper

In developing this scraper we faced a new challenge that was solved by getting to know better the Scrapy framework. In this scraper, we needed to access the course catalogue website of Udacity to extract all the existing courses, as shown in Figure 4.3. Whilst in the previous scraper we only scraped a single page, in the new one we needed to find a way to redirect it to the existing courses, i.e. we needed to create new requests to be parsed later.

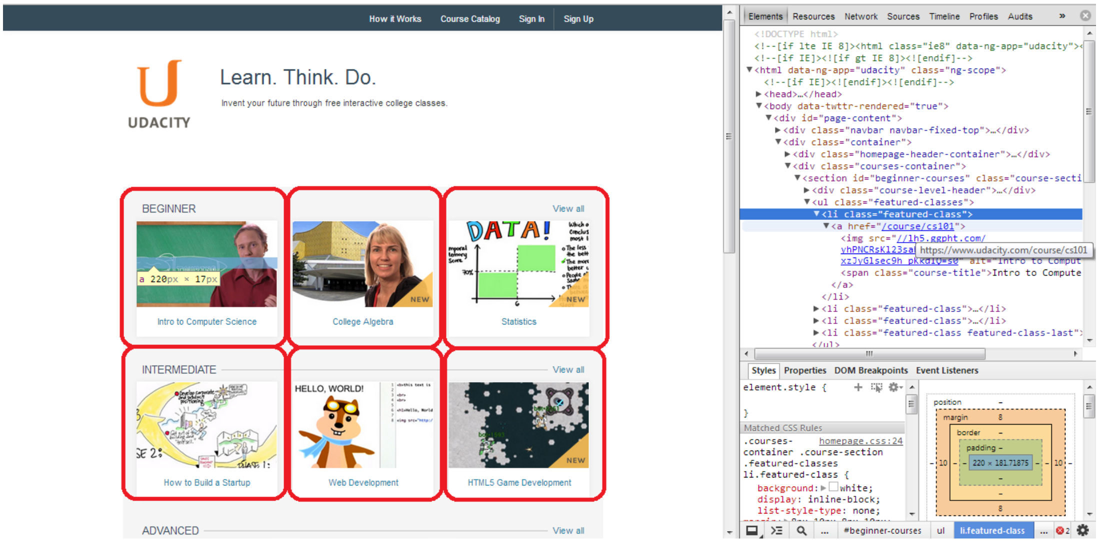

Figure 4.3: Udacity course catalogue website

As we can see from the code present in Listing 4.8, this task becomes very easy to accomplish after we understand how Scrapy works. Basically, we define the allowed domains to let Scrapy know that we do not want to scrap nothing outside those domains, e.g. advertisement, then we define the starting URL has we did before and, then, we use the *SgmLinkExtractor* class provided by Scrapy, also known as *Link Extractors*. Link extractors are objects whose only purpose is to extract links from web pages that will be eventually followed. We noticed that each course's URL is *www.udacity.com/course/courseID*. Accordingly, we use a *rule* that extracts all links from the course catalogue website with this syntax and will redirect those links to the *parse_item* function, which will be responsible to extract the data from each course found.

Listing 4.8: Udacity Scrapy Spider

```
class UdacitySpider(CrawlSpider):
    name="udacity"
    allowed_domains=["udacity.com"]
    start_urls=["https://www.udacity.com/courses"]
    rules = (
        Rule(
            SgmlLinkExtractor(
```

```
                allow=(r'/course/\w+'),
            ),
            callback='parse_item',
            ),
        )
```

Then we had to, once more, understand and discover the patterns in the course web page so that the scraper extracted the correct information for all courses.

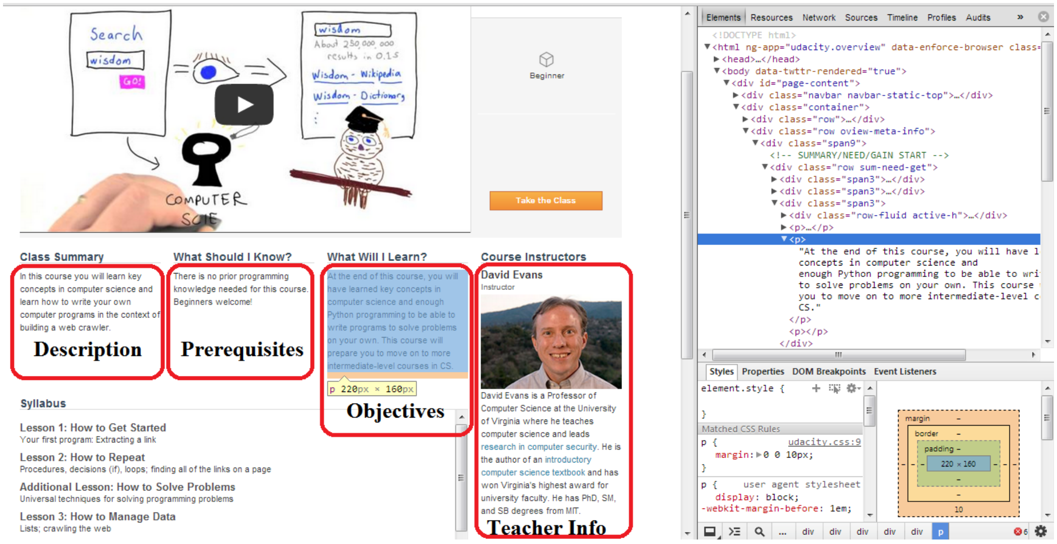

Figure 4.4: Udacity course website

In the Item Pipeline we faced a new challenge, as we had the curricular objectives of each course in plain text and we needed to apply Natural Language Processing, i.e. we needed to enable Scrapy to derive meaning from human or natural language input present in the objectives of each course. Applying natural language processing algorithms is not an easy task, so in an initial iteration we developed a function that would go through the keywords available in each category of the Bloom's taxonomy and would try to find those verbs in the objectives' text.

Accessing the verbs that identify each category of the Bloom's taxonomy was easy, because as we already mentioned, in our educational model (*Linked USDL4EDU*) we added a field that contained this list of verbs for each category. The Cognitive Dimension and Knowledge dimension are easily identified by verbs, whilst the skill dimension does not provide a list of verbs. According to the skill dimension of the Bloom's taxonomy, we could not extract this dimension from the text, since it does not have a list of verbs, unlike the other dimensions.

For each verb, from those that described a category of the taxonomy, that was found in the text it was assumed that a new category of the taxonomy was present in that curricular objective. As an example, in the following curricular objective:

> *At the end of this course, you will have learned key concepts in computer science and enough Python programming to be able to **write** programs*

> *to **solve** problems on your own. This course will **prepare** you to move on to more intermediate-level courses in CS.*

For each verb found we created a new instance of the *Objective* class, that would represent a sub-objective within the *OverallObjective* class. In this case we would have:

- "At the end of this course, you will have learned key concepts in computer science and enough Python programming to be able to write programs to **solve** problems on your own." - **Apply category of Cognitive Dimension**.

- "At the end of this course, you will have learned key concepts in computer science and enough Python programming to be able to **write** programs to solve problems on your own." - **Understand category of Cognitive Dimension**.

- "This course will **prepare** you to move on to more intermediate-level courses in CS." - **Apply category of Cognitive Dimension**.

For each *Objective* found we went through the available educational areas and tried to find the ones that would be more similar, through a simple word comparison procedure. In terms of educational areas, we decided that a given educational area is present in a given objective when there are words in the text that match the title of that area. Once again, this approach is far from optimal, but we must take into account that this was not central to the model itself, since the purpose of these algorithms was to evaluate the model.

We knew that this "algorithm" was not enough, because it would only work when both verbs were in the same form, so we decided to redevelop the algorithm to better identify verbs.

In one of the first steps we decided to use WordNet, a lexical database of English language and *Onto.PT*, which is similar to WordNet but applied to the Portuguese language. We analysed the ontology behind *Onto.PT* and we decided to use it to allow Scrapy to understand each word in the objectives text, and also to use synonyms, in order to increase the scope of the verbs found. Unfortunately, the ontology file was too big, with 160.000 lexical forms. Scrapy became too slow and it was impossible to use this ontology, and in the Item pipeline it was not possible to do new requests to the WordNet API. So we decided to try a new approach.

We verified that the previous algorithm just did not work when the verb form was not the same. After some research we found a technique called "Stemming", which is the process of reducing inflected or derived words to their stem, base or root form. A stem is a part of a word, which does not need to be identical to the morphological root of the word. In 1980 Porter [42] published a stemmer that is widely used and that became a standard algorithm used for English stemming. As an example, the word *waits* (present, 3rd person, singular) when stemmed becomes *wait*.

To use this algorithm we relied on the Natural Language Toolkit [23], a leading platform for building Python programs that work with human language data. We updated our algorithm to use this library, and we checked the results by identifying the true positives and true negatives to calculate the efficiency of each algorithm. As we can see through Tables 4.1 and 4.2 we managed to increase the number of true positives by 10 and reducing the number of true negatives to 0. On the negative side, the number of false positives increased, mostly because of the identification of the verb *compute* when the word *computer* appears, i.e., the stem of the verb *compute* is *comput* and the stem of the word *computer* is also *comput*, so the algorithm thinks that the word *computer* is a verbal form of the verb *compute*. Taking into account the increased number of true positives we decided that this was a balanced approach. Accordingly, we used this algorithm in all scrapers.

	Test Positive	Test Negative
Verb Present	22	0
Verb Absent	12	7
Efficiency	**71%**	

Table 4.1: First algorithm

	Test Positive	Test Negative
Verb Present	32	8
Verb Absent	1	0
Efficiency	**78%**	

Table 4.2: Second algorithm - Using stemming

It should be noted that this automated population is not primordial to this thesis, i.e., we decided to make this automated population to evaluate if the model fits the desired goals in terms of description of the educational services. In a real case scenario of the model the main idea is that the actual teacher is responsible for selecting which category of each dimension of the Bloom's taxonomy is present in a curricular unit. Given this, it is clear that the development of a more complex natural language processing algorithm that would produce even better results in the current work was not primordial to this thesis. This is the main reason why we were pleased with the results of the second algorithm.

The full RDF code for these instances scraped from Udacity can be found in *automated population/udacity.ttl*.

4.3.3 edX Scraper

With the edX scraper we did not face new challenges, we only had to develop and run the same routine as we did with previous scrapers: finding patterns in the course catalogue to be able to access all existing courses, and finding patterns in the course web page to be able to extract the desired data from all courses.

The full RDF code for these instances scraped from edX can be found in *automated population/edx.ttl*.

4.3.4 Coursera Scraper

The Coursera website brought us a new challenge: JavaScript content loading, i.e., the HTTP response carries a web page similar to the one in Figure 4.5. So, once again, we had to get to know better what kind of solutions Scrapy provided, in this case related to the content load of a web page through JavaScript. We found two solutions: *WebKit* and *Selenium*. Both of these frameworks are open source web browser engines that can be embedded in other applications, to be used as a general-purpose display and interaction engine. Although both of them can be used in Scrapy, we only managed to use the Selenium package, which solved our problem.

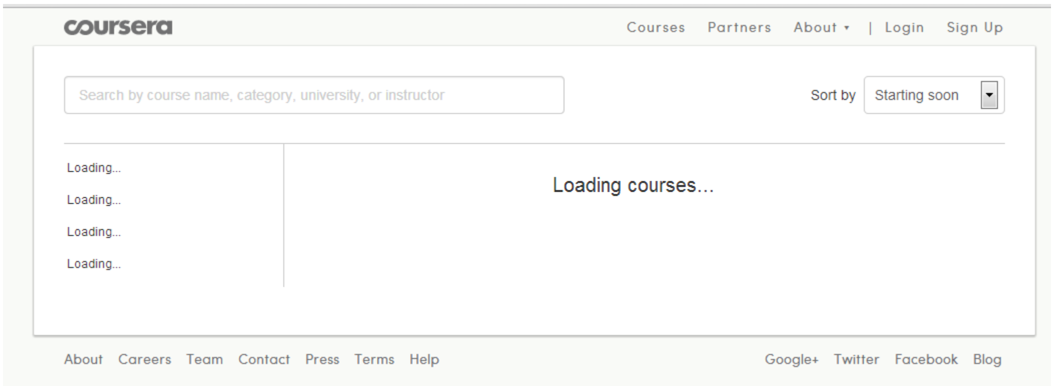

Figure 4.5: Coursera JavaScript challenge

Basically, we added a block of code, shown in Listing 4.9 in the *sipder middleware*, which is one of the components in the Scrapy architecture. This component enables to plug custom functionality to process the requests that are sent to *Spiders*. In our case, we implemented the Selenium web driver, which basically opens a new browser window and will let that browser load the content of the page, hence loading the JavaScript content. Subsequently, we send that content back to the spider to be processed as normal. Selenium provides an API to "drive" the browser natively as a user would do locally.

Listing 4.9: Coursera Middleware using Selenium

```
class WebkitDownloader( object ):
    def process_request( self, request, spider ):
        if( type(request) is not FormRequest and spider.name=="coursera"
            ):
            self.selenium = webdriver.Firefox()
            sel = self.selenium
            sel.get(request.url)
```

```
            if sel:
              #Wait for javascript to load in Selenium
                if request.url=="https://www.coursera.org/courses":
                    time.sleep(50)
                else:
                    time.sleep(10)
            renderedBody = str(sel.page_source.encode("ascii", "ignore"
                ))
            sel.quit()
            return HtmlResponse( request.url, body=renderedBody )
```

In our code we only check if the *Spider* name is "coursera" because we only need to load the content through Selenium in this spider, since the others spiders do not needed. We also check if the page being loaded is the page of the course catalogue or not, because the courses' catalogue web page needs to load 386 courses through JavaScript, which takes longer than loading a single course web page.

The full RDF code for these instances scraped from Coursera can be found in *automated population/coursera.ttl*.

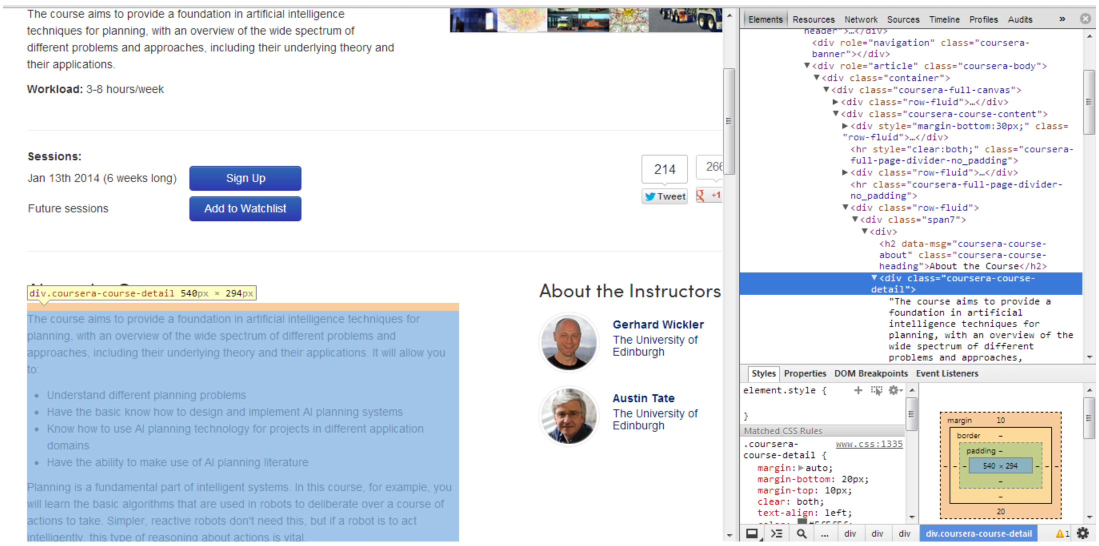

Figure 4.6: Coursera course website

4.3.5 NONIO - UC

This scraper was the one who took more time to develop, and this had several reasons: we need to authenticate in order to access the content of this platform; the links of the courses and course units inside the platform are generated for each request, i.e., if we crawl a web page and subsequently crawl another one the links provided by the first web page will no longer be available; there are a lot of web pages between the login page and the final web pages, i.e., the web page of the courses and the course units. As an example about this last challenge, we had to go through the following web pages, as shown in Figure 4.7 to be able to access each degree and its curricular units. So we had to develop parses for each page scraped.

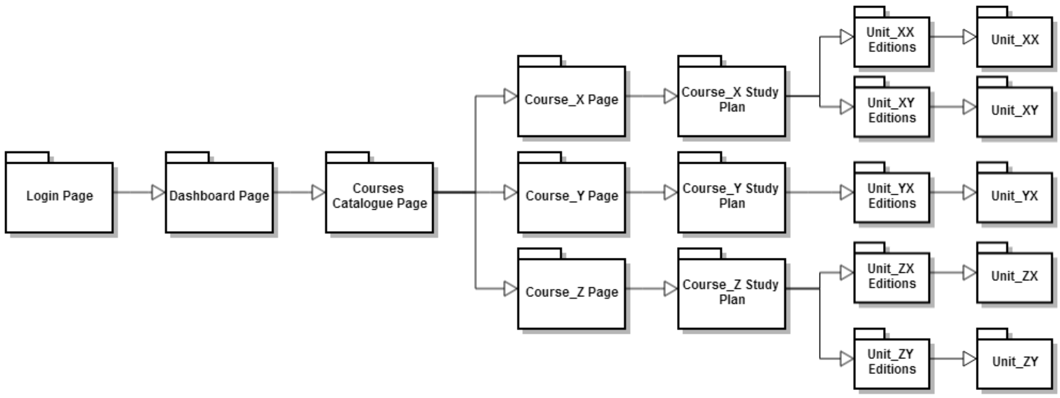

Figure 4.7: Scrapy NONIO - Request flow

As to the first challenge, considering to authentication, we used valid credentials to access NONIO and, once again, discovered more interesting features that proved the power of Scrapy. Scrapy provides a class called *FormRequest*, which enables to send POST/GET HTTP Request as shown in Listing 4.10.

Listing 4.10: NONIO Scrapy Spider

```
class NonioSpider(BaseSpider):
    name = 'nonio'
    start_urls = ['https://inforestudante.uc.pt/nonio/security/init.do']
    allowed_domains = ['inforestudante.uc.pt']

    cursos_domain = "https://inforestudante.uc.pt/nonio/cursos/"
    def parse(self, response):
        print "Gonna send login"
        print response.url
        return [FormRequest.from_response(response,
                    formdata={'method' : 'submeter' ,'username': 'jduro@student.dei.uc.pt', 'password': '********'},
                    callback=self.after_login, dont_filter=True)]
```

After we have successfully authenticated into the NONIO platform we developed parsers to extract the consecutive links in order to get information about the courses and their curricular units.

In order to go through the second challenge we previously mentioned, we had to run the scraper one time per degree so that the links in previous pages did not get outdated. We decided to only extract information about the courses available in the Department of Informatics Engineering since we only had educational areas about the ACM Computing Classification System.

In total we scraped seven different degrees, all from the Department of Informatics Engineering:

- Bachelor and Master Degrees in Design and Multimedia
- Bachelor and Master Degrees in Informatics Engineering
- Master of Software Engineering
- Master in Informatics and Software Project
- Doctoral Program in Information Science and Technology

For each degree, we extracted each curricular unit belonging to the curricular plan. The RDF code of all the degrees extracted from NONIO can be found in *automated population/nonio_[acronym].ttl*.

4.4 Statistics and Results

In the development phase of each scraper we manage to retrieve some results about the number of verbs that were identified by the algorithm developed. With this statistic we believe we could infer if the teachers are developing their curricular objectives with the verbs that establish the linkage between the natural language and the categories of each dimension (mainly the Cognitive dimension) of the Bloom's taxonomy. Many of the teachers have possibly never heard of this taxonomy but still use the verbs, while others, being familiar with the taxonomy, use the verbs believing that they better capture the expressiveness of what is intended by those curricular objectives.

We will start by showing the results to the online platforms that were scraped: Udacity, edX and Coursera.

	Udacity	edX	Coursera
Number of Courses	25	33	339
Verbs Identified Cognitive Dimension	42	159	1296
Verbs Identified Knowledge Dimension	1	15	114
Educational Areas Identified	71	191	1875

Table 4.3: Number of verbs identified by the algorithm

In Table 4.3 we can see the absolute numbers of what was identified by the algorithm. We can see that in these online platforms we have more verbs than courses, i.e. there are courses with several verbs establishing the linkage to the Bloom's taxonomy, but this number is apparently shorter in terms of verbs that link to the knowledge dimension. This can be explained by the fact that each category of the knowledge dimension has only one verb per intersection with the cognitive dimension, i.e. each category of knowledge dimension has only 6 verbs.

	Number of courses with **0** verbs identified	Number of courses with **more than 0** verbs identified
Udacity	7	18
edX	0	33
Coursera	50	289

Table 4.4: Comparison of courses by number of verbs identified

In Table 4.4 we can visualize how many courses have zero verbs identified and how many got at least one verb. This way we can be sure if the verbs are being used across an entire platform. Considering the edX online platform we observe that 100% of the courses have at least one verb identified. Since the new version of the algorithm is able to identify almost all verbs, we can conclude about the courses that do not have any verb identified. We can see that in Udacity there are 7 courses (28%) that do not use any verb. These values are higher than expected, but we believe that with the introduction of a model such as ours, it is possible to increase even more these values and, thus, standardize the curricular objectives to better represent what is intended by the people involved in this process.

We did this analysis to the courses extracted from the NONIO platform (University of Coimbra student platform), as we can see in Table 4.5, but in this case, since we only worked with verbs in the English language, we can only look at the values of the courses that have their descriptions in English, which were only 45 of a total of 171.

	NONIO-UC
Number of Course Units	171
Verbs Identified Cognitive Dimension	239
Verbs Identified Knowledge Dimension	16
Educational Areas Identified	1092

Table 4.5: Number of verbs identified by the algorithm in the Department of Informatics Engineering

	Number of course units with **0** verbs identified	Number of course units with **more than 0** verbs identified
NONIO	5	40

Table 4.6: Comparison of courses by number of verbs identified in the Deparment of Informatics Engineering (English descriptions only)

The results for those 45 course units that have their descriptions in English are shown in Table 4.6. Despite the small population of course units, we can see that the teachers that describe their course units in English tend to use the verbs that better describe the categories of the Bloom's taxonomy. This is a good result because there are only two course units that have 1 verb identified, i.e. there are several course units that have a good amount of verbs, as the courses that have a description in English have, on average, five verbs identified.

4.5 Final Remarks

The evaluation of our model was done through the usage of two approaches: a manual population and a automated population. In the first approach we tested how would the model be populated by hand simulating, for example, a professor creating the description of his own curricular unit, whilst in the second approach we tested if the model is able to be populated by algorithms. In this second approach we had to be very careful about the verbs that identify each category of the Bloom's taxonomy in order to be able to identify these categories in the extracted curricular units.

Overall, we achieved positive results because we managed to identify the contents of each course into the service model.

5
Tool Implementation

This chapter presents the web application that was developed to provide an user interface to visualize instances of the *Linked usdl4edu* model by means of tables and graphs, which enables the retrieval of useful information to help answer today's questions. This tool also can be used for the evaluation of the model.

In the first section we provide information about the implementation of this tool and, in the second section, we provide an example of the application of this tool.

5.1 Implementation

Upon concluding our model we decided to develop a tool that would facilitate its usage and showed what can be done with it. The application that was developed is freely available in a public repository, in `https://github.com/jduro/esoteric/tree/master/webplatform`, where it is possible to see and use the entire developed code.

We decided to build a web application given its advantages when compared to client applications, such as: no need to install software to run the application, centralized data, amongst others. To develop the web application we used the Ruby on Rails language for three main reasons:

- Ruby uses the Model, View, Controller (MVC) architecture, and the language itself resembles the Python language, which makes it very intuitive.
- Ruby relies on user created *gems*, which look similar to packages of developed code to add functionality to the web application. This allows to add such functionalities with almost no effort at all.
- We wanted to increase our Ruby on Rails knowledge and, for this reason we developed, for the first time a Ruby on Rails web application, which was very rewarding.

As we mentioned before, ruby makes use of the Model, View Controller Architecture - as shown in Figure 5.1. When a ruby project is created, a directory with all these directories and files is created, that corresponds to the *MVC* architecture. The view classes will have the HTML code blended with some ruby code that will have the visualization, the models will have the mapping to a database, whilst the controller will be responsible for the processing. If a given action is requested to the application, it is forwarded to a certain defined function in the controller, which will be responsible to process that request, after which it is redirected to the user, the view that corresponds that action. For example, in our application, if the user requests a visualization over a service that has a specific *id* it is forward to the *info* function of our controller. This is defined in the *Rails router* with this code line ***match "/service/:id" =>"welcome#info"***. After that, in the controller, we do all the processing necessary, in this case, we access the ontology file of that service and extract the information necessary to produce the graphs that are presented to the end user. All functions - which represent actions in Ruby - are implemented in the main controller in */app/controllers/welcome_controller.rb*.

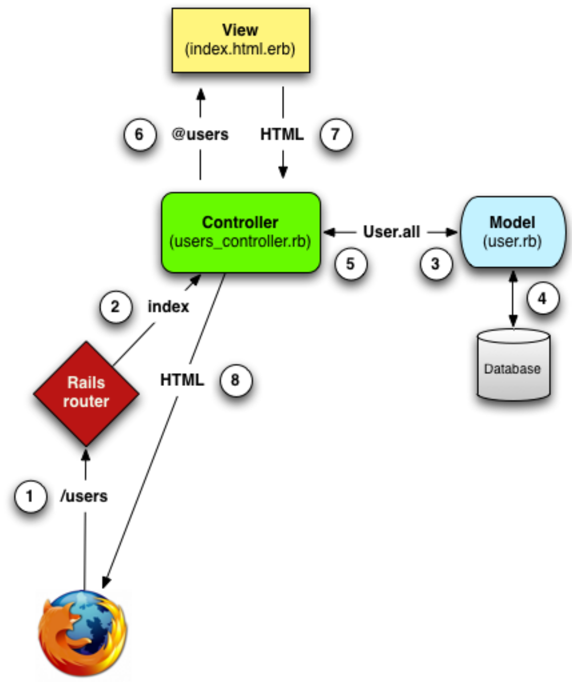

Figure 5.1: Model, View, Controller Architecture of Ruby on Rails

One of the first steps was to add Twitter Bootstrap, which is a free collection of tools for creating websites, containing HTML and CSS-based design templates for typography, buttons, forms, among others, and also JavaScript components to increase the user interface experience.

When we started developing the application we faced the following question: should the web application convert the data from the model to a database (relational model) to be accessed by the application? The decision was that we would not use a database to store the instances of the *Linked USDL4EDU* model, because if we convert the data from the ontology to a relational model that could imply that some semantic meaning could be lost in the process. In order to overcome this challenge (a challenge because Ruby is built to use a database, and everything is easier when we use a database), we decided to allow users to upload their *Linked USDL4EDU* files into the application. Therefore, when a user uploads a file, it is processed in the controller, which will read the content of that file and will parse each educational service that it is found. Subsequently, the file is saved in the public folder (*/public/services/upload/*) of Ruby to be accessed later. For each educational service, a number of details of that service is saved in the database, for example, the title, file path, URL of that service in the ontology (to be able to identify the service in the ontology), and the organization where it was extracted from, among others. These pieces of information saved in the database allow the user to browse the existing services without accessing files continuously. Everything that is part of viewing services are loaded dynamically from the files that were uploaded and stored in the service side of the application.

According to the models created, Ruby uses the concepts of migrations in order to manage its own databases. In total, the web application makes use of 5 tables:

Service this table stores information about the uploaded educational service.

Unit this table stores the units that are not services, i.e., the units that makes part of a certain degree's curricular unit.

Educational Contexts this table manages all the Educational areas that were extracted by the ACM scraper.

Service has Contexts this table establishes the relation of *many to many* between the existing services as their educational areas

Unit has Contexts this table is similar to the one before, but establishing the relation from the Unit table instead.

In Listing 5.1 it is shown an example of the code developed to create a new migration or, in other words, a new table.

Listing 5.1: Create Services table

```
class CreateServices < ActiveRecord::Migration
 def change
  create_table :services do |t|
   t.string :title
   t.string :url
   t.string :urlCourse
   t.string :path
   t.string :organization
   t.boolean :isCourse, :default => false
   t.boolean :haveInfo, :default => true
   t.integer :cogn, :default => 0
   t.integer :know, :default => 0
   t.timestamps
```

```
    end
  end
end
```

In order to parse the Turtle files that include the *Linked USDL4EDU* instances we used the *rdf* and *rdf-turtle gems* available for Ruby. These *gems* enable to access the data through queries, which return the data in the form of arrays.

After we implemented the upload feature of the web application, we started developing features more related to the visualization of the services instantiated with our *Linked USDL4EDU* service model. In this application it is possible to:

Browse the list of uploaded services, whether they are degrees or units. If a service is a degree, it is possible to browse the available units in that degree.

View a specific service, whether it is a degree or a unit. We implemented different pages for each one (degree or unit), because a degree is usually composed by a number of units, so it is important to have a different visualization.

Select different degrees or units and obtain a visualization of that selection.

Similar Services that share educational areas and similar categories of the Bloom's taxonomy with the selected service, i.e. the services that are more likely to be similar to the selected degree/course.

Download the service description of the course/degree that is being visualized.

In total, 4 different views were built, namely concerning: a degree, a course that is part of a degree's curricular plan, a course and, the selection of services.

After all these features were implemented we decided that it would be important to provide better ways of visualization such as, graphs and tables. To add tables in web pages it is relatively easy, since HTML has a tag for building tables, but in terms of graphs we had to search for one that worked with Ruby.

We decided to use *HighCharts*, which is a JavaScript library for building charts, because there is a *gem* available that works with HighCharts, called LazyHighCharts [55]. In order to add each chart we looked into the *Competency Questions* created in Section 4.1. We decided to implement graphs that might be important to answer or help answering some of the questions we created to help us evaluate and demonstrate the applicability of our model in terms of real world usage. All these graphs can be exported as image files to be saved locally by the user. This *gem* was a good example of the power of Ruby, because by simply installing the *LazyHighCharts gem* we avoided all the JavaScript configurations necessary to use *HighCharts*. We only needed to install the *gem*, and add a code line in the JavaScript file of the Ruby project, to let Ruby know that the project is using that JavaScript *gem*. Then, we needed to understand how to build the graphs, which was very intuitive with the Ruby language.

5.2 Educational Areas and Linked USDL4EDU model

Regarding to the educational areas, we manage to implement code to parse all the extracted educational areas from the file. This code was implemented in a new file create in the */config/initializers/* directory. All the classes implemented in this directory will be executed when rails server starts, this way, we only parse the educational areas into the web application when the server is restarted. In this file, we parse each educational area and add them to the models (database), in case it do not exist already.

We needed to access the *Linked USDL4EDU* file in order to be able to understand some instances, such as: the bloom's taxonomy instances, languages, mode of delivery, among others. So, in order to avoid being accessing the *Linked USDL4EDU* file continually for each request, we implemented the code to extract these instances from the *USDL4EDU* file in the */config/application.rb*.

5.3 Using the Tool: An example

In this section we present the web application we developed focusing on some graphs that are reproduced by the application itself. In Figure 5.2 we can see the appearance of the web application, showing, in this case, a course from Udacity.

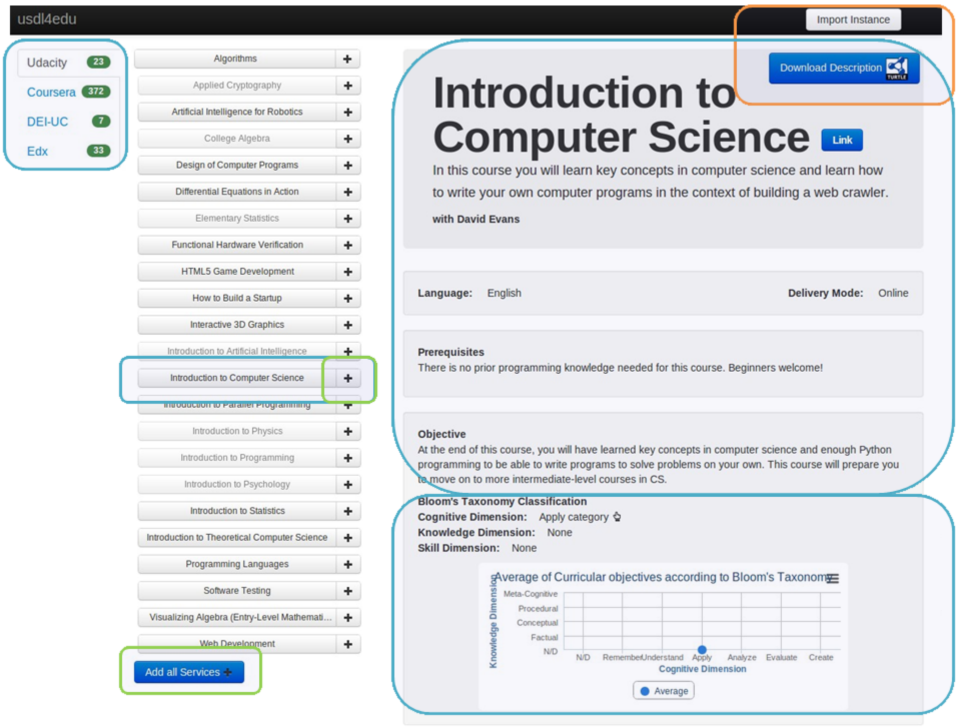

Figure 5.2: The application shows the course Introduction to Computer Science from Udacity

In the left side of the web application is the *browsing* area, where the user can browse through the uploaded services, by selecting the data sources, which are the organizations from which we extracted the data in the previous chapter. The user can view a given educational service by clicking on it, or can add a certain service to the visualization where other services can be aggregated in the graphs - this can be done by clicking in the + sign, or in the button bellow (**Add all Services**). In the right side we can find the visualization of the course, which is spread through the next two images. In the first boxes (name, language, prerequisites, objectives, among others), it is possible to view the information that was extracted about each educational service. However, all the information bellow that, results from the algorithms that were implemented. In this case, we can see the average of the objectives of this course, according to the Bloom's taxonomy. In Figure 5.3, in contrast, we can see each identified objective. This visualization is achieved through a graph and text. This visualization answers the competency questions **Q4** and **Q5**, which refer to the questions provided by the teacher and the student, in Section 4.1.

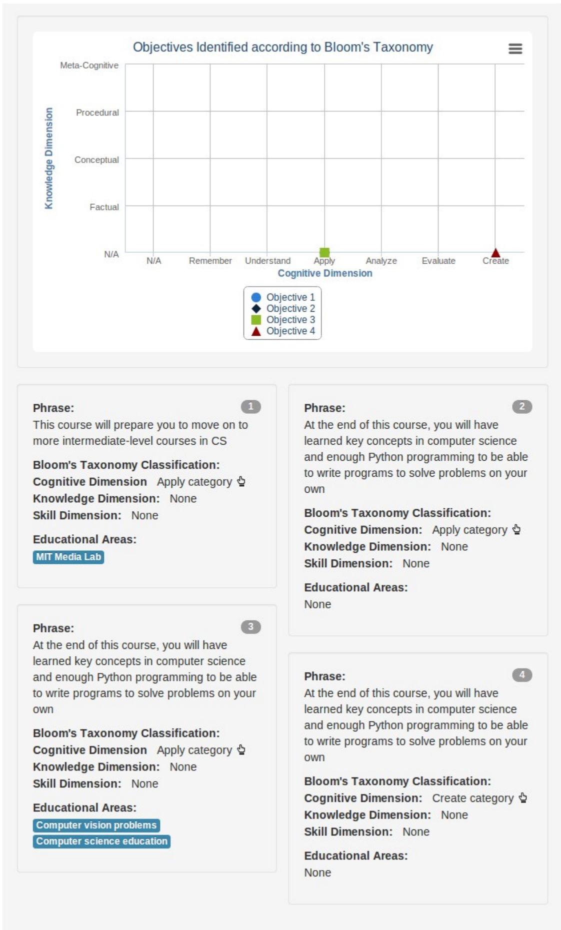

Figure 5.3: Objectives of the course Introduction to Computer Science from Udacity

In Figure 5.4 we can see the services the web application indicated as similar to the service that is being viewed. In contrast with the previous images, this information is calculated by the web application, whilst the other two images present in the model were calculated by the scrapers. The similar services are selected by comparing the values of the Bloom's taxonomy and the number of common educational areas. The returned services are the ones that are more likely to be similar to the one selected. In this case, *"Computer Science 101"* from *Coursera* shares with *"Introduction to Computer Science"* from *Udacity* the educational area *"Computer science education"* and the *cognitive* category *Apply*.

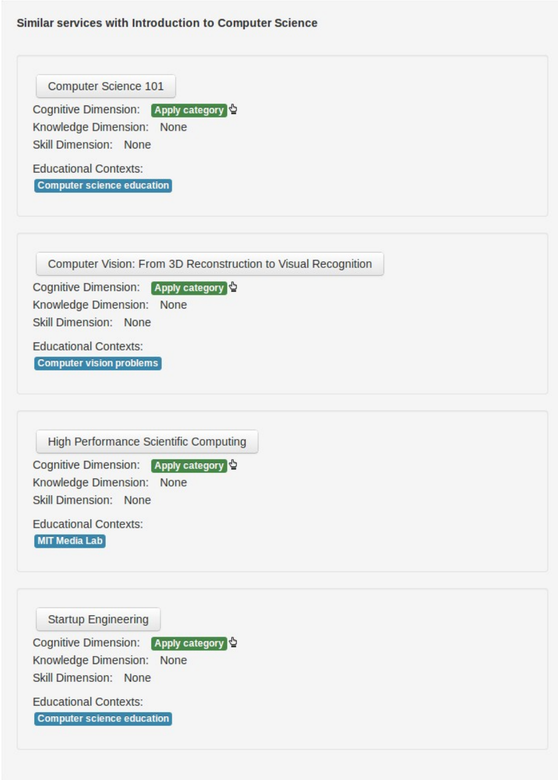

Figure 5.4: Services similar to Introduction to Computer Science are shown in the application

Figure 5.5 shows the graphs that are presented when the user is viewing information about a given degree. In this case the graphs refer to the Master in Informatics Engineering from the University of Coimbra. As we can see, different graphs are presented to the user, depending on what is being visualized. This graph provides information about the classification of each course unit available in that degree, i.e. it is possible to infer where to this course is leaning, according to the categories of the Bloom's taxonomy. In this case, we can see that there are two major clusters whit a large amount of units: the *Apply* and *Analyse* categories, from the cognitive dimension with an undefined value to the knowledge dimension (*N/D*). This kind of graph can help answer to the following competency questions: **Q1**, **Q2**, and **Q3**, which are the questions from the European Union Minister, education Minister of Portugal and the university dean, available in Section 4.1. These *N/D* (not defined) values come from the scraper algorithm, which was not able to detect a verb that could define a value for the Knowledge Dimension of the Bloom's taxonomy.

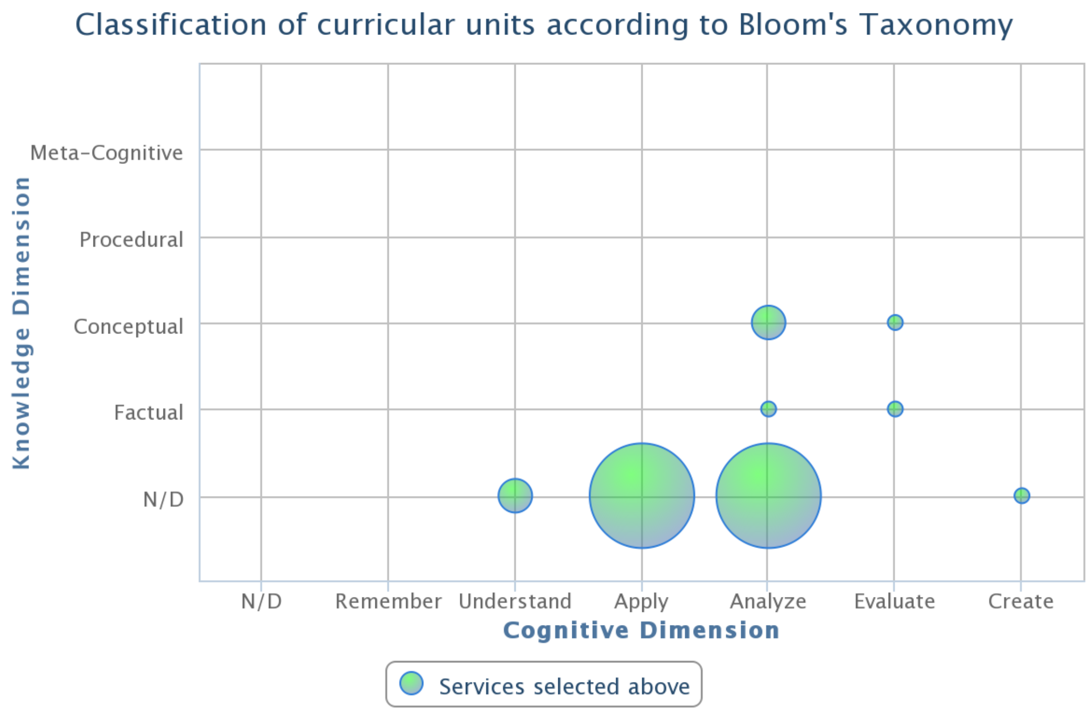

Figure 5.5: Master in Informatics Engineering from University of Coimbra

Figure 5.6 presents a graph similar to the one presented before. The current graphs reveals the courses selected by the user , i.e. it demonstrates that the user has the possibility to select different courses and join them in a given graph. In this case, we aggregated the Master and Bachelor degrees in Informatics Engineering from University of Coimbra, to see if the curricular units from these degrees lean to the same values of the Bloom's taxonomy. From the small changes between the graphs in terms of disposition of the most common values, we can conclude that both courses are similar and well articulated, since they trend for similar categories in terms of the Bloom's taxonomy dimensions.

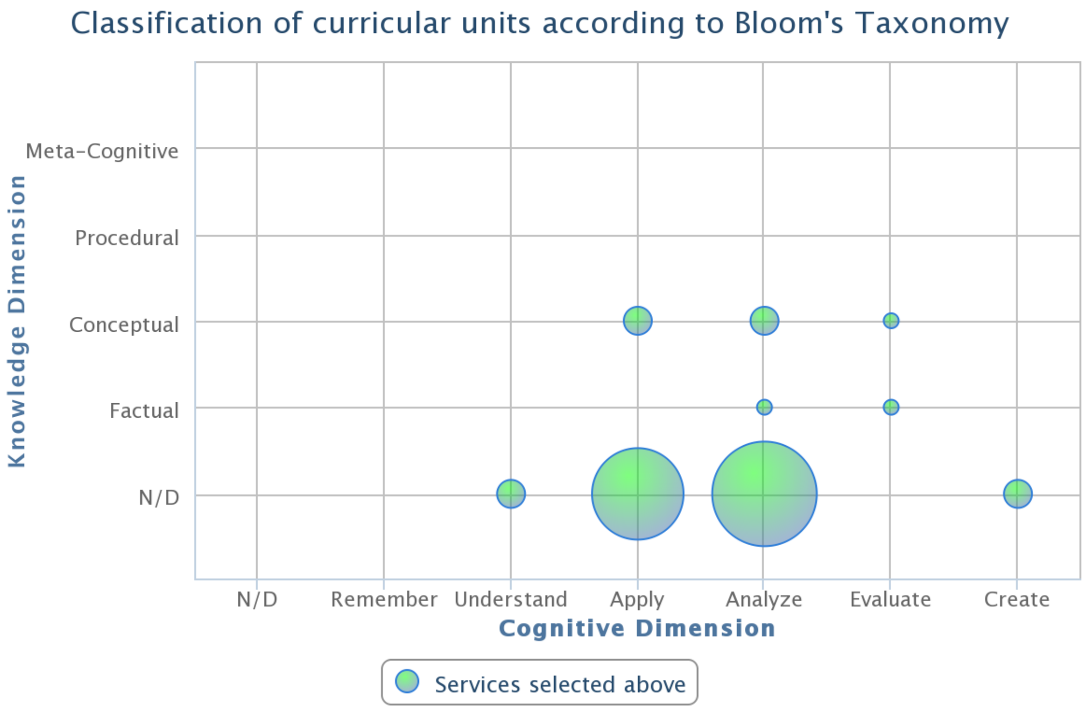

Figure 5.6: Bachelor and Master's degree in Informatics Engineering from University of Coimbra

5.4 Comparing Degrees

In this section we will provide an example of what this tool and the Linked *USDL4EDU* model can achieve in terms of society questions. In this case, we will compare three similar courses, all in the field of artificial intelligence, but from different platforms and from different locations. The courses we used in this comparison are:

Artificial Intelligence from Berkeley University, hosted in *edX* platform (Figure 5.7).

Artificial Intelligence Planning from The University of Edinburgh, hosted in *Coursera* platform (Figure 5.8).

Artificial Intelligence for Robotics from Stanford University, hosted in *Udacity* (Figure 5.9).

This scenario could be a solution to a real case question - as the one we presented in Section 4.1. For example, a deputy from the European Union might be interested in retrieving information about weather a certain course is able to educate learners to innovate and create new ideas in a given area. As mentioned before, this can be achieved by comparing the values on the cognitive dimension. The deputy could select the courses and add them to the selection in order to aggregate them in a given graph, or may view them individually and then interpret each graphic.

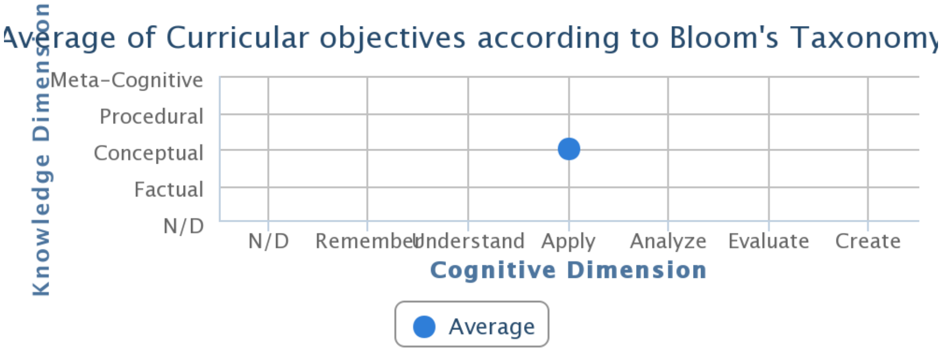

Figure 5.7: Course objective for Artificial Intelligence - edX

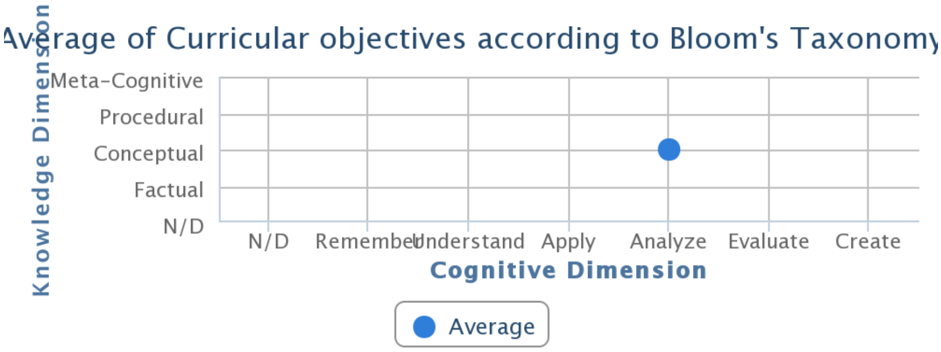

Figure 5.8: Course objective for Artificial Intelligence for Robotics - Coursera

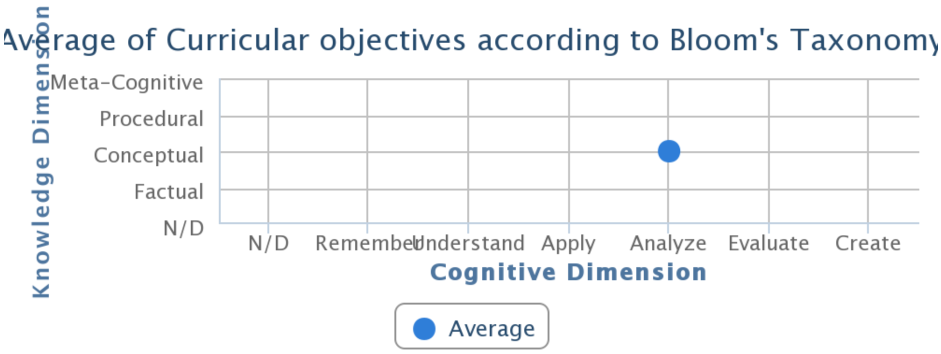

Figure 5.9: Course objective for Artificial Intelligence for Robotics - Udacity

In this case, we decided to extract the graphs from the individual visualization of each course. Despite the similarity of these courses, they have different values in the Bloom's taxonomy dimensions. If we look carefully, we can observe that both courses from California universities (Cousera and Udacity) have the same *Analyse* value on the cognitive dimension and the *Conceptual* on the knowledge dimension, whilst the course from the United Kingdom (edX) has the same (or almost the same) course but with a smaller value on the cognitive dimension - *Apply* - and the same value on the knowledge dimension.

These graphs were directly extracted from the web application, and can show us the impact of such tool when aligned with our educational model. It becomes possible to compare and analyse courses and degrees from different places with a standard model such as Linked *USDL4EDU*, which tries to standardize the curricular objectives with a reasoned and widely used taxonomy.

6 Conclusions

This chapter presents the conclusions of our work and suggests possible future work to be done.

The first section briefly summarizes the contents of this thesis and the approach that we followed. The second section discusses our findings regarding the results we achieved. The third section describes how this work may benefit our society. Lastly, the fourth section suggests some future work possibilities to improve the algorithms used and its web application.

6.1 Summary

Our main goal in this thesis was to conceive a computer readable model for describing educational services, so as to prove that a domain-specific model can catch specificities that a standard model can not. Our model is focused on curricular objectives or learning outcomes relying, for that purpose, on the Bloom's taxonomy. We concentrated on the curricular objectives because we believe that this property is fundamental to understand the focus of each degree or course in terms of the knowledge it passes to learners. By splitting the Bloom's taxonomy into three different dimensions (Cognitive, Knowledge and Skills), we were able to retrieve enough information about the degree or course in order to produce tables and graphs that helps answer important societal questions. We decided to use the Bloom's taxonomy because, nowadays, it is considered fundamental to the creation and description of curricular objectives due to its long history and popularity, it has been condensed, expanded, and reinterpreted in a variety of ways. Research findings have led to the discovery of a variety of interpretations and applications for this taxonomy [20].

In our state of the art review we discussed, and presented, some models that in somehow helped us to develop our model, and we gathered information about service description. The Bloom's taxonomy was the model that better fitted our requisites, which proved to be a good solution to describe the learning objectives.

In order to demonstrate the usefulness of our model in real world usage, and to be able to do a formal evaluation of the model, two approaches were developed: one where we manually populated the model through curricular unit records, and the other where we developed scrapers to extract the content of web sites to populate the model.

Lastly, we developed a web application to provide additional value to the model itself, by means of visualizing the instances (degrees and courses). We also added to the interface graphs and tables. These can help retrieving information about the courses that are being visualized, which combined with the fact that the user is able to select courses and degrees and perform a visualization of the selection, provides the means to better compare courses.

6.2 Findings

From the evaluation processes that were preformed we found encouraging results, which suggested that *Linked USDL4EDU* is a reasonable solution to describe educational services, at least when the focus on the curricular objectives. We also found that the developed web application produced positive improvements to the model, by providing different visualizations over the instances.

Our research, in what concerns the state of the art and Chapter 3, focused on providing a solid background and evaluation processes to increase the model's strength.

6.3 Implications for society

With our standard model for educational services we hope to provide a way to standardize these types of services, i.e. we provide a way for everyone, from universities to open online courses, to have their courses available in a computer readable model that can be shared world wide. All organizations will be able to understand each others' degrees and courses if they all use such a model, which provides ways to trade and share courses, and this should promote the development of European degrees with curricular units spread across Europe's universities.

Since we focused on the curricular objectives, having relied on Bloom's taxonomy, this model can also standardize the curricular objectives by encouraging the usage of the verbs that better identify the categories of the Bloom's taxonomy, thus providing improved and easier ways to describe curricular objectives. This way, the curricular objectives can be easily understood and compared across curricular units.

The model combined with a tool such as the one we developed, can help to understand if in a given country or region the degrees that are available encourage students to new product and innovation or the development of novel ideas in general just by checking the values of the Cognitive dimension of those degrees.

Overall, this model can be combined with numerous tools and be used by numerous organizations to bring such organizations closer to each other and, in particular, to bring universities from across the world closer to each other by sharing their courses under a standard model.

6.4 Future work

One important aspect that can be improved is the algorithm that identifies the categories of the Bloom's taxonomy in the extracted objectives of each course. The algorithm developed is not perfect. The implementation of some natural language processing algorithms could provide better statistics about the usage of the "correct" verbs in the curricular objectives, providing an encouragement to the usage of these verbs in order to standardize the objectives.

Another feature that we decided not to implement and could be useful is an interface to create instances of *Linked USDL4EDU* model and attach this feature to the developed web application. This feature would help people to create their own instances of our model and would help that people to understand the model itself. We decided to not implement this feature because it was not a major achievement in terms of validating the model, but in terms of future work we believe it is important because the user could then use the application to export the RDF file. This interface could also allow to introduce changes to existing instances in the web application.

Bibliography

[1] Amorim, R., Lama, M., Sánchez, E., Riera, A., and Vila, X. A learning design ontology based on the ims specification. *JOURNAL OF EDUCATIONAL TECHNOLOGYAND SOCIETY 9*, 1 (2006), 38.

[2] Anderson, L., Krathwohl, D., and Bloom, B. *A taxonomy for learning, teaching, and assessing.* Longman, 2005.

[3] Barros, A. P., and Oberle, D. *Handbook of service description.* Springer Science and Business Media, New York, N. Y., 2012.

[4] Bixler, B. Instructional goals and objectives. `http://www.personal.psu.edu/bxb11/Objectives/GoalsAndObjectives_print.html`, Jan. 2013.

[5] Bizer, C., Heese, R., Mochol, M., Oldakowski, R., Tolksdorf, R., and Eckstein, R. The impact of semantic web technologies on job recruitment processes. In *Wirtschaftsinformatik 2005*, O. K. Ferstl, E. J. Sinz, S. Eckert, and T. Isselhorst, Eds. Physica-Verlag HD, 2005, pp. 1367–1381. 10.1007/3-7908-1624-8_72.

[6] Bloom, B., Engelhart, M., Furst, E., Hill, W., and Krathwohl, D. Taxonomy of educational objectives: Handbook i: Cognitive domain. *New York: David McKay 19* (1956), 56.

[7] Cardoso, J., Pedrinaci, C., Leidig, T., Rupino, P., and Leenheer, P. D. Foundations of open semantic service networks. *International Journal of Service Science, Management, Engineering, and Technology (IJSSMET). To be published.* (2013).

[8] Cardoso, J., Winkler, M., and Voigt, K. A service description language for the internet of services. In *First International Symposium on Services Science (ISSS'09)* (Leipzig, Germany, 2009). conference.

[9] Cardoso, J., Winkler, M., and Voigt, K. A service description language for the internet of services. In *Proceedings of ISSS* (2009).

[10] Clark, D. R. The art and science of leadership. Retrieved Dec 10, 2012 from `http://nwlink.com/~donclark/leader/leader.html`, 2004.

[11] Cobden, M., Cummings, A., Gibbins, N., and Williams, C. Ecs uri system specification - university of southampton. Retrieved Jun 3, 2013 from `http://id.ecs.soton.ac.uk/docs/`, 2006.

[12] DE KINDEREN, S., AND GORDIJN, J. e3service: An ontological approach for deriving multi-supplier it-service bundles from consumer needs. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual* (jan. 2008), p. 318.

[13] DE LANGEN, F. Education as a service: Moocs, odl and production of knowledge. Retrieved Jun 5, 2013 from http://frankounl.wordpress.com/2013/02/12/education-as-a-service-moocs-odl-and-production-of-knowledge/, 2013.

[14] DECKER, S., MELNIK, S., VAN HARMELEN, F., FENSEL, D., KLEIN, M., BROEKSTRA, J., ERDMANN, M., AND HORROCKS, I. The semantic web: the roles of xml and rdf. *Internet Computing, IEEE 4*, 5 (sep/oct 2000), 63 –73.

[15] DIFALLAH, D. E., DEMARTINI, G., AND WYLOT, M. The bowlogna ontology. Retrieved Jun 3, 2013 from http://diuf.unifr.ch/main/xi/bowlogna, 2011.

[16] DUMAS, M., O'SULLIVAN, J. J., HERAVIZADEH, M., EDMOND, D., AND TER HOFSTEDE, A. H. Towards a semantic framework for service description. In *Data Semantics 9: Semantic Issues in E-Commerce* (Hong Kong, China, 2003), R. Meersman, K. Aberer, and T. S. Dillon, Eds., Kluwer, pp. 277–291. For more information, please refer to the conference's website (see hypertext link) or contact the author. Author contact details: d.edmond@qut.edu.au.

[17] FERRARIO, R., GUARINO, N., JANIESCH, C., KIEMES, T., OBERLE, D., AND PROBST, F. Towards an ontological foundation of services science: The general service model. *Wirtschaftsinformatik, Zurich, Switzerland February* (2011), 16–18.

[18] FERRIS, T., AZIZ, S., ET AL. A psychomotor skills extension to Bloom's taxonomy of education objectives for engineering education. *Exploring Innovation in Education and Research* (2005), 1–5.

[19] FOR COMPUTING MACHINERY, A. The acm computing classification system (ccs). Retrieved Jun 9, 2013 from http://dl.acm.org/ccs_flat.cfm, 2013.

[20] FOREHAND, M. Bloom's taxonomy: Emerging perspectives on learning, teaching and technology. Retrieved Jun 16, 2013 from http://epltt.coe.uga.edu/index.php?title=Bloom%27s_Taxonomy, 2005.

[21] FOREHAND, M. Bloom's taxonomy. http://projects.coe.uga.edu/epltt/index.php?title=Bloom%27s_Taxonomy, 2013.

[22] FRIGG, R., AND HARTMANN, S. Models in science. *Stanford encyclopedia of philosophy* (2009).

[23] Garrette, D., Ljunglöf, P., Nothman, J., Korobov, M., Neergaard, M. M., and Bird, S. Natural language toolkit - nltk 2.0 documentation. http://nltk.org/, Jan. 2012.

[24] Gordijn, J., and Akkermans, J. Value-based requirements engineering: exploring innovative e-commerce ideas. *Requirements Engineering 8* (2003), 114–134. 10.1007/s00766-003-0169-x.

[25] Grüninger, M., and Fox, M. S. Methodology for the design and evaluation of ontologies.

[26] Hoffmann, M. Using Bloom's taxonomy of learning to make engineering courses comparable. In *EAEEIE Annual Conference, 2008 19th* (29 2008-july 2 2008), pp. 205 –209.

[27] Jackson, D., and Newberry, P. *Critical Thinking*. Cengage Learning, 2011.

[28] Jesse. Education as a service. Retrieved Jun 5, 2013 from http://fueled.com/blogpost/education-as-a-service/, 2012.

[29] Kauppinen, T., Trame, J., and Westermann, A. Teaching core vocabulary specification. Retrieved Jun 3, 2013 from http://linkedscience.org/teach/ns/, 2012.

[30] Kinderen, S., and Gordijn, J. Reasoning about substitute choices and preference ordering in e-services. In *Advanced Information Systems Engineering*, Z. Bellahsène and M. Léonard, Eds., vol. 5074 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 390–404.

[31] Krathwohl, D. R. A revision of Bloom's taxonomy: An overview. *Theory Into Practice 41*, 4 (2002), 212–218.

[32] Lopez, M., Gomez-Perez, A., Sierra, J., and Sierra, A. Building a chemical ontology using methontology and the ontology design environment. *Intelligent Systems and their Applications, IEEE 14*, 1 (1999), 37–46.

[33] Mernik, M., Heering, J., and Sloane, A. M. When and how to develop domain-specific languages. *ACM Comput. Surv. 37*, 4 (Dec. 2005), 316–344.

[34] Moore, M., and Kearsley, G. *Distance Education: A Systems View of Online Learning*. What's New in Education Series. Wadsworth Publishing Company, 2011.

[35] Najjar, J., Derntl, M., Klobucar, T., Simon, B., Totschnig, M., Grant, S., and Pawlowski, J. A data model for describing and exchanging personal achieved learning outcomes (palo). *International Journal of IT Standards and Standardization Research 8*, 2 (August 2010), 87–104.

[36] Ng, I. C. L., and Forbes, J. Education as service: The understanding of university experience through the service logic. *Journal of Marketing for Higher Education 19*, 1 (2009), 38–64.

[37] Oberle, D., Barros, A., Kylau, U., and Heinzl, S. A unified description language for human to automated services. *Information Systems 38*, 1 (2013), 155 – 181.

[38] Organisation for Economic Co-operation and Development (OECD). Balance of payments (mei) - Balance on services. http://stats.oecd.org/index.aspx?queryid=429#, Feb. 2013.

[39] Pedrinaci, C., and Leidig, T. Linked usdl. Retrieved Jun 16, 2013 from http://linked-usdl.org/.

[40] Pinto, S. S. *Gestão dos serviços: a avaliação da qualidade.* Editorial Verbo, Lisboa, 2003.

[41] Porter, M. E. The five competitive forces that shape strategy. *Harvard Business Review 86*, 1 (Jan. 2008), 78–93.

[42] Porter, M. F. An algorithm for suffix stripping. *Program: electronic library and information systems 14*, 3 (1980), 130–137.

[43] Razo-Zapata, I., De Leenheer, P., Gordijn, J., and Akkermans, H. Service value networks for competency-driven educational services: A case study. In *6th International BUSITAL Workshop* (2011).

[44] Razo-Zapata, I. S. *Service Value Networks.* Phd thesis, VU University Amsterdam, 2013.

[45] Research, S., and AG, S. Internet of services: Usdl. Retrieved Jun 16, 2013 from http://www.internet-of-services.com/index.php?id=288&L=0.

[46] SAP. *Designed for Your Industry, Scaled to Your Business, Ready for Your Future.* SAP, 2004. http://www.sap.com/romania/industries/Image.pdf Accessed: 03/12/2012.

[47] Shadbolt, N., Gibbins, N., Glaser, H., Harris, S., and Schraefel, M. Cs aktive space, or how we learned to stop worrying and love the semantic web. *Intelligent Systems, IEEE 19*, 3 (may-jun 2004), 41 – 47.

[48] Shimomura, Y., and Tomiyama, T. Service modeling for service engineering. In *Knowledge and Skill Chains in Engineering and Manufacturing*, E. Arai, F. Kimura, J. Goossenaerts, and K. Shirase, Eds., vol. 168 of *IFIP International Federation for Information Processing*. Springer US, 2005, pp. 31–38.

[49] Shvaiko, P., and Euzenat, J. a. a. Ontology matching: State of the art and future challenges. *Knowledge and Data Engineering, IEEE Transactions on 25*, 1 (jan. 2013), 158 –176.

[50] Silvestro, R., Fitzgerald, L., Johnston, R., and Voss, C. Towards a classification of service processes. *International Journal of Service Industry Management 3*, 3 (1992), 62–75.

[51] Styles, R., and Shabir, N. Academic institution internal structure ontology (aiiso). Retrieved Jun 3, 2013 from `http://vocab.org/aiiso/schema`, 2008.

[52] Team, R. Rdflib documentation. Retrieved Jun 10, 2013 from `https://rdflib.readthedocs.org/en/latest/`, 2009.

[53] van Deursen, A., Klint, P., and Visser, J. Domain-specific languages: an annotated bibliography. *SIGPLAN Not. 35*, 6 (June 2000), 26–36.

[54] Wang, L., von Laszewski, G., Younge, A., He, X., Kunze, M., Tao, J., and Fu, C. Cloud computing: a perspective study. *New Generation Computing 28* (2010), 137–146. 10.1007/s00354-008-0081-5.

[55] Xiao, D., and Michelson, M. Lazyhighcharts - official gem repository. Retrieved Jun 16, 2013 from `https://github.com/michelson/lazy_high_charts`, 2013.